

AMERICAN UNIVERSITY OF BEIRUT

A NOVEL SPATIO-TEMPORALLY  
ADAPTIVE PARALLEL  
THREE-DIMENSIONAL DSMC SOLVER FOR  
UNSTEADY RAREFIED MICRO/NANO GAS  
FLOWS

by

MIRVAT OMAR SHAMSEDDINE

A dissertation  
submitted in partial fulfillment of the requirements  
for the degree of Doctor of Philosophy  
to the Department of Mechanical Engineering  
of the Faculty of Engineering and Architecture  
at the American University of Beirut

Beirut, Lebanon  
December 2018

AMERICAN UNIVERSITY OF BEIRUT


A NOVEL SPATIO-TEMPORALLY  
ADAPTIVE PARALLEL  
THREE-DIMENSIONAL DSMC SOLVER FOR  
UNSTEADY RAREFIED MICRO/NANO GAS  
FLOWS

by

MIRVAT OMAR SHAMSEDDINE

Approved by:

\_\_\_\_\_  
Dr. George Turkiyyah, Professor  
Computer Science



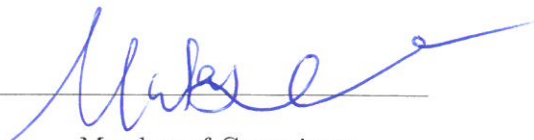
Committee Head

\_\_\_\_\_  
Dr. Issam Lakkis, Professor  
Mechanical Engineering



Advisor

\_\_\_\_\_  
Dr. Mu'Tasem Shehadeh, Associate Professor  
Mechanical Engineering



Member of Committee

\_\_\_\_\_  
Dr. Leila Issa, Assistant Professor



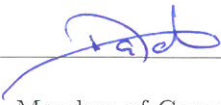
Member of Committee

Applied Mathematics

---

Dr. Nadim Diab, Assistant Professor

Mechanical Engineering



Member of Committee

*disertation*  
Date of thesis defense: December 14, 2018

# AMERICAN UNIVERSITY OF BEIRUT

## THESIS, DISSERTATION, PROJECT RELEASE FORM

Student Name: Shemseddine Harvet Omer  
Last First Middle

Master's Thesis       Master's Project       Doctoral Dissertation

I authorize the American University of Beirut to: (a) reproduce hard or electronic copies of my thesis, dissertation, or project; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes.

I authorize the American University of Beirut, to: (a) reproduce hard or electronic copies of it; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes after: **One**  **year from the date of submission of my thesis, dissertation or project.**  
**Two** \_\_\_ years from the date of submission of my thesis, dissertation or project.  
**Three** \_\_\_ years from the date of submission of my thesis, dissertation or project.

Harvet March 19, 2019  
Signature Date

This form is signed when submitting the thesis, dissertation, or project to the University Libraries

# Acknowledgements

Hereby, I want to thank all those who have contributed to this dissertation and supported me all along the way. First and foremost, I am truly indebted and thankful to my advisor, Prof. Issam Lakkis, for his invaluable assistance, support, and great motivation throughout my dissertation research and at all times. I am confident this success would not have been possible without his guidance and inspiration. Besides, I am also sincerely grateful to all committee members for their invaluable comments, useful tips, and enlightening discussions. Last but not least, my sincerest love and gratitude go to my beloved family and my husband, for their understanding and endless love, through the duration of my studies.

# An Abstract of the Dissertation of

Mirvat Omar Shamseddine for Doctor of Philosophy  
Major: Mechanical Engineering

Title: A Novel Spatio-Temporally Adaptive Parallel Three-Dimensional DSMC Solver for Unsteady Rarefied Micro/Nano Gas Flows

An efficient parallel multi-scale direct simulation Monte Carlo algorithm to simulate three-dimensional rarefied gas flows over complex geometries is presented. The proposed algorithm employs a novel spatio-temporal adaptivity scheme. Based on the gradients of flow macro-properties, the spatio-temporal adaptivity scheme computes the cell size distribution and assigns the appropriate number of time sub-steps for each cell. The temporal adaptivity scheme provides local time step adaptation through different temporal levels employed in different cells. Spatial representation is based on a hierarchical octree Cartesian grid with low memory storage requirement. The hierarchical octree grid endows the method with straightforward and efficient data management suitable for particle ray tracing and dynamic grid refinement and coarsening. Solid objects, represented by triangulated surfaces, are incorporated using a cut-cell algorithm. A new parallelization scheme suitable for simulating strongly unsteady, non-equilibrium flows is proposed. The parallelization scheme implemented for multi-core Central Processing Units (CPUs) significantly reduces the computational cost of modeling these flows. Performance of the method is assessed by comparing with benchmarked test cases for various rarefied gas flows.

# Contents

<b>Acknowledgements</b>	<b>v</b>
<b>Abstract</b>	<b>vi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Literature Review . . . . .	1
1.2 Motivation and Objectives . . . . .	4
1.3 Thesis Outline . . . . .	7
<b>2 Direct Simulation Monte Carlo</b>	<b>8</b>
2.1 DSMC Method Overview . . . . .	8
2.2 DSMC Methodology . . . . .	11
2.3 DSMC Initialization . . . . .	12
2.4 Particle Movement . . . . .	15
2.5 Boundary Interaction . . . . .	16
2.6 Solid Wall Boundary Conditions . . . . .	16
2.6.1 Velocity Slip and Temperature Jump . . . . .	17
2.7 Periodic Boundary Condition . . . . .	18
2.8 Inflow/Outflow Boundary Conditions . . . . .	18
2.8.1 Pressure Boundary Conditions . . . . .	19
2.8.1.1 Particle Flux Conservation Scheme . . . . .	19
2.8.1.2 Characteristic Theory Based Scheme . . . . .	20
2.8.1.3 Zeroth-Order Extrapolation Scheme . . . . .	21
2.8.1.4 Iteration Time Step Based Correction Scheme . . . . .	22
2.8.2 Methodologies of Particle Generation . . . . .	22
2.8.2.1 Reservoir Method . . . . .	22
2.8.2.2 Standard Method . . . . .	23
2.9 Particle Collisions . . . . .	26
2.9.1 Molecular Models in DSMC . . . . .	26
2.9.1.0.1 The Hard Sphere Model (HS) . . . . .	27
2.9.1.0.2 The Variable Hard Sphere Model (VHS) . . . . .	28
2.9.1.0.3 The Variable Soft Sphere Model (VSS) . . . . .	29
2.9.2 Collision Pairs . . . . .	30

2.9.3	Post Collision Velocities . . . . .	30
2.10	Sampling . . . . .	32
<b>3</b>	<b>New Parallel Adaptive Three-Dimensional DSMC Algorithm</b>	<b>35</b>
3.1	Octree Data Structure . . . . .	35
3.2	Three-Dimensional Hybrid Mesh Scheme . . . . .	36
3.3	Three-Dimensional Particle Ray-Tracing Scheme . . . . .	41
3.4	Spatial and Temporal Adaptivity Scheme . . . . .	42
<b>4</b>	<b>Validation of The Proposed Algorithm Against Benchmark Simulations</b>	<b>54</b>
4.1	Oscillatory Shear-Driven Couette Flow . . . . .	54
4.2	Impulsive Started Couette Flow . . . . .	57
4.3	Thermal Couette Flow . . . . .	59
4.4	Poiseuille Flow . . . . .	59
4.5	Slider Bearing Problem . . . . .	63
4.6	Hypersonic Flows . . . . .	65
4.6.1	Hypersonic Flow Past a Flat-Nosed Cylinder . . . . .	65
4.6.2	Hypersonic Flow Over a Cylinder . . . . .	66
<b>5</b>	<b>Conclusion and Future Work</b>	<b>70</b>



# List of Figures

2.1	Molecular and continuum fluid models. . . . .	9
2.2	The Knudsen number limits on the mathematical models of gas flows [1] . . . . .	10
2.3	DSMC flow chart [2]. $\Delta t$ : Simulation time step; $\Delta t_S$ : Sampling time; $t_L$ : Long time . . . . .	13
2.4	Maxwell velocity distribution as a function of molecular speed. The lines indicate the most probable speed, the mean speed, and the root-mean-square speed. . . . .	15
2.5	Maxwellian Reservoir particle generation method: Ghost cell is indicated in gray ( $L_R$ : reservoir depth). . . . .	24
2.6	Standard particle generation method: The surface which emits particles is represented by the shaded region. . . . .	24
2.7	Illustration of the impact parameters [3] . . . . .	27
2.8	(a) The schematic of the time-averaging of the flow properties over a long interval of simulation time. (b) The schematic of the ensemble-averaging of the flow properties over three independent DSMC simulations, each initiated from a different random number generator seed [4]. . . . .	34
3.1	Octree data structure . . . . .	36
3.2	Example of node neighbors in octree structure (a) a face neighbor, (b) an edge neighbor, and (c) a vertex neighbor. . . . .	37
3.3	Generalized 4x4 transformation matrix in homogeneous coordinates . . . . .	38
3.4	(a) Schematic representation of a triangulated surface mesh of a sphere embedded in a 3D octree Cartesian grid. (b) Bounding box and a cut-cell representation. . . . .	38
3.5	Geometric intersection tests to identify the complex 3D physical object in the octree-based Cartesian structure [5]. . . . .	39
3.6	Ray-Box (a) and Ray-Triangle (b) intersection tests . . . . .	43
3.7	Reflection of a simulated molecule from the boundary triangle element . . . . .	44
3.8	Ray-Tracing scheme . . . . .	45

3.9	The schematic of running transient DSMC simulations on different threads. Top Diagram: Each thread runs sequentially $N$ realizations of the transient simulation consisting of $M$ time steps. $N_s C$ realizations $r_{ijk}, j = 1..N_s, k = 1..C$ , are averaged every output time interval, $T_o = N_o \Delta t$ , to compute the macroscopic properties distributions at output time step $N_o$ . The decision to include a new set of $N_s C$ transient simulations (increment $N$ by $N_s$ ) is based on the relative statistical difference between macroscopic properties of the last $NC$ realizations and the previous $(N - N_s)C$ realizations. Note that $N$ is an integer multiple of $N_s$ . Bottom Diagram: The spatio-temporal adaptivity is carried out for all threads every $N_a = T_a / \Delta t$ time steps. $C$ realizations $r_{ijk}, k = 1..C$ are averaged (by the sniffer) at $t = T_a, 2T_a, \dots, T_f$ (corresponding to time steps $i = N_a, 2N_a, \dots, M$ ) to estimate the macroscopic properties distributions needed for the spatio-temporal adaptivity criteria. These criteria will set the grid size and the associated temporal levels distribution for the time intervals $[T_a + \Delta t, 2T_a], [2T_a + \Delta t, 3T_a], \dots$ . While the sniffer carries out sampling of the microscopic properties and subsequently updates the distribution of cell sizes and temporal levels, the threads pause. Once the sniffer completes its task, the threads resume. . . . .	46
3.10	Simplified flow chart of the implemented DSMC algorithm. $\Delta t_a$ : adaptive time; $\Delta t_s$ : sampling time; $t_L$ : Long time. . . . .	47
3.11	Temporal adaptation algorithm within the DSMC code. $N_l$ : number of temporal levels in the domain; $2^l$ : number of time steps in temporal level $l$ ; $\Delta t_l$ : time step in temporal level $l$ . . . . .	52
3.12	Schematic describing the temporal adaptation procedure with two temporal levels. . . . .	53
4.1	Schematic view of oscillatory Couette flow . . . . .	55
4.2	Normalized velocity profile for the shear-driven oscillatory Couette flow at $\text{Kn} = 0.1$ , $\text{Ma} = 0.3248$ , and $\beta = 5.0$ . . . . .	58
4.3	Normalized wall shear stress for the shear-driven oscillatory Couette flow at different times. $\text{Kn} = 0.1$ , $\beta = 5.0$ , and $\tau_0 = U_0/H$ . . . . .	59
4.4	Velocity (top) and Stress (bottom) fields for the impulsive start Couette flow for $\text{Kn} = 0.21$ at times $16.2\epsilon^{-1}$ . . . . .	60
4.5	Temperature profile in a thermal Couette flow problem . . . . .	61
4.6	Centreline pressure distribution (top figure) and flow field refined mesh for a microchannel in the slip flow regime. . . . .	62
4.7	Schematic of the slider bearing geometry, $L = 5\mu\text{m}$ , $H_o = 5\mu\text{m}$ , $U = 25\text{m/s}$ . . . . .	63
4.8	Slider bearing normalized pressure profile for $\text{Kn}_o = 1.25$ , $\Lambda = 61.6$ , $\text{Ma} = 0.08$ . . . . .	64

4.9	Flow Field Refined Mesh for a Flat-Nosed Cylinder. . . . .	66
4.10	Temperature and density contours for hypersonic flow past a flat-nosed cylinder. A comparison of the results in this work with those computed in previous work done by Bird [3]. . . . .	67
4.11	Temperature and density contours for hypersonic flow past a flat-nosed cylinder. A comparison of the results in this work with those computed in previous work done by Scanlon et al. [6]. . . . .	67
4.12	Sketch of the computational domain of argon hypersonic flow over a cylinder at $Kn_\infty = 0.01$ , $Ma_\infty = 10$ , $T_\infty = 200$ K, $n_\infty = 4.274 \times 10^{20}$ particles/m <sup>3</sup> . . . . .	68
4.13	Contours of temperature of Mach-10 hypersonic flow past a circular cylinder; colored lines are DSMC data; the grey spheres are VMR data [7]. . . . .	69
4.14	Density and Temperature distribution along a vertical line before the cylinder ( $x=0.205$ m). (a) Density (b) Temperature. . . . .	69
4.15	Density and Temperature distribution along a vertical line in the wake region ( $x=0.6$ m). (a) Density (b) Temperature. . . . .	69

# List of Tables

- 1.1 Overview of geometry and general features of the well-known DSMC solvers and the presented one. . . . . 5
- 4.1 Comparison of elapsed time per DSMC realization for non-adaptive and adaptive Poiseuille flow. . . . . 62
- 4.2 Elapsed time(s) of DSMC processes at flow sampling time step for benchmark simulations. . . . . 64
- 4.3 Comparison of elapsed time per DSMC realization for non-adaptive and adaptive hypersonic flow past a flat-nosed cylinder. . . . . 66

# Chapter 1

## Introduction

### 1.1 Literature Review

Micro and nano-technologies are advancing rapidly and the computational tools to predict the flow dynamics efficiently and accurately at these scales are continuously in demand. Gas flows in Micro and Nano devices are typically rarefied and usually fall into the slip and transition flow regimes with a Knudsen number range  $0.001 < \text{Kn} < 10$  [1]. The direct simulation Monte Carlo (DSMC) method [3] is the most widely used computational tool for efficiently simulating fluid flows at these scales. The method has been successfully applied to investigate physical phenomena in a wide range of applications. These include shock waves [3], spacecraft aerodynamics [8], squeeze-films and oscillating microstructures [9], microsensors [10], microfluidics [11], and various rarefied flows in micro/nano-systems [12].

The geometry model in DSMC simulations refers to both the computational mesh of the flow domain and the surface representation of solid objects. Two primary approaches for the geometry model in existing state-of-the-art DSMC solvers have been used. These include body-fitted unstructured grids such as in MONACO [13] and dsmcFOAM [6], and Cartesian structured grids such as in DAC [14], SMILE [15], Bird's DS<sub>n</sub>V [16], MGDS [17], and SPARTA [18]. MONACO code [13] uses an unstructured body-fitted quadrilateral or tetrahedral meshes to fit complex surface geometries. MONACO also employs a localized data structure based on a computational cell to increase the performance of workstation processors that can be used in parallel. Further, it allows an adaptive grid with varying particle weights and variable time steps specified in each cell. dsmcFOAM [6] is the recent DSMC solver implemented within the OpenFOAM software framework, and parallelized with MPI and fully object-oriented C++ based approach. It is able to handle unstructured, arbitrary polyhedral meshes [19]. In contrast to body-fitted grids, the DSMC analysis Code (DAC) [14] developed at NASA's Johnson Space Center employs a two-level Cartesian grid and pos-

esses parallel processing capabilities. DAC offers an automatic discretization of the flow field to ensure appropriate refinement throughout the computational domain. Each level-1 cell is refined into any number (in each coordinate direction) of level-2 Cartesian cells [14]. Different particle weights and variable time steps are allowed for each level-1 computational cells. However, the distribution of the DAC code is restricted to United States users. Bird’s DS<sub>n</sub>V [16] implementations use an equally-spaced background Cartesian grid with several levels of refinements generated within a bounding box of the flow domain. Every partition of the grid is divided into a finer grid of elements such that the number of elements is on the order of the number of simulated molecules. The SMILE code by Ivanov et al. [15] depends on a two-layer rectilinear adaptive grid and a cut-cell method for simulating complex geometries. It is designed for parallel computations on multiprocessor computers. The MGDS code [17] employs an embedded 3-level Cartesian mesh, accompanied by a cut-cell method to incorporate arbitrary triangulated surface geometry into the flow field. It performs fully automated adaptive mesh refinement (AMR) and automatically sets different time steps in each level-3 computational cells. SPARTA [18], developed in C++, employs a parallel processing technique based on the domain decomposition approach. It uses a hierarchical Cartesian grid and a recursive coordinate bisection (RCB) method to decompose the domain among the processors. The code can run on single or multiple processors using the message passaging interface (MPI) library and domain decomposition of the simulation domain. It allows grid refinement in regions with steep gradients of the flow quantities and assigns different particle weights in collision cells proportional to the number of computational particles they possess. Of these codes, DAC [14], SMILE [15], and MGDS [17] are similar in the way of using hierarchical Cartesian grids and employing a ”cut-cell” method that cuts an arbitrary triangulated surface geometries out of the Cartesian flow field grid.

DSMC simulations employ a computational grid needed for collision partner selection and sampling of macroscopic properties. Several problems of practical importance, such as shock waves at micro-scales, boundary layers, pressure sensors, MEMS, etc., have attracted interest to transient phenomena in rarefied gas dynamics. Many of these problems are characterized by large-scale macroscopic gradients and the presence of rarefied and continuum flow domains which dynamically vary with time. Thus, variable resolution of different flow regions is required for an accurate prediction of these flows. In this manner, an adaptive mesh refinement (AMR) has been increasingly used in modern DSMC codes [14, 20, 7]. Recently, a unified flow solver (UFS) for simulating transient rarefied-continuum flows is developed [21]. The solver uses Adaptive Mesh and Algorithm Refinement (AMAR) with dynamically adaptive Cartesian mesh and automatic selection of kinetic solvers (particle-based DSMC or direct Boltzmann solvers using discrete velocity method) based on continuum breakdown criteria [22]. The parallel capabilities of these solvers and an implicit scheme for the kinetic equation are

illustrated in [23]. Hierarchical mesh technology offers unique advantages for automatic mesh for gas flows over complex geometries and moving/deforming structures, dynamic mesh adaptation to local flow properties, and parallelization [24]. Kolobov et al. [22] have developed a UFS that can automatically select the appropriate solver in different parts of the computational domain, where the direct numerical solution of the Boltzmann transport equation is used in rarefied regions, and continuum flow dynamics solvers (Euler, Navier-Stokes) are used elsewhere. Two and three-level Cartesian adaptive DSMC meshes are utilized in a number of DSMC codes [25, 16, 26]. However, very little investigations have been devoted to the potential advantages of an octree data structure in DSMC simulations. Olson and Christlieb [27] used the gridless character provided by an octree-based algorithm to develop a gridless approach for sorting of nearest-neighbor gas particles into local clusters. Octree data structure provides a potentially more general strategy for different cell volumes over multiple orders of magnitudes without the restrictions associated with a fixed number of refinement levels. In this manner, Arslanbekov et al. [28] used this approach and developed a UFS-DSMC code that consists of the two types of kinetic solvers coupled via (AMAR) methodology and can efficiently perform some of the same functions as transient sub-cells for collision partner selection, and makes unified parallelization of the code.

DSMC solvers can run with a single processor or in parallel using the Message Passing Interface (MPI) library on multiple processors. Parallelization is carried out by decomposition of the spatial domain among the processors. Bird [3] points out that dynamic grid adaptation is a main concern when applying the DSMC method to multi-dimensional problems and states that an ideal DSMC grid must fulfill three requirements: high computational efficiency, grid adaptation to arbitrary geometries as well as to local flow conditions. Besides parallel computing implementation, fully automated mesh adaptation of the flow field can effectively save computational cost and provides efficient management of grid resolution. While time step adaptation depends on the local mean collision time, the adaptation of collision grid cells is based on the local mean-free-path, the number of simulated molecules, and other aspects such as the presence of surface meshes. The use of a single fixed time step is computationally inefficient when localized high-density gradients regions exist in the computational domain. In such flows, different time steps are needed in different parts of the domain. With variable time steps, a single iteration of the DSMC algorithm no longer represents the same amount of physical time in each collision cell [29]. One way to tackle this challenge is proposed by Kannenberg and Boyd [29], where the disparity in the elapsed time is accounted for by weighting all particles by a time scale factor, defined as the ratio of the local time step to the reference time step for the simulation. The use of varying time steps according to this scheme increases computational efficiency by reducing the number of simulated molecules in the computational domain while maintaining relatively uniform molecule distribution

and sufficient molecules for obtaining accurate collision statistics per cell. Implementations, such as DAC [14] and MGDS [17], that use variable scaling of particle weighting with spatially dependent time-steps are common. In the DAC algorithm, however, the particle weight and time step size vary independently. This requires cloning or deleting molecules to guarantee a balance of flux when the molecule crosses one collision cell to the next. A different approach for handling variable time steps is presented as a recent improvement to the DSMC algorithm in [30], and has implemented in a recent work done by Wade et al. [31]. This approach is based on updating a desired local time step (DTS) for each collision cell, which is set to the minimum of user-specified fractions of the relative collision and transit times of the cell. A time parameter is assigned to all molecules and all collision cells in which each molecule in a particular cell inherits the time step of that cell. The flow time is advanced in steps equal to the smallest value of DTS over all cells in the computational domain. The cell and molecule time parameters are advanced based on the flow time and the DTS values within the cells. Other variable time step methods [32, 33] propose a local time stepping scheme which specifies a movement time step and a collision time step for each representative particle in the computational domain. The two time steps are independent and respectively represent the motion and collision evolution of the particle system. In addition, only when both the time attributes evolve to a certain time, is the gas flow treated to develop to the same time [33]. Table 1.1 gives an overview of the geometry and general features of existing state-of-the-art DSMC solvers.

## 1.2 Motivation and Objectives

The scope of this thesis is to develop a novel three-dimensional multi-scale direct simulation Monte Carlo algorithm for the simulation of unsteady rarefied gas flows over complex geometries. We aim to develop an ideal DSMC algorithm with high computational efficiency, efficient data management, low memory storage requirement, increased flexibility, and dynamic grid adaptation to arbitrary geometries as well as to local flow conditions. The flow domain is represented by a hybrid mesh consisting of a hierarchical octree-based Cartesian grid [34, 28], whereas the surfaces of solid objects are represented by a triangular mesh. The hierarchical octree-based Cartesian grid representation of the domain allows for an efficient data storage and management that is compatible with the spatio-temporal adaptation scheme. When compared to unstructured meshes, such representation significantly improves memory requirement and is, therefore, more suitable for simulating large-scale DSMC problems. The hierarchical octree-based Cartesian grid representation also enables a potentially more general scheme for varying cell volumes over a large range of the molecular length scales. The hybrid mesh representation allows for simple integration of a variety of effective geometric



Table 1.1: Overview of geometry and general features of the well-known DSMC solvers and the presented one.

DSMC Algorithm	Grid		Parallel Processor Distribution	Spatial Adaptivity Criteria		Temporal Adaptivity Criteria		Other
	Structured	Unstructured		Grid Adaptation	Grid Adaptation	Time Step	Time Step	
MONACO	Unstructured grid	Unstructured grid	Over grid	Based on $\lambda$	Based on $\lambda$	Single fixed time step	Single fixed time step	Localized data structure Ray tracing through an unstructured grid
dsmcFOAM	Unstructured grid	Unstructured grid	Over grid	Based on $\lambda$	Based on $\lambda$	Single fixed time step	Single fixed time step	openFOAM C++ toolbox Cut-cell method
DAC	Two-level Cartesian grid	Two-level Cartesian grid	<b>X</b>	Based on $\lambda$	Based on $\lambda$	Variable time step Variable particle weight	Variable time step Variable particle weight	Restricted distribution to US users Parallel processing capabilities
DSuV	Cartesian grid	Cartesian grid	<b>X</b>	Based on $N_{sim}$	Based on $N_{sim}$	Variable time step	Variable time step	<b>X</b>
SMILE	Cartesian grid	Cartesian grid	<b>X</b>	Based on $\lambda$	Based on $\lambda$	<b>X</b>	Variable time step Variable particle weight	Cut-cell method Parallel processing capabilities
MGDS	Three-level Cartesian grid	Three-level Cartesian grid	Over grid	Based on $\lambda$	Based on $\lambda$	Variable time step Variable particle weight	Variable time step Variable particle weight	Cut-cell method Ray tracing through a Cartesian grid
SPARTA	Hierarchical Cartesian grid	Hierarchical Cartesian grid	Over grid	Based on $\lambda$	Based on $\lambda$	Single fixed time step	Single fixed time step	Highly probable C++ Run one/multiple simulations simultaneously in parallel
Code Presented	Hierarchical Cartesian grid	Hierarchical Cartesian grid	Over realization	Based on $\lambda$ & $N_{sim}$	Based on $\lambda$ & $N_{sim}$	Variable time step Different Temporal levels	Variable time step Different Temporal levels	Octree-based Cartesian grid Geometric tools in computer graphics Ray tracing through a structured grid Spatio-temporal adaptivity scheme

tools used in computer graphics, including fast particle-tracing algorithms. This enables DSMC calculations to be performed with less number of operations, such as in successive grid adaptation, particle movement, and particle sorting. A cut-cell method to simulate flows around immersed objects of complex boundaries is implemented [35]. For near-boundary computational cells that are cut by the true physical boundary of the solid object, the method computes the effective volume of cut-cells for accurate prediction of molecular collisions and macroscopic properties in these cells. The cut-cell method also allows for decoupling of the flow field mesh from the solid boundaries surface mesh, making it suitable for simulating near-continuum flows with large density variations.

Novel aspects of the proposed algorithm include parallelization over realizations and a new spatio-temporal adaptation scheme. Most existing DSMC solvers involve parallelization on multiple processors using the Message Passinging Interface (MPI) library. The spatial domain decomposition parallelization method is often used. However, This flavor of parallelization suffers from two drawbacks. The first is the challenge of load balancing and distributed storage of the computational domain. This challenge is particularly severe when the solver employs a spatio-temporally adaptive algorithm that dynamically adjusts the computational grid and the time step in response to the evolving flow field structures in unsteady flows. The second challenge is that the criteria for spatio-temporal adaptivity are based on macroscopic properties computed as averages over a statistically meaningful number of realizations (a realization is a DSMC simulation initiated from a unique random number generator seed). This latter observation suggests parallelization over independent realizations. In the proposed framework, the DSMC algorithm is optimized for simulating unsteady flows in parallel over multiple cores. In contrast with distributing the computational domain over the cores (or threads, or CPUs), the independent realizations are distributed over the cores. Each thread or core processes a realization of the simulation of the unsteady flow over the entire computational domain. Due to the lack of communication between the cores when each is handling an independent realization, the parallelization efficiency is almost 100 %. In addition, this type of parallelization is optimal when simulating strongly unsteady, non-equilibrium flows, where dynamic spatial-temporal adaptation is required. With each core being assigned a DSMC realization, local macroscopic flow properties at different time steps are collected simultaneously in parallel and computed as statistical averages over a number of realizations for local spatial-temporal adaptation.

The proposed algorithm implements a novel spatio-temporal adaptivity scheme efficient for simulating highly unsteady rarefied flows over complex geometries. These flows typically experience considerable variability in the spatial gradients of the macroscopic thermodynamics properties, and as such, spatial adaptation needs to be frequently carried out. Moreover, the localized high-density gradients regions that exist in these flows require the use of different time steps in different parts of the domain. This calls for an automated adaptive time stepping scheme.

Instead of advancing with most limiting (smallest) time step, the scheme handles spatial dependence of the time step by employing a number of discrete temporal levels. The method implements a smart algorithm that efficiently loops over these levels, in a descending order of the time step size, where within each loop, all cells sharing the same time step are handled. Table 1.1 presents the main features of the proposed DSMC solver in comparison with existing DSMC solvers.

### 1.3 Thesis Outline

The thesis is organized as follows: Chapter 2 gives a description of the specifics of direct simulation Monte Carlo method. It highlights the implementation of different boundary conditions used in the numerical simulation of rarefied flows by the DSMC Method. Chapter 3 presents the proposed three-dimensional unsteady parallel DSMC algorithm. It provides the details of the implementation of the three-dimensional hybrid mesh scheme, the effective three-dimensional particle ray-tracing scheme, and the spatio-temporal adaptivity scheme. In Chapter 4, extensive validation of our algorithm through several benchmark 2D and 3D numerical simulations is performed. A wide variety of problems will be investigated, including microfluidics, thermal processes, low speed subsonic flows, and hypersonic flows. It will be shown that the developed algorithm can produce reliable results. Chapter 5 presents a summary of conclusions and recommends future work that can be done to further advance the proposed algorithm to predict unsteady flows in complex geometries encountered in micro- and nano-electromechanical systems (MEMS/NEMS).

# Chapter 2

## Direct Simulation Monte Carlo

### 2.1 DSMC Method Overview

There are two existing models to study gas flow behavior at the macroscopic and the microscopic level: Continuum-based models and molecular models. At the macroscopic level, continuum-based models treat the fluid as a continuum and define fluid density, flow velocity, pressure, temperature, and other macroscopic flow quantities at every point in space and time. Conservation laws such as conservation of mass, momentum, and energy are a set non-linear partial differential equations (Euler, Navier-Stokes, Burnett, etc.). At the microscopic level, molecular models are divided into deterministic, such as the best-known molecular dynamics (MD) approach, and statistical, such as the direct simulation Monte Carlo (DSMC) method. These particle methods model gas behavior by tracking the interaction of computation particles, each with a position, velocity, and internal energy, etc., and mimicking the discrete molecular nature of the actual flow. They are intuitively attractive because they lack continuum assumptions and therefore be valid where traditional, continuum-based, computational fluid dynamics (CFD) techniques break down due to flow rarefaction. Fluid modeling classification is depicted schematically in Figure 2.1.

Among the particle simulation methods, DSMC, pioneered by G. A. Bird in the 1960s [3], is the most powerful numerical technique for the simulation of complex, non-equilibrium rarefied gas flows in kinetic theory. The method can be viewed as either a simplified molecular dynamics (DSMC being several orders of magnitude faster) or as a stochastic particle method for solving the time-dependent non-linear Boltzmann equation. For physicists and chemists, the most well-known particle method is molecular dynamic (MD). In traditional molecular dynamics, the particle interaction routines have been an order of  $O(N^2)$  since each simulated molecule is allowed to interact with every other molecule in the physical domain. With the introduction of interaction potential functions, the computational time has improved, but the problem is still computationally expensive and

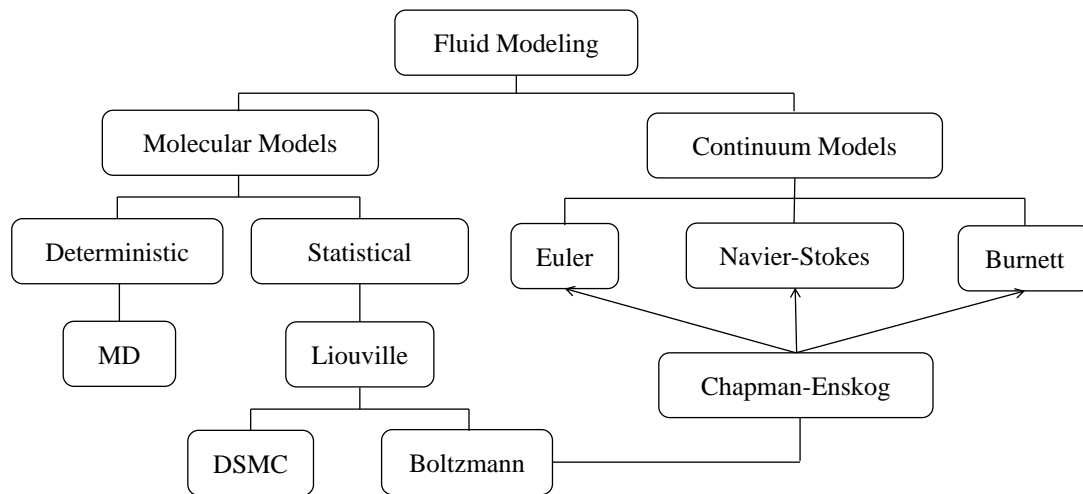


Figure 2.1: Molecular and continuum fluid models.

extremely time-consuming because of the coupling between molecular motion and collisions. Consequently, the algorithm sped up considerably with the dilute gas approximation which allows the separate computation of the collision dynamics of molecules from their motion. To take the full advantage of this simplification, Bird in 1963 has proposed an efficient means of sorting and collision sampling routines [36] which allow candidate collision pairs to be in the order  $O(N)$  CPU time. The result was the direct simulation Monte Carlo (DSMC) algorithm that was two or three orders of magnitude faster than molecular dynamic (MD).

The governing dimensionless parameter for prediction of continuum is Knudsen number ( $Kn$ ), which is defined as the ratio between the molecular mean free path  $\lambda$  and the characteristic length scale of the physical system  $L$ . It indicates the degree of flow rarefaction and varies from zero when the gas is considered as continuum to infinity when the intermolecular collisions can be discounted. In general, a continuum description is valid when  $\lambda$  is much smaller than the characteristic flow dimension  $L$ , i.e.  $Kn < 0.1$ . Rarefied gas flows are in general encountered in low-pressure applications such as high altitude atmospheric flows, and in small geometries such as in micro-electromechanical systems (MEMS). According to the magnitude of the Knudsen number, gas flows can be empirically classified into four regimes [1] as depicted in Figure 2.2 and can be summarized as follows: When the Knudsen number is small ( $Kn$  tends to zero), non-equilibrium effects are insignificant, and the standard Navier-Stokes-Fourier (NSF) equations then reduce to the inviscid Euler equations. For  $Kn \leq 0.001$ , the NSF equations with no-slip boundary conditions are valid. As  $Kn$  increases ( $0.001 \leq Kn \leq 0.1$ ), rarefaction effects become more important, and the NSF equations with velocity slip and temperature jump boundary conditions can accurately predict the gas behavior. However, once the Knudsen number increases

to transition-continuum ( $0.1 \leq \text{Kn} \leq 10$ ) and free-molecular ( $\text{Kn} > 10$ ) flow regimes, the continuum and thermodynamic equilibrium hypotheses breakdown; therefore, the Boltzmann equation (BE) must be considered to analyse such flows. The (BE) and the direct simulation Monte Carlo (DSMC) has proven to be the most reliable in describing rarefied gas flows in the slip flow and transition flow regimes.

Euler :  $\text{Kn} \rightarrow 0 (\text{Re} \rightarrow \infty)$   
 Navier - Stokes without slip boundary :  $\text{Kn} < 10^{-3}$   
 Navier - Stokes with slip boundary :  $10^{-3} \leq \text{Kn} < 10^{-1}$   
 Continuum - transition regime :  $10^{-1} \leq \text{Kn} < 10$  (small  $\text{Re}$ )  
 Free - molecule regime :  $\text{Kn} \geq 10 (\text{Re} \rightarrow 0)$

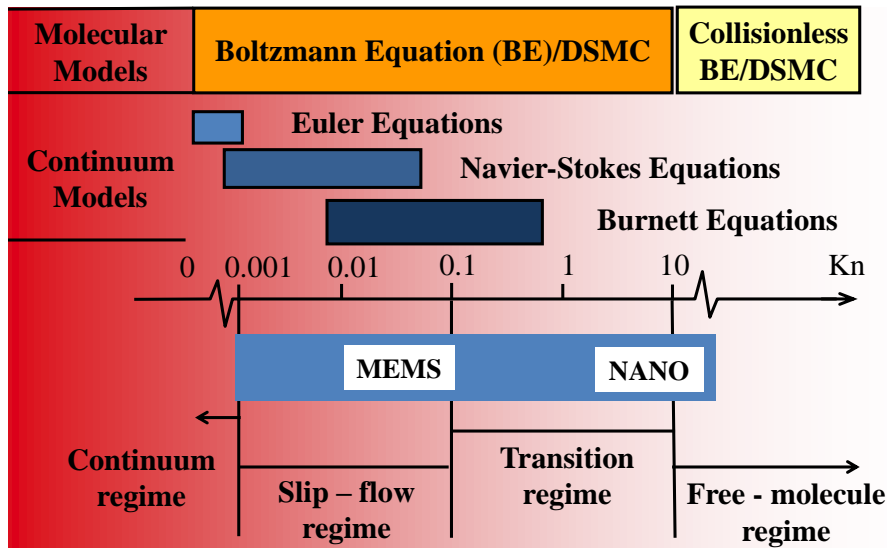


Figure 2.2: The Knudsen number limits on the mathematical models of gas flows [1]

Knudsen number can be related to other two important dimensionless parameters in macroscopic fluid mechanics, the Reynolds number defined as the ratio of inertia to viscous forces  $\text{Re} = \rho v_0 L / \mu$ , where  $\rho$  is the fluid density,  $v_0$  is the characteristic velocity,  $L$  the characteristic flow length-scale, and  $\mu$  the dynamic viscosity of the fluid, and the well-known Mach number  $\text{Ma} = v_0 / a_0$  defined as the ratio of the characteristic gas flow speed ( $v_0$ ) to the speed of sound ( $a_0$ ).  $\text{Ma}$  is a measure of fluid compressibility and considered as the ratio between inertia and elastic fluid forces. From the kinetic theory of gases, the mean free path  $\lambda$  for gases is well defined as the average distance travelled by molecules between collisions. For an ideal gas modeled as rigid spheres, the mean free path is given

by (Bird 1994 [3]):

$$\lambda = \frac{1}{\sqrt{2}\pi n\sigma^2} \quad (2.1)$$

where  $n$  is the number density (number of molecules per unit volume), and  $\sigma$  is the effective molecular diameter. According to the kinetic theory of viscosity, thermal conduction and diffusion in gases [37], the mean free path is related to the viscosity  $\mu$ , density  $\rho$ , and the mean molecular speed  $\bar{c}$  as:

$$\mu \approx \frac{1}{2}\rho\bar{c}\lambda \quad (2.2)$$

where  $\bar{c} = \sqrt{8kT/\pi m}$ ,  $m$  is the molecular mass,  $T$  is the absolute temperature, and  $k$  is the Boltzmann constant  $k = 1.38 \times 10^{-23} J/K$ . The mean molecular speed  $\bar{c}$  is related to the sonic speed  $a_0 = \sqrt{\gamma kT/m}$ , where  $\gamma$  is the specific heat ratio, according to the equation:

$$a_0 = \sqrt{\frac{\pi\gamma}{8}}\bar{c} \quad (2.3)$$

Combining the above two equations yields to  $\lambda = 1.26\sqrt{\gamma}\mu/a_0\rho$ . It is easy to obtain the required fundamental relationship:

$$\text{Kn} = \frac{\lambda}{L} = 1.26\sqrt{\gamma}\text{Ma}/\text{Re} \quad (2.4)$$

A brief summary of the DSMC method, highlighting key aspects relevant to the algorithm we are proposing in this work, is presented in the next sections. The method is described in detail in the monograph by Bird [3] and the recent edition *Molecular Gas Dynamics and the Direct Simulation of Gas Flows* (Bird 1994 [3]).

## 2.2 DSMC Methodology

Based on the well-developed kinetic theory of gases, the DSMC method emulates the physics of a real gas and provides a solution to the non-linear Boltzmann equation. It follows a representative set of randomly selected simulated molecules, each representing a large number of physical molecules, as they collide and move in physical space. The molecules' motion, their interactions with boundaries and intermolecular collisions alter with time their spatial coordinates, velocity components, and their internal energies. Molecular motions are modeled on a deterministic basis, while their collisions are treated on a probabilistic basis according to an appropriate collision model. The simulation of the real gas flow is carried out by statistical sampling of the macroscopic flow properties within grids in the physical space of the flow field. Figure 2.3 illustrates a simplified flow chart of the four major processes involved in a DSMC algorithm for an unsteady

or steady flow problem [2]: 1) moving the molecules over a time step and model their interactions with boundaries, 2) indexing and tracking the molecules within the grid of collision cells, 3) selecting molecules for collision on a probabilistic basis and applying the appropriate collision model, 4) sampling the macroscopic flow properties.

The validity of the DSMC method is associated with space and time discretization. The collision grid cell size,  $\Delta x$ , must be small compared to the local mean free path,  $\lambda \sim |\pi/\nabla\pi|$ ; the length scale characterizing the spatial variations of the macroscopic properties,  $\pi$ . So, we choose  $\Delta x \ll \lambda$ , where  $\lambda$  generally varies with space and time. The simulation time step,  $\Delta t$ , over which molecular motions and collisions are uncoupled must be smaller than the local mean collision time,  $\Delta t < \tau_c = \lambda/v_{\text{mp}}$ , where  $v_{\text{mp}}$  is the most probable velocity. It must be smaller than the molecule time-of-flight across a typical collision cell size, where it is preferable for a molecule to spend two to three-time steps before leaving a cell. The time step must also be smaller than the viscous diffusion time,  $\Delta t < L^2/\nu$ , and the period of oscillations for unsteady periodic flows,  $\Delta t < 2\pi/\omega$ , where  $\omega$  is the frequency of oscillations. The number of simulated gas molecules per cubic mean free path,  $N$ , must be larger than a minimum (typically 20 molecules) in order to preserve collision statistics and the molecules yield a reasonable approximation to the local velocity distribution function. Unless the gas is highly rarefied and the simulation domain is small, the constraints on  $\Delta x$ ,  $\Delta t$ , and  $N$  make DSMC computationally expensive. Thus, adaptive techniques are required in hypersonic near-continuum flows spanning several orders of magnitude of length and time scales.

## 2.3 DSMC Initialization

Initially, the number of simulated molecules in the computational domain is randomly distributed with uniform density throughout the system. Once the positions of molecules are determined, each molecule is initialized with a velocity according to the Maxwellian velocity distribution. Recall that the direct simulation Monte Carlo method is essentially equivalent to the Boltzmann equation and that the velocity distribution function is the only dependent variable in the Boltzmann equation. The Maxwell-Boltzmann distribution function  $f(\mathbf{U})$  that describes the distribution of molecular velocities in an ideal gas, established first by James Clerk Maxwell and later proved by Ludwig Boltzmann [38, 39, 40, 41], is given as:

$$f(\mathbf{U}) = \left(\frac{\beta}{\sqrt{\pi}}\right)^3 e^{(-\beta^2\|\mathbf{U}\|^2)} \quad (2.5)$$

where  $f(\mathbf{U})d\mathbf{U}$  defines the probability of a given molecule selected at random and has a velocity  $\mathbf{U}$  in the range  $[\mathbf{U}, \mathbf{U} + d\mathbf{U}]$  at any instant, and  $\beta$  is the inverse of



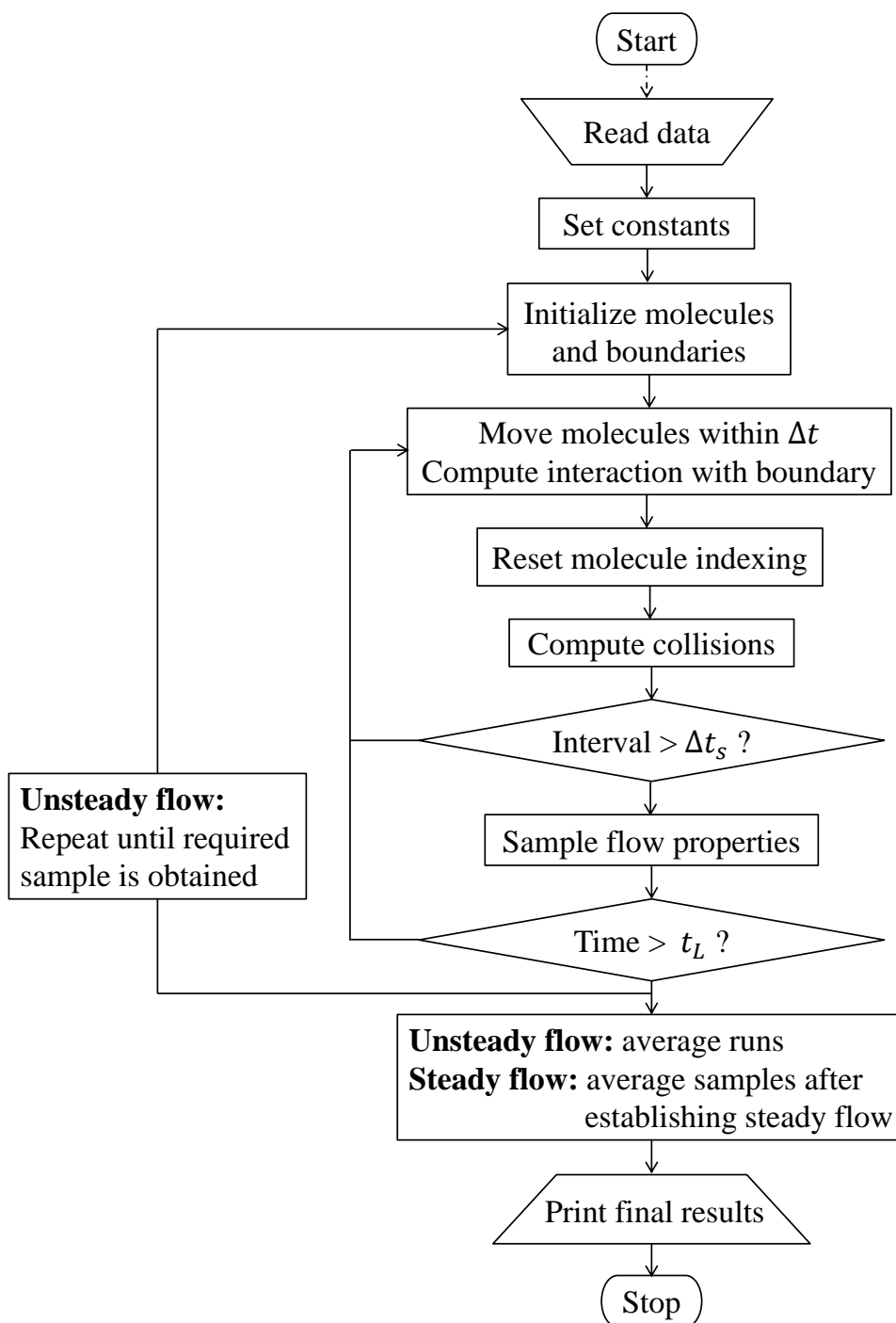


Figure 2.3: DSMC flow chart [2].  $\Delta t$ : Simulation time step;  $\Delta t_s$ : Sampling time;  $t_L$ : Long time

the most probable molecular thermal speed, given by:

$$\beta = \frac{1}{\sqrt{2kT/m}} \quad (2.6)$$

where  $k$  being the Boltzmann constant,  $T$  is the absolute temperature of the gas, and  $m$  is the molecular mass. The equation  $f(\mathbf{U})$  above can be sampled as a product of three Gaussian distributions as follows:

$$f(u, v, w) = f(u)f(v)f(w) = \frac{\beta}{\sqrt{\pi}}e^{-\beta^2u^2} \cdot \frac{\beta}{\sqrt{\pi}}e^{-\beta^2v^2} \cdot \frac{\beta}{\sqrt{\pi}}e^{-\beta^2w^2} \quad (2.7)$$

Accordingly, the Box-Muller algorithm [42] in polar coordinates  $(r, \theta, \phi)$ , where  $x = r\sin\phi$ ,  $y = r\sin\theta$ , and  $z = r\cos\theta$ , is applied along with the principle of inversion to numerically find the molecular velocities in each of the three dimensions as follows:

$$u = r\sin(\theta) \quad (2.8)$$

where  $r = \sqrt{\frac{-2kT}{m}\log(R_{f_1})}$  and  $\theta = 2\pi R_{f_2}$

$$\begin{aligned} v &= r\sin(\theta) \\ w &= r\cos(\theta) \end{aligned} \quad (2.9)$$

where  $r = \sqrt{\frac{-2kT}{m}\log(R_{f_3})}$  and  $\theta = 2\pi R_{f_4}$ . Note that  $R_{f_i}$  ( $i = 1, 2, 3, 4$ ) is a random number generated by the compiler between 0 and 1 ( $0 \leq R_{f_i} < 1$ ).

In the context of the Kinetic Molecular Theory of Gases, a gas contains a large number of particles in rapid motions. Each particle has a different speed, and each collision between particles changes the speeds of the particles. An understanding of the properties of the gas requires an understanding of the distribution of molecular gas speeds. Many useful properties of the gas in equilibrium state can be obtained from the Maxwellian distribution. The Maxwell-Boltzmann distribution can be used to determine the distribution of the kinetic energy for a set of molecules which is identical to the distribution of speeds for gas at a certain temperature.

Various 'average' molecular speeds, irrespective of direction, may be obtained easily from the distribution function as follows: The most probable speed corresponds to the maximum value of the Maxwell speed distribution function. It is given as:

$$\hat{u} = \|\mathbf{U}\|_{\left(\frac{df}{du}=0\right)} = \sqrt{\frac{2kT}{m}} \quad (2.10)$$

The mean speed is:

$$\langle u \rangle = \int_0^\infty f(\|\mathbf{U}\|) \|\mathbf{U}\| d\|\mathbf{U}\| = \sqrt{\frac{8kT}{m}} \quad (2.11)$$

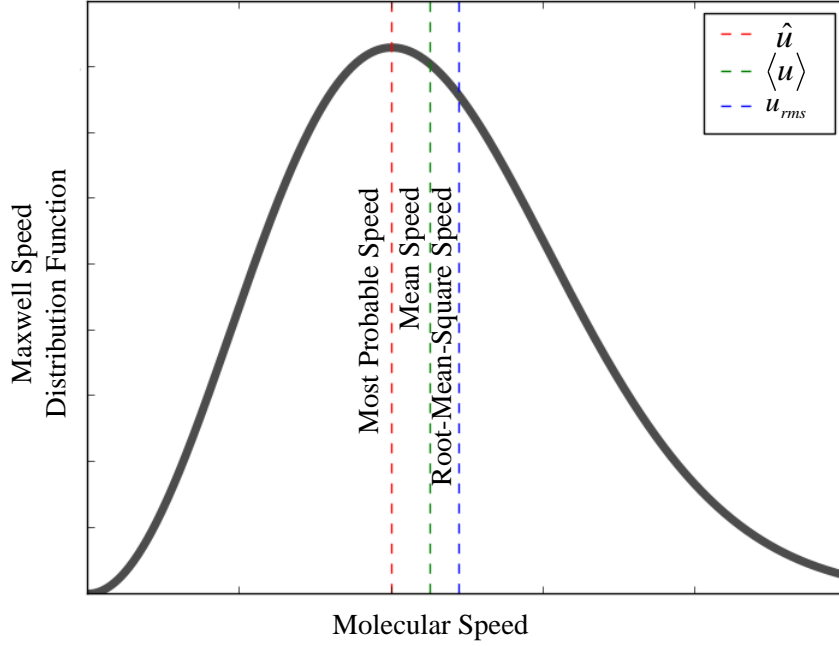


Figure 2.4: Maxwell velocity distribution as a function of molecular speed. The lines indicate the most probable speed, the mean speed, and the root-mean-square speed.

The root-mean-square speed is given by:

$$\langle u^2 \rangle^{1/2} = u_{rms} = \sqrt{\int_0^\infty f(\|\mathbf{U}\|) \|\mathbf{U}\|^2 d\|\mathbf{U}\|} = \sqrt{\frac{3kT}{m}} \quad (2.12)$$

## 2.4 Particle Movement

After initializing the state of the system by the positions and velocities of simulated molecules,  $\{\vec{r}_i, \vec{v}_i\}$ , the DSMC time marching starts and the simulation time step is defined. Simulated molecules are then moved as if they do not interact, that is, their positions are updated to  $\vec{r}_i + \vec{v}_i \Delta t$  according to their microscopic velocity components in a straight line path. Appropriate boundary conditions are applied whenever a molecule strikes a boundary. When a molecule reaches a boundary, the time-step is adjusted to  $(\Delta t - \Delta t_w)$ , where  $\Delta t_w$  is the time of flight needed by the particle to cover a straight-line trajectory from the molecule's initial position  $\vec{r}_i$  to the point of impact  $\vec{r}_w$ .  $\Delta t_w$  is given by  $\Delta t_w = (\vec{r}_w - \vec{r}_i) \cdot \vec{n} / (\vec{v}_i \cdot \vec{n})$ , where  $\vec{n}$  is the unit normal to the surface. Then, the molecule moves with a new reflected velocity as dictated by the boundary conditions using the new time-step.

## 2.5 Boundary Interaction

The interaction of simulated molecules with physical boundaries is an important boundary condition in DSMC simulations. During their motion, simulated molecules may encounter boundaries either at the periphery of the computational domain or at surfaces of immersed bodies within the domain. The proper numerical treatment and implementation of the boundary condition when a molecule crosses a boundary is of great importance for accurate DSMC simulation. Different types of boundary conditions are employed in DSMC simulations. These include solid wall boundary conditions, periodic boundaries, and infow/outflow boundary conditions.

## 2.6 Solid Wall Boundary Conditions

A given gas-solid surface interaction can be treated as being fully specular, fully diffuse, or by some combination of the two. This description is identified by the tangential momentum accommodation coefficient that ranges from 0, for no accommodation (specular reflection), to 1, for full accommodation (diffuse reflection).

In specular reflection, the molecular velocity component normal to the incident surface is simply reversed and the tangential component is left unchanged. The specular wall boundary condition is the computational representation of an inviscid wall and may also be used to represent a symmetry plane. The shear stress at the wall and the rate of heat transfer in such a case are zero.

However, diffuse reflection from a perfect thermal wall, at temperature  $T_w$ , cause random reorientation of all the three components of the outgoing velocity of the reflected molecule, where the normal component is reset according to the biased-Maxwellian distribution:

$$P_{\perp}(v_{\perp}) = \frac{m}{kT_w} v_{\perp} e^{-mv_{\perp}^2/2kT_w} \quad (2.13)$$

and the parallel components are reset according the standard Maxwell-Boltzmann distribution:

$$P_{\parallel}(v_{\parallel}) = \sqrt{\frac{m}{2\pi kT_w}} e^{-mv_{\parallel}^2/2kT_w} \quad (2.14)$$

where  $k$  is the Boltzmann constant. The shear stress at the wall and the rate of heat transfer are correspondingly non-zero. For a more detailed explanation and a derivation of the diffuse wall boundary condition equations, refer to (Bird 1994 [3]).

In addition to the specified wall temperature boundary condition, specified wall heat flux boundary condition is extensively encountered in the DSMC method for the rarefied gas simulations of Micro/Nano electro-mechanical systems (MEMS/NEMS)

with a desired heat energy exchange. Recently, Akhlaghi and Roohi [43] and Roohi et al. [44] implemented and validated a novel algorithm to impose a desired (heating/cooling) wall heat flux in DSMC. This technique is based on an iterative modification of the local wall temperature in order to achieve the desired wall heat flux.

### 2.6.1 Velocity Slip and Temperature Jump

At sufficiently low Knudsen number ( $\text{Kn} < 10^{-3}$ ), both heat conduction and viscous diffusion and dissipation are negligible, and the assumption of a continuum with no-slip boundary conditions hold. Then, the velocity and temperature of the gas near the wall are in equilibrium and equal to the wall velocity and temperature. As Kn number increases, the flow becomes more rarefied and non-equilibrium effects arise near the wall due to the insufficient number of molecular collisions under rarefied conditions, thus invalidating the no-slip boundary conditions. This results in conditions with velocity slip and temperature gradients near the wall, known as slip conditions. The slip boundary condition shows at first a linear relationship with the Knudsen number ( $\text{Kn} < 0.1$ ); then non-linear effects take over in the transition ( $0.1 < \text{Kn} < 10$ ) and free molecular ( $\text{Kn} > 10$ ) regimes. The first-order Maxwell slip velocity and Maxwell-Smoluchovski temperature jump boundary conditions are given as [45, 46]:

$$U_s - U_w = C_m \lambda \frac{\partial U}{\partial n} + C_s \frac{\mu}{\rho T} \frac{\partial T}{\partial x} \quad (2.15)$$

where  $U_w$  is the velocity at the wall,  $U_s$  is the velocity slip on the solid surface,  $n$  is normal to the wall,  $x$  is parallel to the wall,  $C_m$  is the velocity slip coefficient,  $\lambda$  is the mean free path,  $C_s$  is the thermal creep coefficient,  $\mu$  is the viscosity,  $\rho$  is the mass density, and  $T$  is the temperature.

$$T_s - T_w = C_t \lambda \frac{\partial T}{\partial n} \Big|_{n=0} \quad (2.16)$$

where  $T_w$  is the gas temperature at the wall,  $T_s$  is the surface wall temperature,  $C_t$  is the temperature jump coefficient.

The Boltzmann equation inherently takes into account the slip-boundary conditions; therefore, the DSMC method takes account of them as well. The slip velocity and temperature jump are calculated based on the direct microscopic sampling of the corresponding molecules macroscopic properties that strike the wall boundary. These relations, implemented in Bird's DS2V algorithm [16, 47], are given as:

$$U_s = \frac{\sum ((m/|U_n|) U_p)}{\sum (m/|U_n|)} \quad (2.17)$$

$$T_{tra,j} = \frac{1}{3R} \frac{\sum ((m/|U_n|) (\|\mathbf{U}\|^2)) - \sum (m/|U_n|) U_s^2}{\sum (1/|U_n|)} - T_{tra,w} \quad (2.18)$$

where  $U_s$  is the velocity slip,  $U_n$  is the velocity normal to the wall,  $U_p$  is the velocity parallel to the wall,  $\|\mathbf{U}\|$  is the velocity magnitude,  $T_{tra,j}$  is the translational temperature jump, and  $T_{tra,w}$  is the wall translational temperature and the summations are taken over all particles that strike the surface.

## 2.7 Periodic Boundary Condition

The periodic boundary condition is the computational representation of an infinitely large periodic domain. It is used when the physical geometry of interest and the flow field have a periodically repeating nature. Thus, two opposite periodic planes are always used to specify that whatever enters through one periodic plane must simultaneously exit the opposite periodic plane. This ensures that every simulated molecule crossing a boundary re-enters at the periodic mirror boundary to conserve the number of simulated molecules as if the simulation boundary is infinitely replicated in the direction of the molecule's motion. There is no change in the molecule velocity; therefore, all macroscopic flow properties remain unchanged.

## 2.8 Inflow/Outflow Boundary Conditions

The inflow/outflow boundary conditions can accommodate supersonic and subsonic flows. For supersonic flows, free stream boundary condition of Dirichlet type is employed at the inlet with known free stream properties. Thus, particles are injected based on specified number density, temperature, and mean flow velocity of the external flow. However, vacuum boundary condition is applied at the outlet where no molecules are allowed to enter the computational domain and all molecules striking the "vacuum" boundary are removed from the flow. In DSMC simulations of external hypersonic flow fields, the stream velocities are much higher than molecule velocities and very few molecules enter the domain from the outflow boundaries; hence, vacuum boundary conditions can accurately predict flow properties at the outlet boundary. However, this is not the case in microflow applications where the flows in microsystems are often subsonic, the flow conditions are not uniform, and free stream boundary conditions are not physically correct. In addition, the flow velocities are much lower than thermal velocities, and hence, molecules can enter the computational domain from both boundaries. Typically, accurate measurements of velocity in microscale experiments are rather difficult whereas pressure and temperature are more easily measured. Therefore, pressure boundary conditions are employed such that with the specified pressure, other flow variables such as the velocity are evaluated in an implicit manner from the cells inside the domain once the inflow/outflow boundary conditions propagate into the domain as the flow simulation progresses.

Several treatments have been proposed in the literature to model pressure boundary conditions and are summarized below.

## 2.8.1 Pressure Boundary Conditions

Pressure boundary conditions are widely employed to predict gaseous flows in micro-channels. Generally, two types of boundary conditions are used: (i) specified inlet and exit pressure and (ii) specified mass flow rate at the inlet (inlet pressure and temperature) and specified pressure at the outlet. Various schemes have been proposed in the literature to model inlet/outlet pressure boundary conditions such as particle flux conservation scheme [48, 49], schemes based on the theory of characteristics [50, 51, 52, 53], schemes based on extrapolations from the interior domain [52, 54].

### 2.8.1.1 Particle Flux Conservation Scheme

Ikegawa and Kobayashi [48] proposed this scheme based on the particle-conservation concept and determined the particle fluxes at the inlet and outlet boundaries by computing the number of particles crossing the computational boundary. Accordingly, the mean flow velocity,  $u(t)$ , at the boundary is computed using the number of particles flowing in/out the boundary within a time step. This is given as:

$$u(t) = \frac{N_+^{t-1} - N_-^{t-1}}{n\Delta t A / F_N}$$

where the subscripts + and – refer to the particles flowing in and out of the computational boundary,  $n$  is the number density,  $t$  is the current time step,  $A$  is the boundary cross-sectional area, and  $F_N$  is the number of real particles represented by each simulated particle.

For the outlet boundary, the temperature is not known in advance, and the boundary may not be at the same temperature. Hence, the temperature is computed in an implicit manner from the cells inside the domain using the zeroth-order extrapolation as follows:

$$(T_e)_j = T_j \quad (2.19)$$

where the subscript  $e$  corresponds to conditions at the ghost cell by the outlet, and  $j$  indexes the boundary cell at the outlet. The number density at the outlet boundary is then computed using ideal gas equation:

$$n_e = \frac{P_e}{k(T_e)_j} \quad (2.20)$$

The particle flux conservation scheme was also applied by Nance et al. [50] at the inlet boundary and Wu et al. [49] at both boundaries. However, the authors determined the mean flow velocities on a per-cell basis and computed the particle

flux using a Maxwellian distribution instead of counting the actual number of particles crossing the computational boundary.

### 2.8.1.2 Characteristic Theory Based Scheme

Nance et al. [50] appealed to the Whitefield's characteristic theory [55], which is widely used in continuum calculations to derive boundary conditions for subsonic rarefied flows, and calculated the outflow properties at the outlet boundary. Wang and Li [51] implemented the theory of characteristics for both inflow and outflow boundaries. Liou and Fang [52] implemented a simpler approach in which they used the specified exit pressure to initialize the molecules at the outlet using the theory of characteristics, and proposed the stream-wise mean velocity at the inlet boundary to be extrapolated in an implicit manner from the neighbour cell. Wang and Li [51] and Liou and Fang [52] calculated the local pressure at the outlet boundary using the overall temperature. White et al. [53] proposed a new form of the above scheme to account for rotational non-equilibrium effects. They stated that there are consequences of considering non-equilibrium between the translational and rotational energy modes such as gas flows in a microchannel with a high inlet to exit pressure ratio or at small aspect ratio, and flows of diatomic and polyatomic gases. In such conditions, White et al. [53] suggested that the pressure should be measured using the translational kinetic temperature instead of the overall temperature. According to the kinetic theory of dilute gases, the overall kinetic temperature is defined as an average mean of the translational and internal temperatures, and the ideal gas equation does not apply to the overall temperature in a non-equilibrium situation [3]. Thus, they computed the translational temperature using the perfect gas law, and the rotational temperature using the zeroth-order extrapolation from the nearest boundary cell of the computational domain. The implementation of this boundary condition is summarized below.

At the inflow boundary, the inlet pressure and temperature,  $P_{in}$  and  $T_{in}$ , are known. The number density  $n_{in}$  is obtained from the ideal gas equation of state:

$$n_{in} = \frac{P_{in}}{kT_{in}}$$

where  $k$  is the Boltzmann constant.

The average inlet velocities at the boundary cells are determined as follows:

$$(u_{in})_j = u_j + \frac{P_{in} - P_j}{\rho_j a_j}$$

$$(v_{in})_j = v_j$$

$$(w_{in})_j = w_j$$

where the subscript  $in$  corresponds to the value at the ghost cell by the inlet,  $j$  indexes the boundary cell at the inlet, the density  $\rho$  ( $\rho = n\bar{m}$ ) and the local speed of sound  $a$  are obtained from the boundary cell at the current time step.



Similarly, at the subsonic outflow boundary of the computational domain, the only known flow parameter is the exit pressure  $P_e$ . The other flow quantities such as density and mean flow velocity are extrapolated from the adjacent cell using the theory of characteristics as follows:

$$\begin{aligned}(\rho_e)_j &= \rho_j + \frac{P_e - P_j}{a_j^2} \\(u_e)_j &= u_j + \frac{P_j - P_e}{\rho_j a_j} \\(v_e)_j &= v_j \\(w_e)_j &= w_j \\(T_e)_j &= \frac{P_e}{R(\rho_e)_j}\end{aligned}$$

where the subscript  $e$  corresponds to conditions at the ghost cell by the outlet,  $j$  indexes the boundary cell at the outlet, and  $R$  is the ordinary gas constant related to the Boltzmann constant  $k$  by  $k = mR$  and  $m$  is the molecular mass.

### 2.8.1.3 Zeroth-Order Extrapolation Scheme

Due to the nature of the particle flux method, fluctuations in the imposed flow parameters at the boundaries can be rather large under low-speed flow conditions, which make the numerical solution unstable. Based on this issue, Liou and Fang [52] proposed an alternative method using zeroth-order extrapolation to update the boundary information obtained from the previous iteration. They showed that this significantly improves the stability of the numerical solution. The inlet stream velocity is determined in an implicit manner from the boundary cell according to the equation:

$$(u_{in})_j = u_j$$

Farbar and Boyd [54] applied this scheme at both inlet/outlet boundaries; however, they computed the properties at the boundary as a weighted average of the instantaneous value and that calculated from previous time step. The equations are summarized as follows:

$$\begin{aligned}(u_{in}^t)_j &= (\alpha u'_{in})_j + (1 - \alpha) (u_{in}^{t-1})_j \\(u_e^t)_j &= (\alpha u'_e)_j + (1 - \alpha) (u_e^{t-1})_j \\(n_e^t)_j &= (\alpha n'_e)_j + (1 - \alpha) (n_e^{t-1})_j \\T_e &= \frac{P_e}{k (n_e^t)_j}\end{aligned}$$

where the prime values correspond to the instantaneous properties,  $\alpha$  is a weight parameter between 0 and 1.

### 2.8.1.4 Iteration Time Step Based Correction Scheme

Yang et al. [56] proposed an improved pressure boundary treatment which introduces the pressure information into the velocity calculation and, in addition, considers the time step and the cell length into consideration. They demonstrated that this scheme leads to major improvements in terms of convergence and applicability compared to other methods. The equations proposed by Yang et al. [56] at inlet/outlet boundaries are given by the below equations:

$$\begin{aligned}
 (u_{in})_j &= u_j + \frac{P_{in} - P_j}{\rho_j \Delta x_c} \Delta t \\
 (\rho_e)_j &= \rho_j + \frac{P_e - P_j}{(\Delta x_c)^2} (\Delta t)^2 \\
 (u_e)_j &= u_j + \frac{P_j - P_e}{\rho_j \Delta x_c} \Delta t \\
 (v_e)_j &= v_j \\
 (w_e)_j &= w_j \\
 (T_e)_j &= \frac{P_e}{R(\rho_e)_j}
 \end{aligned}$$

In these equations,  $\Delta x_c$  represents the cell size.

## 2.8.2 Methodologies of Particle Generation

Flow boundaries in a DSMC simulation are implemented by injecting particles into the computational domain corresponding to the external flow conditions. Two methodologies are proposed for their implementation in DSMC: standard and reservoir method. Once the macroscopic properties at the inlet/outlet boundaries are determined, the number of molecules entering the flow field and their molecular velocities are determined accordingly to the reservoir or standard method. The work done by Lilley et al. [57] and Perez et al. [58] aim to study the computational efficiency of the two methodologies of inflow/outflow boundaries in subsonic gas flows for both low and high-speed regimes. The results agree that inclusion of the pressure correction term in the inflow/outflow boundaries improve the computational efficiency in both methods with the standard method behaves slightly better in convergence rate than the reservoir method in subsonic gas flows. The two methodologies are discussed in separate sections below:

### 2.8.2.1 Reservoir Method

The well-known Maxwellian Reservoir method corresponds to ghost cells located at the boundaries of the DSMC computational domain (see Figure 2.5), and acted as sink and source for the simulation particles. The volume generation reservoir

algorithm, described in Tysanner and Garcia [59], fills a fixed reservoir volume at the beginning of each time step by random placement of an adequate number of particles with randomly assigned, appropriately distributed velocities drawn from a Maxwellian distribution at the simulated temperature. The particle positions are advanced in time (one time step  $\Delta t$ ); particles that cross the boundary and enter the simulation domain over a time step become part of it; the remainder are discarded and the simulation proceeds as usual. It is essential to mimic a reservoir of infinite capacity by refreshing it with an entirely new set of particles at each time step. The only subtlety is ensuring that the reservoir depth is sufficient  $L_R \geq u_{max}\Delta t$  ( $\Delta t$ : time that the particle travels within the cell;  $u_{max}$ : particles maximum velocity normal to the local domain boundary) so that it is extremely improbable that a particle could travel more than one depth of a reservoir at single time step. To simulate an equilibrium reservoir of volume  $V_R$  at number density  $n_R$  and temperature  $T_R$ , the algorithm proceeds as follows:

1. Compute the number of particles to be generated within the reservoir as  $N_R = n_R V_R$ .
2. Randomly assign the molecular velocity components of each particle according to the Maxwell-Boltzmann distribution about the reservoir cell temperature  $T_R$  [Bird 1994]

$$P(v) = \sqrt{\frac{m}{2\pi k T_R}} e^{-mv^2/2kT_R}$$

where  $m$  is the molecular mass,  $k$  is the Boltzmann constant. A flow velocity may be assigned to a reservoir to model inflow or outflow boundaries.

3. Move the particles at their assigned velocity for time  $\Delta t$ ; retain the particles that cross the boundary into the DSMC domain and disregard any that do not enter the domain.

### 2.8.2.2 Standard Method

This method involves a particle emission surface set at the flow boundary (see Figure 2.6). The number of particles to be injected during a time step is determined from the number flux and their velocities are generated from surface distribution. Boundary Conditions in DSMC simulations are often based on the equilibrium Maxwell-Boltzmann distribution function. Assuming a Maxwellian velocity distribution, the number of flowing-in molecules from the inflow or outflow boundaries in a given time step with a mean stream velocity  $\mathbf{V}$  and temperature  $T$  is given by the following equation [3]:

$$N = \Delta t A n \frac{1}{2\beta\sqrt{\pi}F_N} \left\{ e^{-s^2 \cos^2\theta} + \sqrt{\pi} s \cos\theta [1 + \text{erf}(s \cos\theta)] \right\} \quad (2.21)$$

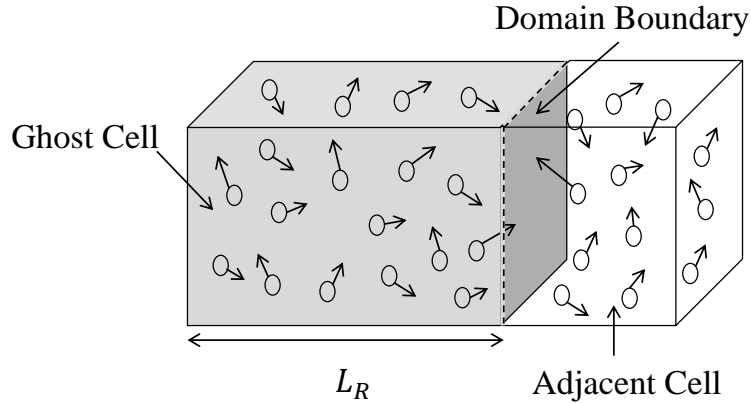


Figure 2.5: Maxwellian Reservoir particle generation method: Ghost cell is indicated in gray ( $L_R$ : reservoir depth).

where  $n$  is the number density,  $s = \beta V$  is the molecular speed ratio i.e. the ratio of the mean stream speed to the most probable speed,  $\beta$  is the reciprocal of the most probable molecular speed ( $= \sqrt{2RT}$ ),  $F_N$  is the number of real molecules represented by each simulated molecule,  $A$  is the cross-sectional area,  $R$  is the universal gas constant,  $\theta$  is the angle between the stream velocity vector and the unit normal vector to the surface element, and erf is the error function.

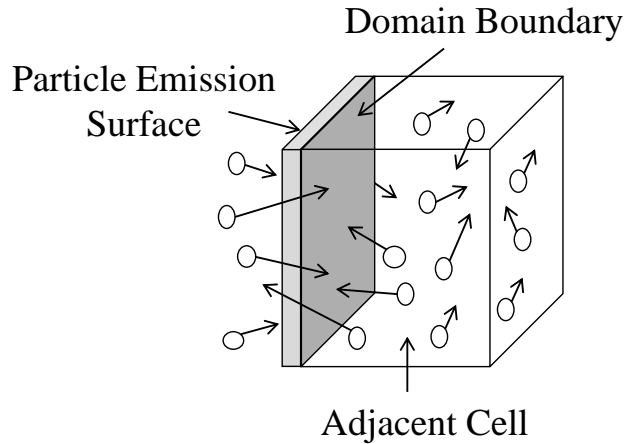


Figure 2.6: Standard particle generation method: The surface which emits particles is represented by the shaded region.

The velocity components of the molecules entering the flow field are generated according to the velocity distribution characteristic of the external flow. Velocity components  $v$  and  $w$  parallel to the boundary are generated independently from the ordinary Maxwell-Boltzmann distribution function, while the velocity component  $u$  normal to the inflow/outflow boundary that corresponds to the bi-

ased Maxwell-Boltzmann distribution is generated according to the acceptance-rejection method.

Consider a gas flow across a typical stream boundary in a DSMC simulation with stream velocity vector of normal component  $U$  such that  $U > 0$  for an inlet boundary and  $U < 0$  for an outlet boundary, then the molecular and thermal velocity normal components are given by  $u$  and  $u' = u - U$ , respectively. The probability of a molecule entering the inflow/outflow boundary is proportional to the velocity normal to the boundary  $u$ . The molecules are randomly located at the boundary and their molecular velocity distribution is given as [3]:

$$f(u) = C^{-1}u \exp(-\beta^2 u'^2)$$

where  $C^{-1}$  is the normalization constant calculated using the normalization condition that is the integration with respect to  $u$  from 0 to  $\infty$  must be equal to unity, and is given by:

$$2\beta^2 C = s\sqrt{\pi}(1 + \operatorname{erf} s) + \exp(-s^2)$$

where  $s = \beta U$  is the non-dimensional speed ratio. Accordingly, the ratio of the probability to the maximum probability is given by [3]:

$$\frac{P}{P_{max}} = \frac{2(\beta u' + s)}{\phi} \exp(\chi - \beta^2 u'^2) \quad (2.22)$$

where  $\phi = s + \sqrt{s^2 + 2}$  and  $2\chi = 1 + s [s - \sqrt{s^2 + 2}]$ .

The procedure for the generation of typical velocity components from a prescribed probability density distribution function  $f(u)$  in the acceptance-rejection algorithm is done in two steps. First, a random value of  $u$ , denoted  $u^*$ , that is uniformly distributed within the range of interest is generated. Second, the value of  $f(u^*)/f(u)$  is then evaluated. If  $f(u^*)/f(u) > R_f$ , where  $R_f$  is a uniformly distributed random fraction between 0 and 1, then the chosen  $u^*$  is accepted; otherwise, it is rejected and the loop must be restarted by selecting a new random  $u^*$  until an accepted value is attained. It is necessary to limit the range of the selected normal velocity  $u^*$ . The acceptance-rejection method replaced the upper limit of  $\infty$  by a limit value of  $3/\beta$ ; therefore, the thermal velocity  $u'$  is randomly sampled in the interval  $[-U, 3/\beta]$  and the stream total velocity of entering molecules,  $u = U + u'$ , is then between  $u_{min} = 0$  and  $u_{max} = U + 3/\beta$ . Thus, the velocities of molecules entering the simulation space are given as follows:

$$\begin{aligned} u &= (U + 3/\beta) R_f \\ v &= \left( \sqrt{-\ln(R_f)}/\beta \right) \cos(2\pi R_f) + V \\ w &= \left( \sqrt{-\ln(R_f)}/\beta \right) \sin(2\pi R_f) + W \end{aligned}$$

where  $V$  and  $W$  represent the local mean flow velocity components in parallel directions to the flow boundary. Once the velocity components of entering molecules are obtained, they are moved just off the boundary and their location is determined by the generation of further values of  $R_f$ .

## 2.9 Particle Collisions

Simulating molecular collisions is a statistical process which allows DSMC to achieve faster numerical performance than deterministic simulation methods such as molecular dynamics. The concept of collision in DSMC implies short-range intermolecular interactions described through Boltzmann equation [39]. Under the assumptions of molecular chaos and a rarefied gas, where the mean molecular diameter is much smaller than the mean molecular space in the gas, only binary collisions between the gas particles need to be considered, and particles that are near each other are selected as collision partners. In consequence, molecular motions are decoupled from intermolecular collisions over a time interval smaller than the physical collision time.

To evaluate molecular collisions in the gas, the physical domain is partitioned into collision cells, and the molecules are sorted into these cells. Several collisional modeling techniques have been applied successfully within the framework of DSMC. All simulate an appropriate choice of collision pairs, and implement a certain number of representative collisions between randomly selected pairs of molecules within each cell. Roohi et al. [60] reported a comprehensive review of the different collision models developed in the framework of the DSMC method.

The probability of a collision between two molecules in a homogeneous gas is proportional to the product of the relative velocity of the colliding partners  $v_r$  and the total collision cross section  $\sigma$ . The mean value of the product is computed in each collision cell, and the maximum value is recorded. The collision pairs are selected based on the acceptance-rejection method, and the probability of the chosen pair depends on the ratio of their product to the maximum product. This method would have a computation time proportional to the square of the total number of molecules. Bird in 1994 has introduced the no-time counter method (NTC) technique for the computation time to be directly proportional to the number of molecules. In conjunction with this technique, the sub-cell method (Bird, 1988 [61]) that restricts possible collision pairs to sub-cells is used. This procedure ensures that collisions occur only between near neighbors, and thereby reduces the memory requirements and improves the accuracy of the simulation.

### 2.9.1 Molecular Models in DSMC

During the collision process, intermolecular collisions occur in each cell on the basis of probability. This probability is proportional to the relative velocity between

the two molecules and the total collision cross section. In DSMC simulation, the total collision cross-section  $\sigma_T$  may be interpreted as the area around a collision molecule in which the center of the incoming molecule cannot penetrate. It is written as:

$$\sigma_T = 2\pi \int_0^\pi \sigma \sin\chi d\chi = 2\pi \int b db \quad (2.23)$$

where  $\sigma$  is the cross sectional area defined as [62]

$$\sigma = \frac{b}{\sin\chi} \left| \frac{db}{d\chi} \right| \quad (2.24)$$

where  $b$  is an impact parameter of the distance of closes approach between the pre-collision/post-collision velocities of the two collision molecules, and  $\chi$  is the deflection angle as shown in Figure 2.7. Different collision models based on

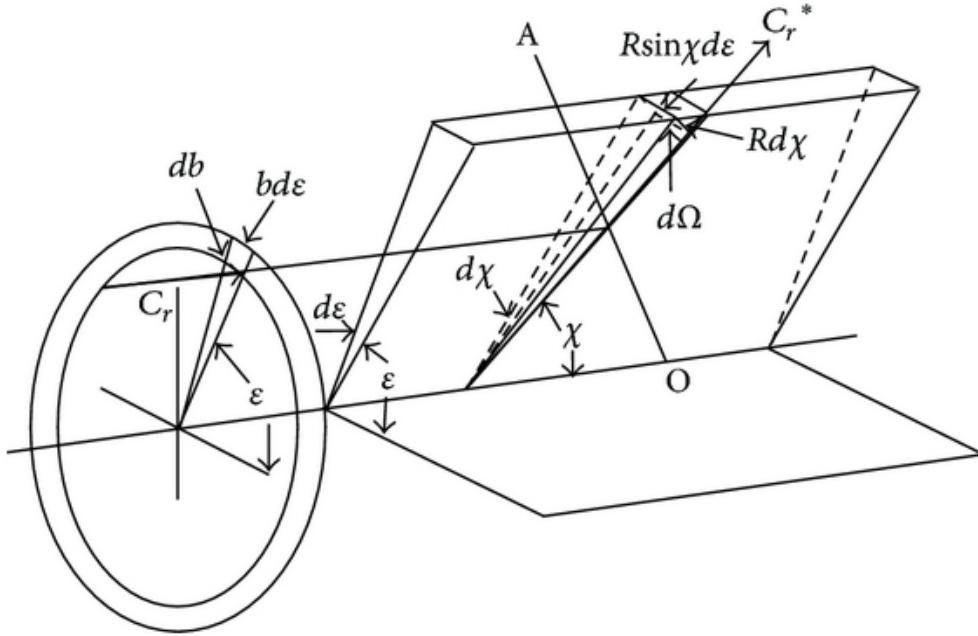


Figure 2.7: Illustration of the impact parameters [3]

the kinetic theory of gases exist in literature and can be applied to represent intermolecular collisions. Three most frequent molecular models, including the classical hard sphere (HS), the variable hard sphere (VHS), and the variable soft sphere (VSS) model, are used in the simulation.

**2.9.1.0.1 The Hard Sphere Model (HS)** The HS model is the simplest molecular model in which the gas molecule has a fixed diameter  $d$ , the closest approach  $b$  is defined as  $b = d_{12} \cos(\frac{1}{2}\chi)$ , where  $d_1$  and  $d_2$  are the diameters of the colliding molecules, and thereby the total collision cross section defined as

$\sigma_T = \pi d_{12}^2$  is constant. The molecular scattering is isotropic so that the post-collision velocity is uniform in solid angle. In HS model, the molecule diameter can be calculated as

$$d_{\text{HS}} = \left[ \frac{(5/16)(mkT_{\text{ref}}/\pi)^{1/2}}{\mu_{\text{ref}}} \right]^{1/2} \quad (2.25)$$

where  $\mu_{\text{ref}}$  is the reference viscosity at reference temperature  $T_{\text{ref}}$ ,  $m$  is the molecular mass, and  $k$  is the Boltzmann constant  $k = 1.380658 \times 10^{-23} \text{ J/K}$ . The scattering law in HS model is not realistic as  $\sigma_T$  is a strong function of  $v_r$  and the translational energy  $E_r = (1/2)mv_r^2$  for a real gas. To correct this primary deficiency of the HS model, the variable hard sphere model (VHS) was introduced by (Bird 1981 [63]).

**2.9.1.0.2 The Variable Hard Sphere Model (VHS)** In the VHS model, the molecule is a hard sphere with diameter  $d$  that is a function of the relative velocity  $v_r$ ; as follows:

$$d_{\text{VHS}} = d_{\text{ref}} \left( \frac{c_{r,\text{ref}}}{v_r} \right)^\xi = \left[ \frac{(15/8)(m/\pi)^{1/2}(kT_{\text{ref}})^\omega}{\Gamma((9/2) - \omega) \mu_{\text{ref}} \epsilon_t^{\omega - (1/2)}} \right]^{1/2} \quad (2.26)$$

where the subscript "ref" denotes the reference values at the reference temperature  $T_{\text{ref}}$ ,  $\xi = 2/(\eta - 1) = \omega - 1/2$  where  $\eta$  is the power determining the "hardness" of molecules from the inverse power law (IPL) model, and  $\omega$  is the viscosity index comes from the power-law temperature dependence of the coefficient of viscosity by  $\mu \propto T^\omega$ ,  $\epsilon_t = (1/2)m_r v_r^2$  is the relative mean kinetic energy. In the VHS model, the closest approach  $b$  can be defined as

$$b = d_{\text{VHS}} \cos \left( \frac{1}{2} \chi \right) \quad (2.27)$$

The total collision cross section is expressed as:

$$\sigma_T = \sigma_{T,\text{ref}} \left( \frac{d_{\text{VHS}}}{d_{\text{ref}}} \right)^2 \quad (2.28)$$

In a DSMC simulation,  $\sigma_{T,\text{ref}}$ ,  $T_{\text{ref}}$ ,  $d_{\text{ref}}$ , and  $\omega$  are known parameters,  $v_r$  is determined and the value of  $\sigma_T$  is calculated. Typical values of  $\omega$  and  $d_{\text{ref}}$  are given in Table 3 (Bird 1994 [3]).

For both hard sphere models (HS) and (VHS), the deflection angle of the relative velocity is given by:

$$\chi = 2 \cos^{-1} \left( \frac{b}{d_{12}} \right) \implies \cos(\chi) = 2 \left( \frac{b}{d_{12}} \right)^2 - 1 \quad (2.29)$$



The factor  $\left(\frac{b}{d_{12}}\right)^2$  is uniformly distributed between 0 and 1. Thereby, the scattering in the HS and VHS models is isotropic and fully covers the plane ( $-\pi$  to  $\pi$ ). The scattering angle is numerically generated in the DSMC code as:

$$\cos(\chi) = 2(R_f) - 1 \quad (2.30)$$

where  $R_f$  is a random number generated between 0 and 1.

Unlike HS model, the effective diameter is an energy-dependent variable in the VHS model. However, both models obey the isotropic scattering law and do not correctly predict the molecular diffusions in real gases flows especially of gas mixtures. The variable soft sphere model (VSS) [64, 65, 66] was introduced to account for the anisotropic scattering of real gases.

**2.9.1.0.3 The Variable Soft Sphere Model (VSS)** In the VSS model, the molecular diameter varies in the same way as the VHS model, but there is a deflection exponent,  $\alpha$ , in the relation of the closest approach  $b$  such that:

$$b = d_{\text{VSS}} \cos^\alpha \left( \frac{1}{2} \chi \right) \quad (2.31)$$

This model is called variable soft sphere because  $\chi_{\text{VSS}}$  is smaller than  $\chi_{\text{VHS}}$ . The molecular diameter is defined as:

$$d_{\text{VSS}} = \left[ \frac{5(\alpha + 1)(\alpha + 2)(m/\pi)^{1/2}(kT_{\text{ref}})^\omega}{16\alpha\Gamma((9/2) - \omega)\mu_{\text{ref}}\epsilon_t^{\omega-(1/2)}} \right]^{1/2} \quad (2.32)$$

where  $\alpha$  is the scattering coefficient ( $\alpha$  is 1 for the VHS model). The VSS total collision cross section can be expressed as:

$$\sigma_T = \pi d_{\text{VSS}}^2 \quad (2.33)$$

The deflection angle  $\chi$  in the VSS model is given by:

$$\chi = 2 \cos^{-1} \left( \left( \frac{b}{d_{12}} \right)^{1/\alpha} \right) \implies \cos(\chi) = 2 \left( \frac{b}{d_{12}} \right)^{2/\alpha} - 1 \quad (2.34)$$

The factor  $\left(\frac{b}{d_{12}}\right)^2$  is uniformly distributed between 0 and 1. Thereby, the scattering in the VSS model is not isotropic, and the scattering angle is numerically generated in the DSMC code as:

$$\cos(\chi) = 2(R_f)^{\frac{1}{\alpha}} - 1 \quad (2.35)$$

where  $R_f$  is a random number generated between 0 and 1.

## 2.9.2 Collision Pairs

In the NTC collision procedure, the number of particle pairs that should be checked for collision within a collision cell of volume  $V_c$  at a time step ( $\Delta t$ ), is defined as:

$$N_{collisionpairs} = 1/2 N \bar{N} F_N (\sigma_T v_r)_{max} \Delta t / V_c \quad (2.36)$$

where  $N$  is the instantaneous number of simulated molecules in a cell at a time step  $\Delta t$ ,  $\bar{N}$  is the time average number of simulated molecules in a cell,  $F_N$  is the number of real molecules represented by each simulated molecule,  $\sigma_T$  is the total collision cross section, and  $v_r$  is the relative speed in the collision. In 2007, Bird has proposed a modification to the NTC method where  $N(N-1)/V_c$  replaces  $N\bar{N}/V_c$  in the equation of possible collision pairs. Referring to his notes [67], this solves three problems associated with the NTC method. First, the above equation includes the term  $\bar{N}F_N/V_c$  which is the number density and it is undesirable for a microscopic collision procedure to depend on a macroscopic flow property. Second, the collision procedure requires two molecules and there will be a problem if one molecule per collision cell is presented. Finally, it takes time to establish the value of averaged quantities such as number density. Thus, the number of possible collision pairs becomes:

$$N_{collisionpairs} = 1/2 N (N - 1) F_N (\sigma_T v_r)_{max} \Delta t / V_c \quad (2.37)$$

Thus, if  $\Delta t$  is kept constant and the volume  $V_c$  is divided into eight octants, we obtain  $N_{collisionpairs} \sim 8(\frac{N}{8})(\frac{N}{8} - 1)\Delta t / (\frac{V_c}{8}) \sim N(N-8)\Delta t / V_c$ , which means that  $N_{collisionpairs}$  is conserved as long as  $N \gg 1$ . Furthermore, if  $N$  and  $V_c$  are kept constant and  $\Delta t$  is divided by  $m$ , then  $N_{collisionpairs} \sim N(N-1)m(\Delta t/m)/V_c$  and  $N_{collisionpairs}$  is conserved if we carry  $m$  collisions. From kinetic theory, the collision probability for hard sphere gases is proportional to their relative velocity. Once the number of possible collision pairs is determined within each cell  $N_c$ , each pair  $(i, j)$ ,  $1 \leq i < j \leq N_c$  chosen at random from within the same subcell of a collision cell, is checked for a collision probability test where the pair of particles is accepted for collision if:

$$R_f < \frac{\sigma_T v_r}{(\sigma_T v_r)_{max}} \quad (2.38)$$

where  $\sigma_T v_r$  represent the product of the mean relative velocity and the collision cross section of pair  $(i, j)$ , and  $R_f$  is a uniform random number ranging from 0 to 1. If the pair is rejected, a new pair is randomly chosen and the procedure is repeated until the required number of candidate pairs  $N_c$  in the collision cell has processed.

## 2.9.3 Post Collision Velocities

After the collision pair is chosen, their post-collision velocities,  $\mathbf{v}'_i$  and  $\mathbf{v}'_j$ , need to be evaluated. From conservation of linear momentum, the center-of-mass ( $\mathbf{v}_{cm}$ )

velocity remains unchanged by the collision,

$$\mathbf{v}_{cm} = \frac{1}{2} (\mathbf{v}_i + \mathbf{v}_j) = \frac{1}{2} (\mathbf{v}'_i + \mathbf{v}'_j) = \mathbf{v}'_{cm} \quad (2.39)$$

The conservation of energy states that the magnitude of the relative velocity is also unchanged by collision,

$$v_r = \|\mathbf{v}_i - \mathbf{v}_j\| = \|\mathbf{v}'_i - \mathbf{v}'_j\| = v'_r \quad (2.40)$$

The conservation of momentum and energy provide four of the six constraints in  $\mathbf{v}'_i$  and  $\mathbf{v}'_j$  needed to determine the post-collision velocities. The two remaining unknowns are selected at random with the assumption that the direction of the post-collision relative velocity is uniformly distributed over the unit sphere,

$$\mathbf{v}'_r = v_r [(sin\theta cos\phi)\hat{\mathbf{x}} + (sin\theta sin\phi)\hat{\mathbf{y}} + cos\theta\hat{\mathbf{z}}] \quad (2.41)$$

where  $\phi$  is the azimuthal impact angle uniformly distributed between 0 and  $2\pi$ , and so it is computed using a random number as  $\phi = 2\pi R_f$ , and  $\theta$  is the deflection angle defined as the angle between the pre-collision relative velocity and the post-collision relative velocity and distributed according to the probability,

$$P(\theta)d\theta = \frac{1}{2} sin\theta d\theta \quad (2.42)$$

The probability equation can be written as  $P(q)dq = \frac{1}{2}dq$  under the change of variable  $q = cos\theta$ , and so  $q$  is uniformly distributed in the interval  $[-1,1]$  and the deflection angle is computed using another random number  $R_f$ ,

$$\begin{aligned} cos(\theta) &= 2R_f - 1 \\ sin(\theta) &= \sqrt{1 - cos^2\theta} \end{aligned} \quad (2.43)$$

For the VSS model, the post-collision relative velocity depends on the value of the parameter  $d$  (Bird 1994), it is given by:

$$d = \sqrt{\|\mathbf{v}_{ry}\|^2 + \|\mathbf{v}_{rz}\|^2} \quad (2.44)$$

If  $d > 10^{-6}$ , the post-collision relative velocity components are given by:

$$\begin{aligned} v'_{rx} &= b\|\mathbf{v}_{rx}\| + a sin(c)d \\ v'_{ry} &= b\|\mathbf{v}_{ry}\| + a \left( \frac{\|\mathbf{v}_r\|\|\mathbf{v}_{rz}\|cos(c) - \|\mathbf{v}_{rx}\|\|\mathbf{v}_{ry}\|sin(c)}{d} \right) \\ v'_{rz} &= b\|\mathbf{v}_{rz}\| - a \left( \frac{\|\mathbf{v}_r\|\|\mathbf{v}_{ry}\|cos(c) + \|\mathbf{v}_{rx}\|\|\mathbf{v}_{rz}\|sin(c)}{d} \right) \end{aligned} \quad (2.45)$$

If  $d < 10^{-6}$ , the post-collision relative velocity components are given by:

$$\begin{aligned}v'_{rx} &= b\|\mathbf{v}_{rx}\| \\v'_{ry} &= a\cos(c)\|\mathbf{v}_{rx}\| \\v'_{rz} &= a\sin(c)\|\mathbf{v}_{rx}\|\end{aligned}\tag{2.46}$$

where  $b = 2(R_{f_1})^{\frac{1}{\alpha}} - 1$ ,  $a = \sqrt{1 - b^2}$ , and  $c = 2\pi(R_{f_2})$ ,  $R_{f_1}$  and  $R_{f_2}$  being two random numbers between 0 and 1.

The post-collision velocities of the two molecules in a collision pair  $(i, j)$  can be obtained as:

$$\begin{aligned}\mathbf{v}'_i &= \mathbf{v}'_{cm} + \frac{1}{2}\mathbf{v}'_r \\ \mathbf{v}'_j &= \mathbf{v}'_{cm} - \frac{1}{2}\mathbf{v}'_r\end{aligned}\tag{2.47}$$

This assumption is exact for the hard-sphere model and has been found to be an excellent approximation in general [3]

## 2.10 Sampling

As DSMC is inherently a stochastic method, the macroscopic flow properties of interest are computed as averages at the geometric center of the collision cells. In the sampling process, the number of simulated molecules  $N_c$ , the number density  $n_c$ , the three velocity components  $u_c$ ,  $v_c$ , and  $w_c$ , are the main macroscopic physical quantities sampled in a particular cell of volume  $V_c$ . The number density,  $n_c$ , is given as:

$$n_c = \frac{N_c N_{realpersim}}{N_{samples} V_c}\tag{2.48}$$

where  $N_{samples}$  is the number of samples collected, and  $N_{realpersim}$  is the number of real molecules represented by one simulated molecule.

The three velocity components are computed according to:

$$\begin{aligned}u_c &= \frac{1}{N_c} \sum_{i=1}^{N_c} u_i \\ v_c &= \frac{1}{N_c} \sum_{i=1}^{N_c} v_i \\ w_c &= \frac{1}{N_c} \sum_{i=1}^{N_c} w_i\end{aligned}\tag{2.49}$$

where the sum is over simulated molecules in the statistics cell of volume  $V_c$ . Once the three macroscopic velocity components are sampled, the macroscopic

translational temperature is computed as the average value of the three non-equilibrium temperature components according to:

$$T = \frac{1}{3}(T_x + T_y + T_z) = \left( \frac{1}{N_c} \sum_{i=1}^{N_c} (u_i^2 + v_i^2 + w_i^2) - (u_c^2 + v_c^2 + w_c^2) \right) \frac{m}{3k} \quad (2.50)$$

where  $m$  is the molecular mass and  $k$  the Boltzmann constant.

Next, the pressure can be found using the ideal gas law  $P_c = n_c k T_c$  (if it is valid), or by integrating the change in particle momentum over all collisions within a solid surface (for example, high Mach number condition where the pressure is not isotropic) i.e. explicitly evaluating the pressure tensor [37].

Unless the number of computational particles in a given cell is large, there will be unacceptably large uncertainties in the average quantities. Several samples of each cell are therefore necessary. The DSMC method is an explicit time marching simulation algorithm. As such, it always produces unsteady flow simulations. The statistical scatter of a DSMC computation varies according to the inverse square root of the sample size. The DSMC simulation starts from a set of prescribed initial conditions and proceeds in small time steps identified with the physical time in the real flow. Figure 2.8 illustrates the two sampling methods used in DSMC: steady and unsteady sampling techniques. For predicting steady flows, each independent Monte Carlo simulation proceeds until a steady behaviour is established at a sufficiently large time. Time-averaging of the macroscopic flow quantities over a number of time steps is required to reduce statistical fluctuations and obtain smooth results. For simulating unsteady flows, ensemble averaging of many independent Monte Carlo simulations, each originating from a different random number generator seed, is carried out to obtain final results with an acceptable statistical accuracy. The flow field is sampled at the appropriate flow sampling time steps, denoted by dark bands in Figure 2.8. Unsteady flow sampling to yield the macroscopic flow quantities at a given time step requires averaging over all the independent realizations of the transient simulations at that time step. More details on unsteady sampling methods are presented in (Cave et al. 2008 [4]).

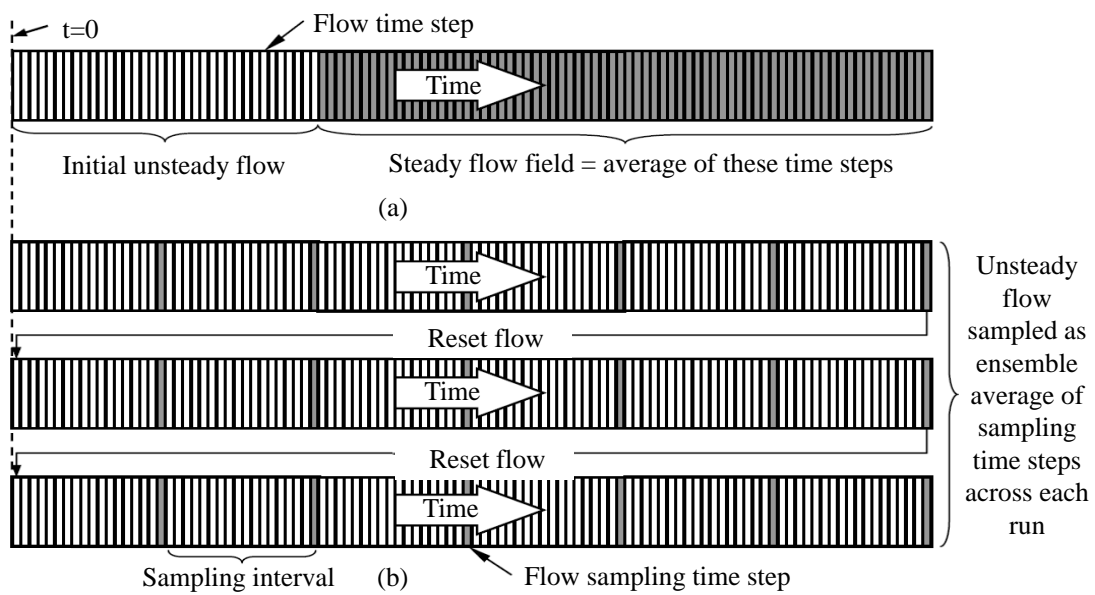


Figure 2.8: (a) The schematic of the time-averaging of the flow properties over a long interval of simulation time. (b) The schematic of the ensemble-averaging of the flow properties over three independent DSMC simulations, each initiated from a different random number generator seed [4].

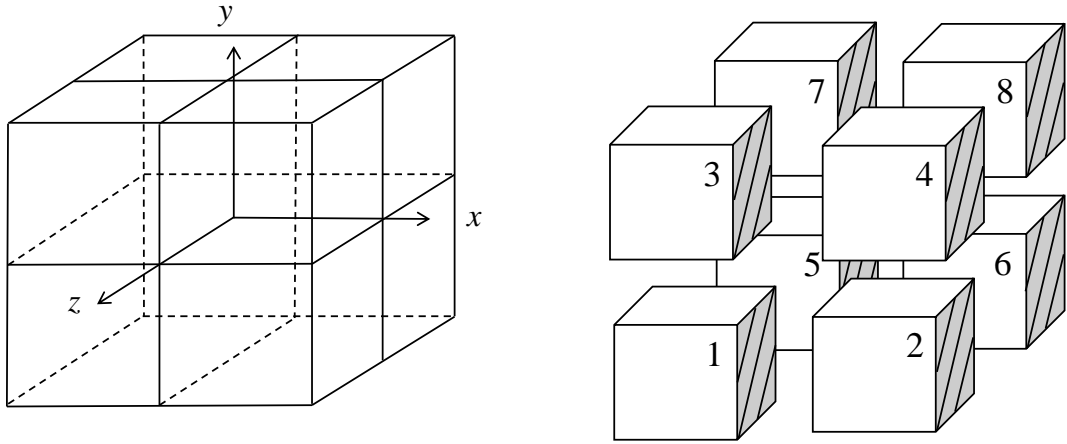
# Chapter 3

## New Parallel Adaptive Three-Dimensional DSMC Algorithm

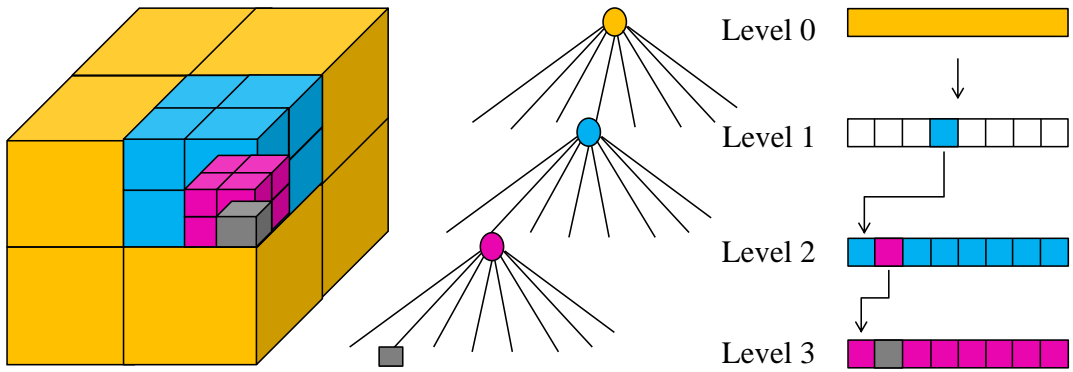
### 3.1 Octree Data Structure

An octree is a tree data structure that employs a hierarchy of tree-like sub-division of a three-dimensional domain. It recursively partitions a three-dimensional space into cubes, dividing each cube into eight equally-sized octants of halved-sized dimensions. This recursive subdivision continues until some termination criteria are met. Quad-trees are the two-dimensional analog of octrees where each node has at most four children. The enclosing region for the entire tree is the bounding box for the root node of the tree, a node with no parent. Each interior node has exactly one parent and eight child octant; however, the leaf node is a node with no children. The depth of a node from the root is referred to as its level, with the root having level 0 (refer to Figure 3.1). In-depth studies of various kinds of octrees can be found in [34]. Once the data structure is constructed, a recursive neighbor subroutine is called to efficiently find all possible neighbor nodes of each collision cell in the octree. The concept of a neighbor in a tree structures (quadtrees or octrees) is defined as follows: We say that a node  $N$  is a neighbor of node  $N'$  in direction  $I$  if  $N$  corresponds to the smallest block (it may correspond to a non-leaf node) adjacent to  $N'$  (i.e., touching even if just at a point) in direction  $I$  of size greater than or equal to the block corresponding to  $N'$ . Whereas in two dimensions we have eight possible directions, in three dimensions, we have 26 possible directions. In particular, in two dimensions, two nodes can be adjacent, and hence neighbors, along with an edge (four possibilities) or along a vertex (four possibilities). In contrast, in three dimensions, two nodes can be adjacent, and hence neighbors, along with a face (six possibilities), along with an edge (12 possibilities), or along a vertex (eight possibilities). Such neighbors are

termed face-neighbors, edge-neighbors, and vertex-neighbors, respectively. These relations are shown in Figure 3.2. The neighbor-finding process is important to traverse the tree during the ray-tracing phase of molecular movements.



(a) The hierarchical structure of the octree and the order of the eight octants



(b) Octree block decomposition of a three-dimensional object and its tree representation

Figure 3.1: Octree data structure

## 3.2 Three-Dimensional Hybrid Mesh Scheme

In the current study, an octree-based Cartesian grid divides the computational domain into cubic cells, whereas the surface of the 3D solid object is triangulated using the preprocessing open-source software SALOME 7.5.1 [68]. Tree-based methods with the simple Cartesian structure and embedded hierarchy make use of recursive encoding schemes. These schemes render processes such as mesh adaptation, rebuilding, data access, and handling of fluid-solid interaction both simple and efficient.



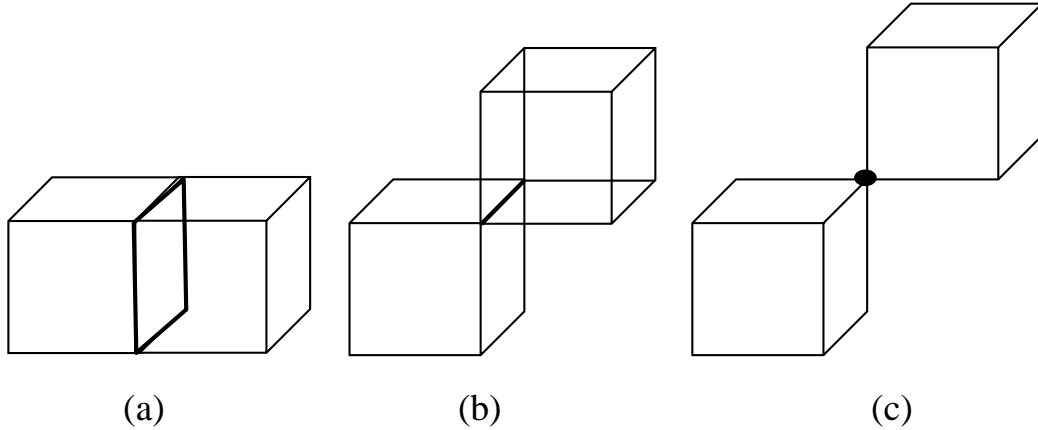


Figure 3.2: Example of node neighbors in octree structure (a) a face neighbor, (b) an edge neighbor, and (c) a vertex neighbor.

The flow domain is represented by a hybrid mesh consisting of a hierarchical octree-based Cartesian grid, where the 3D solid object is discretized into tetrahedral meshes using the open source pre-processor SALOME 7.5.1. The mesh is then exported to a data file. The simulator read in the mesh, and three-dimensional affine transformations are done (if necessary) before embedding the object within the three-dimensional grid system. The transformations are possible as the object is represented by point sets or nodes. Three-dimensional affine transformations include linear 3D transformations (translation, rotation, scaling, shearing, and reflection) and perspective transformations according to a generalized 4x4 transformation matrix in homogeneous coordinates as summarized in Figure 3.3.

A special procedure is developed to accurately identify the 3D solid object and record the object triangular surface mesh in the octree-based Cartesian structure. Figure 3.4(a) shows a schematic diagram illustrating the hybrid mesh scheme. Each solid object is bounded by a rectangular box. An axis-aligned box-box intersection test [5] is then carried out to identify collision cells neighbours and all cubic cells that overlap with the bounding box surface. In addition, the fast 3D triangle-box overlap testing by Moller [69] is implemented to test overlapping between triangular elements of the solid object surface mesh and cubic cells inside the bounding box. This test enables linking each surface mesh triangular element to the overlapping Cartesian cells. Cartesian cells intersecting any of the object's surface triangles will be given a FLAG, and the indices of the surface triangles related to each cell are re-coded. Figure 3.5 represents a schematic representation of the geometric intersection tests used to identify the 3D solid object in the octree-based Cartesian structure.

The proposed three-dimensional hybrid mesh scheme employs a flexible data

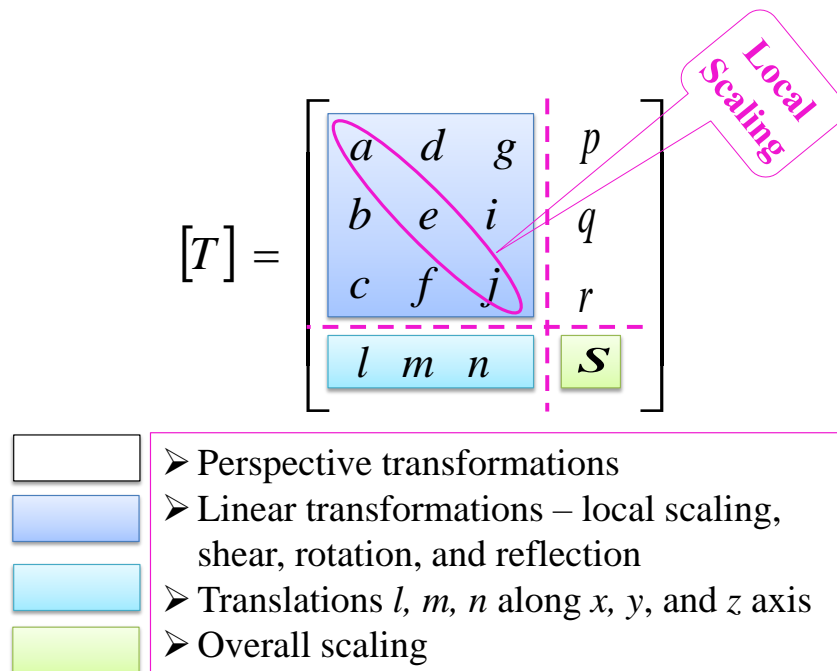


Figure 3.3: Generalized 4x4 transformation matrix in homogeneous coordinates

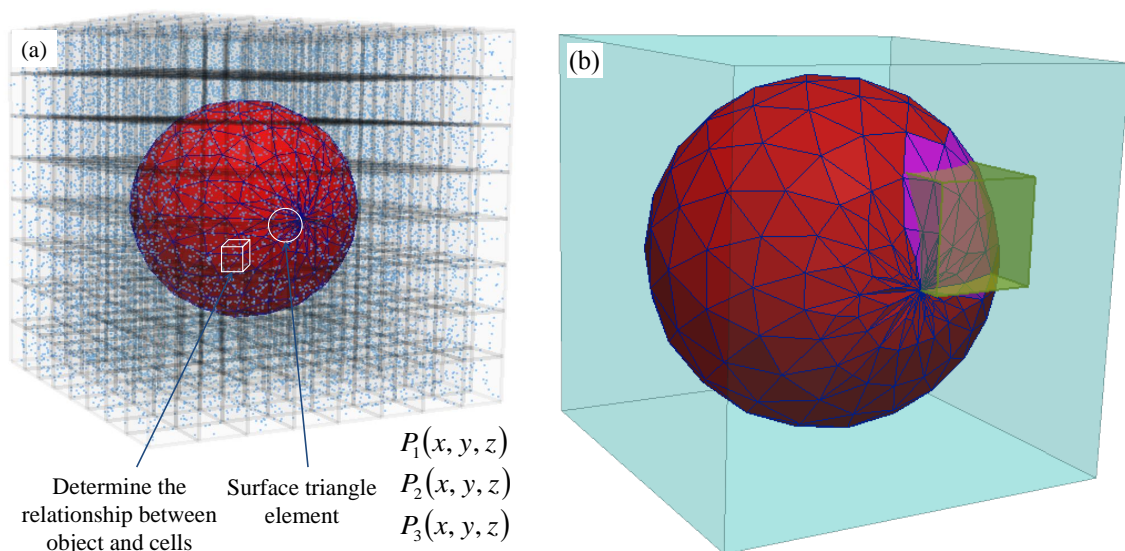
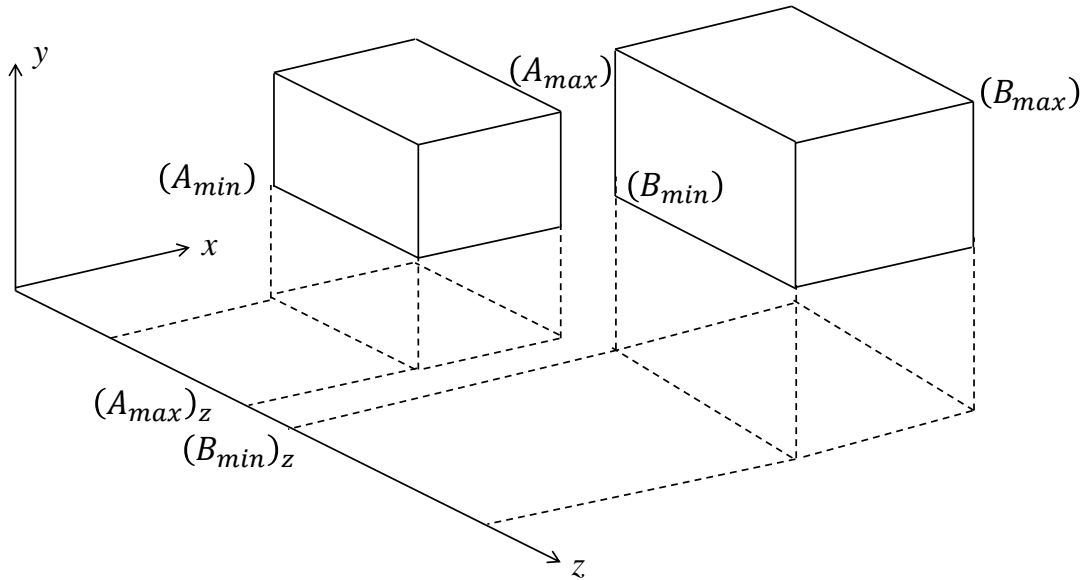
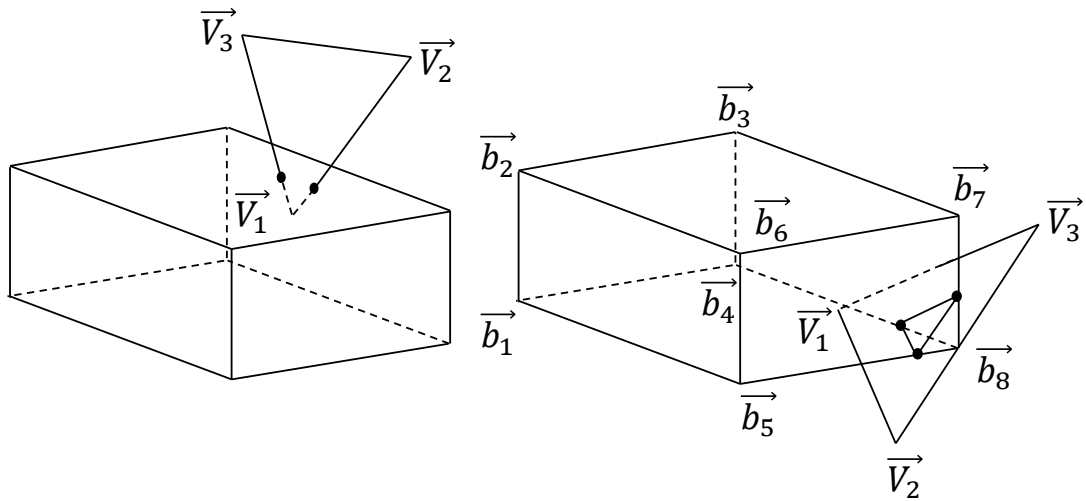


Figure 3.4: (a) Schematic representation of a triangulated surface mesh of a sphere embedded in a 3D octree Cartesian grid. (b) Bounding box and a cut-cell representation.



(a) Axis-aligned box-box intersection test. Each box is defined by its minimum and maximum vertices. The intersection is based on the separating-axis theorem.



(b) Box-Triangle intersection test. (Left) The case when one of the triangle vertices intersects the box; (Right) The plane containing the triangle intersects the box.

Figure 3.5: Geometric intersection tests to identify the complex 3D physical object in the octree-based Cartesian structure [5].

structure which enables simulation of particles movement and sorting processes with fewer operations, thereby reducing the CPU time. The derived data structure in Fortran 90 is adapted to efficiently store and manage the complete information related to the millions of particles, grid cells, and triangular surface elements during a DSMC simulation. It is a collection of different elementary data structures such as variable, pointers, linked lists, and arrays. Fortran 90 allows allocatable arrays of derived data type and arrays of derive-type objects which have pointers as components in which various components of the derived data type are accessed using percent signs. The former is used to store the complete particles' information including global position, velocity, molecule cell/sub-cell address that represents the global index of the cell/sub-cell containing this molecule, etc., and the complete boundary triangles' information including vertex nodes, triangle's normal vector, and orthonormal triangle basis. The latter is used to store the data of collision cells including cell geometry, cell neighbors, resident particles, children global indices, and boundary triangle indices.

Special treatment of the cells being crossed by the solid boundary, i.e., the so-called cut-cells, is also applied. Figure 3.4(b) shows a zoomed-in view of the box bounding the solid object (sphere) and a cut-cell representation. Two main cut-cell methodologies for DSMC simulations of rarefied gas flows around moving obstacles have been proposed [35, 70]. These methods used to estimate the cut-cell effective volume needed to accurately model collisions and predict the macroscopic properties in the cut-cell. They use the random marker-based approach and vary only in the way the immersed solid object is represented numerically. The first one discretized the boundary surface mesh into triangular facets, and the effective cell volume is computed either by polyhedron decomposition utilizing the facets and the collision cell faces into a number of pyramids and compute their volume or by Monte Carlo random marker method [35, 70]. The latter expresses the boundary surface by analytical expressions as an input [70]. The Monte Carlo random marker cut-cell method is implemented in this work [35]. It performs two main functions: First, all triangular surface elements are sorted into the appropriate octree Cartesian cells within the geometry data structure. Second, a number of particles,  $N_p$ , is randomly generated within each cell with a volume  $V$ . Possible intersections between a ray going out from a particle and directed along the unit normal vector of a given surface triangle element are determined. If no intersections occur with all surface triangle elements within the cell, the particle lies inside the flow field. Then, the volume of the cut-cell is simply determined by dividing the fraction of particles determined to lie within the flow field,  $N_c$ , by the total number of Monte Carlo particles considered,  $N_p$ , as:

$$V_c = V (N_c/N_p)$$

The error in such a volume calculation is inversely proportional to the square root of  $N_p$ . Hence, the effective cut-cell volume converges to its exact value as the

number of Monte Carlo particles used increases. In this work, an initial guess of the cut-cell volume is set equal to half the uncut Cartesian cell, and a threshold error ( $\approx 1\%$ ) is specified, and we keep increasing the number of particles within the cell until the percent error computed is below the threshold value. The cut-cell algorithm is called at the beginning of each simulation when the octree is constructed and not every time step.

### 3.3 Three-Dimensional Particle Ray-Tracing Scheme

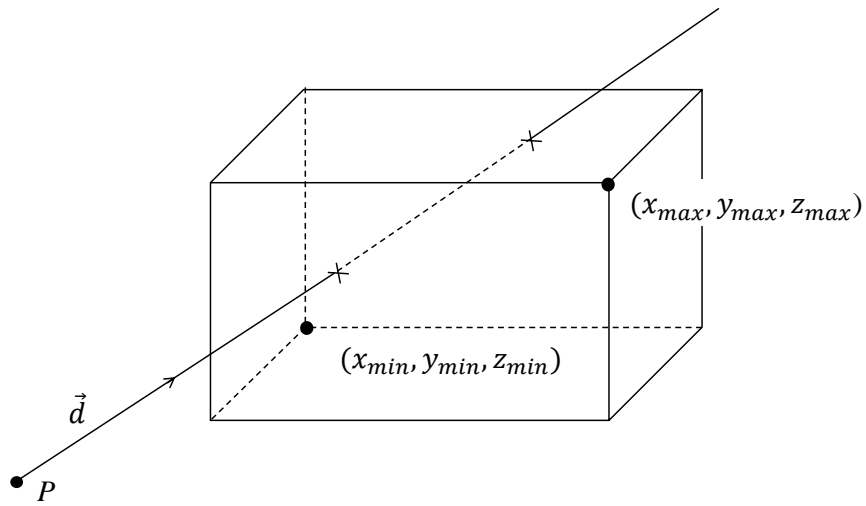
Molecular movement routine constitutes a significant fraction of the computational cost in a DSMC simulation. Simulated molecules move along linear trajectories defined in a vector form as  $\mathbf{r}_f = \mathbf{r}_i + \mathbf{v}\Delta t$ , where  $\mathbf{r}_f$  is the final position of the particle,  $\mathbf{r}_i$  is the initial position vector, and  $\Delta t$  is the simulation time step. To achieve both robustness and efficiency in tracking particle movement within the hierarchical octree-based Cartesian grid, a special particle ray-tracing technique is employed. The ray-tracing algorithm takes advantage of the cell connectivity information provided by the mesh data to efficiently handle the intersection of the line segment traced by the particle with arbitrary triangulated complex boundary geometry. The most significant cause of inefficiency is testing for intersection between a ray and each triangle of a sizeable complex body. This is extremely slow and very costly when for example, the geometry is made up of approximately 70,000 triangles and 70,000 tracks have to be tested for intersection. A review on the root of ray tracing and ray traversal algorithms for a variety of commonly used data structures can be found in (Chang 2001 [71]). The first octree traversal algorithm applied to ray tracing was introduced by Glassner [72]. Wu et al. implement a cell-by-cell ray-tracing technique for particle movement in unstructured grids [73]. The method of the particle ray-tracing technique used in their work is very similar to previous work done by Kannenberg [74] which involve tracing a particle's trajectory to the nearest cell face then moving the particle to the intersecting position on the corresponding face where a further journey of the particle depends on the face condition. As in other DSMC codes [75], ray tracing is performed only for particles which leave the assigned cell or impact the solid boundary to minimize the efficiency reduction due to (brute-force) ray tracing all simulated particles in the simulation domain.

The ray-tracing scheme is used in the vicinity of the solid object surface where the region of the bounding box is treated as follows. During a single time step, a molecule cannot move more than one collision cell size along each dimension (a DSMC time constraint). Ray tracing is performed only for particles that leave their assigned cell and intersect the box bounding the solid object. If the particle doesn't reach the bounding box or crosses it before the end of the simulation time step, the task of particle tracking is to test whether the particle intersects the simulation domain boundary or not. If it hits the simulation domain bound-

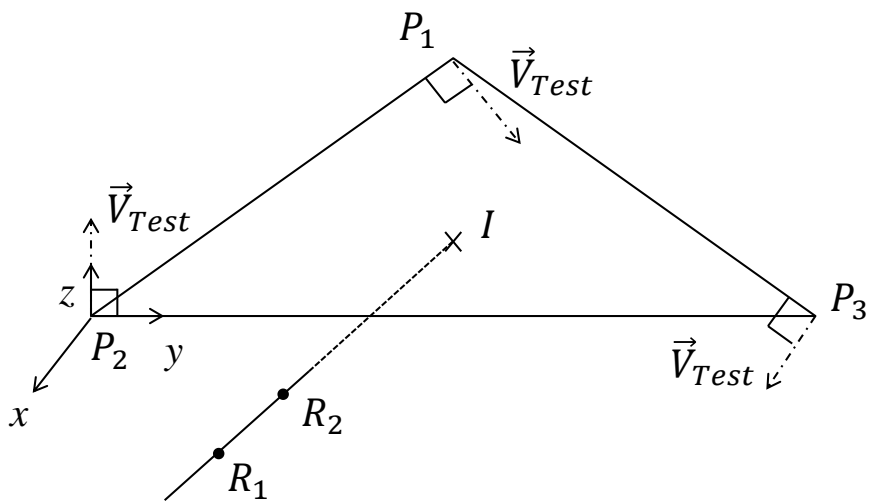
ary, then the particle is removed if an inflow/outflow boundary is encountered; otherwise, its interaction with the boundary is processed according to the specified wall boundary condition, its position is updated accordingly, and the task of its movement is routine. If not, then tentatively update the final position of the particle. If an intersection with the bounding box occurs (this includes the cases where the particle is within the bounding box, or the particle is outside and reaches the bounding box during its movement), a cell-by-cell particle tracking procedure is performed to determine whether the particle reaches a boundary surface triangle, stays in or leaves the current cell. This process is done in an extremely efficient way by taking advantage of the hierarchical octree-based data structure. The octree structure allows the most significant acceleration schemes for the ray tracing algorithm wherein grid cells are axis-aligned boxes whose edges are parallel to the basis vectors. Effective geometric tools used in computer graphics, including ray-box and ray-triangle intersection tests, as shown in Figure, are used within the ray tracing algorithm. If no ray-triangle intersection occurs, the particle's position is updated if the particle remains in the current cell; otherwise, ray-box intersection tests with all possible neighbor collision cells are performed to track the particle from the current cell to its nearest neighbor collision cell. The particle-tracking algorithm is then invoked again to move the particle over the remainder of the time step. If the particle intersects with one or more of the triangulated boundary surface mesh elements contained within a cell, the boundary triangle element with the minimum intersection distance is specified. The particle's position and velocity are determined according to the appropriate boundary condition, and the task of the particle movement is also a routine till using up the remaining of the whole time step. Figure represents the reflection of the particle after it intersects a boundary surface mesh element. If the particle leaves the bounding box before the end of the simulation time step, then an intersection test with the simulation domain boundary is performed. At the completion of the molecular movement phase, each particle is automatically stored within its last cell by the sort subroutine. Figure 3.8 summarizes the primary ray tracing steps in a flowchart.

### 3.4 Spatial and Temporal Adaptivity Scheme

A unique feature of the proposed algorithm is that it runs transient parallel Monte Carlo simulations simultaneously and independently on multicore CPUs. Most DSMC solvers are parallelized through decomposition of the physical domain into groups of cells that are distributed among the processors. The efficiency of such parallelization scheme may suffer due to the intensive communications between the processors and load imbalance among the processors. The spatial domain decomposition parallelization scheme is convenient for simulating low speed flows where a uniform Cartesian grid is used, and high-gradient flows where grid res-



(a)



(b)

Figure 3.6: Ray-Box (a) and Ray-Triangle (b) intersection tests

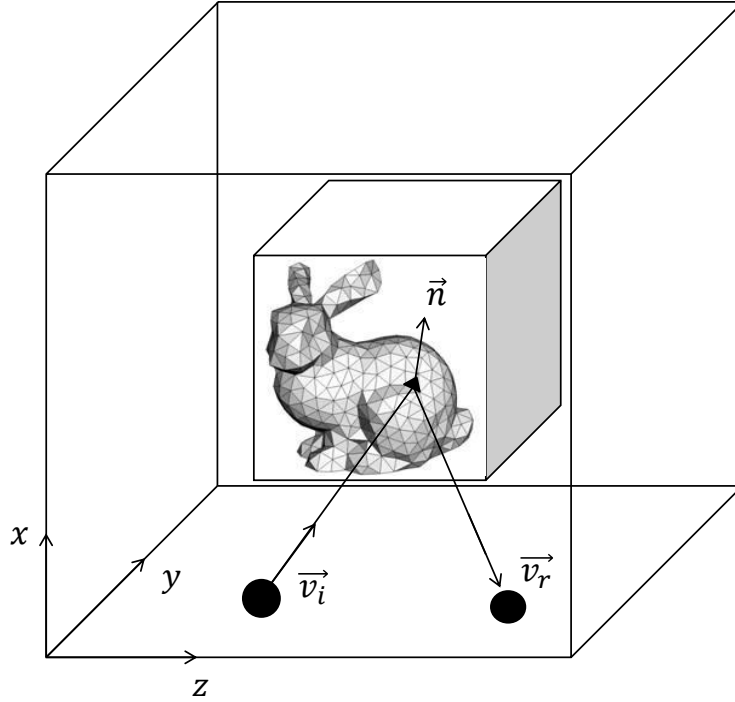


Figure 3.7: Reflection of a simulated molecule from the boundary triangle element

olution in both space and time is evoked once before steady state is reached. In contrast, the proposed parallelization scheme is more suitable for simulating highly unsteady rarefied flows. The length and time scales in such flows vary considerably over the domain. Thus, frequent spatio-temporal adaptation for variable resolution of the different flow regions is required. The upper diagram in Figure 3.9 shows a schematic representation of the proposed parallelization scheme using multiple threads. Each thread runs sequentially  $N$  realizations (each initiated from a unique random number generator seed) of the transient DSMC simulation consisting of  $M$  time steps. The realizations from different threads are averaged (by a SNIFFER algorithm) to compute the macroscopic properties distributions at flow sampling time steps. Due to the lack of communication between the threads when each is handling an independent realization, the parallelization efficiency is almost 100 %.

The proposed algorithm employs a novel spatio-temporal adaptivity scheme to simulate flows with length and time scales that vary considerably over the domain. Accurate prediction of these flows requires variable resolution of different flow regions. The spatio-temporal adaptivity scheme adjusts dynamically local grid spacing and time steps, and accurately resolve local flow features. Figure 3.10 shows the flow chart for the implemented DSMC method using the adaptive approach. The kinetic spatial scale, defined by the mean free path  $\lambda$ , and the temporal scale, defined by the mean collision time  $\tau_c = \lambda/\sqrt{2kT/m}$ ,



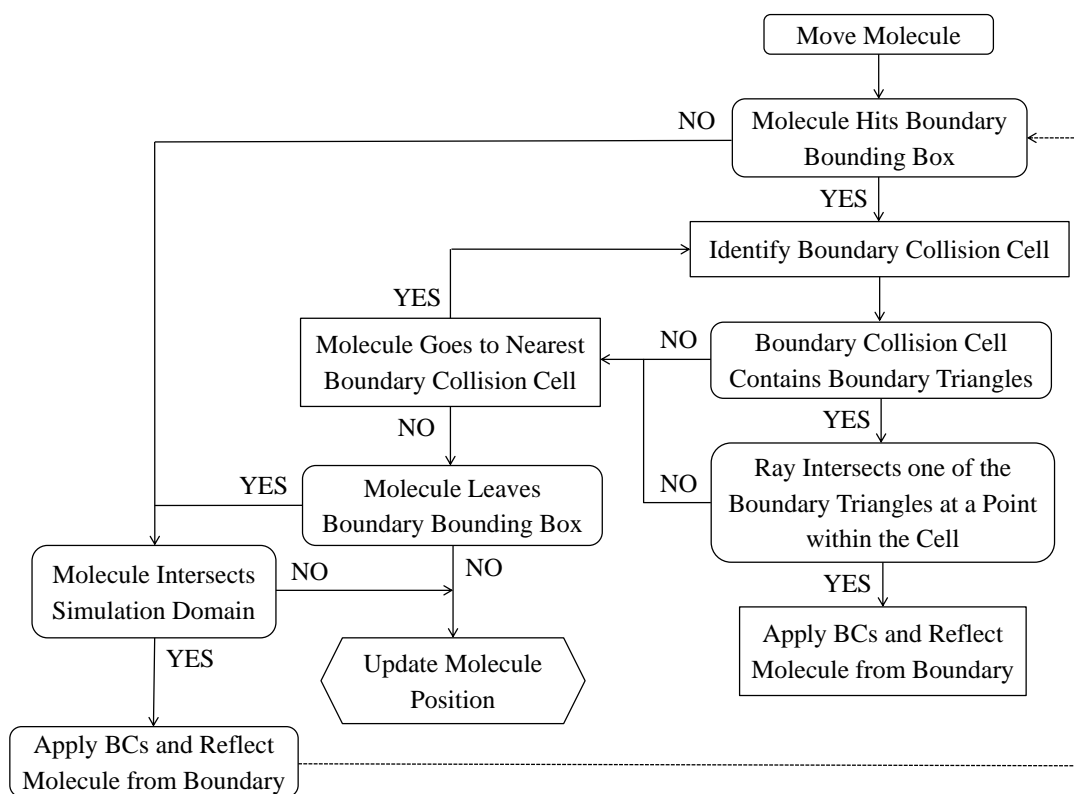


Figure 3.8: Ray-Tracing scheme

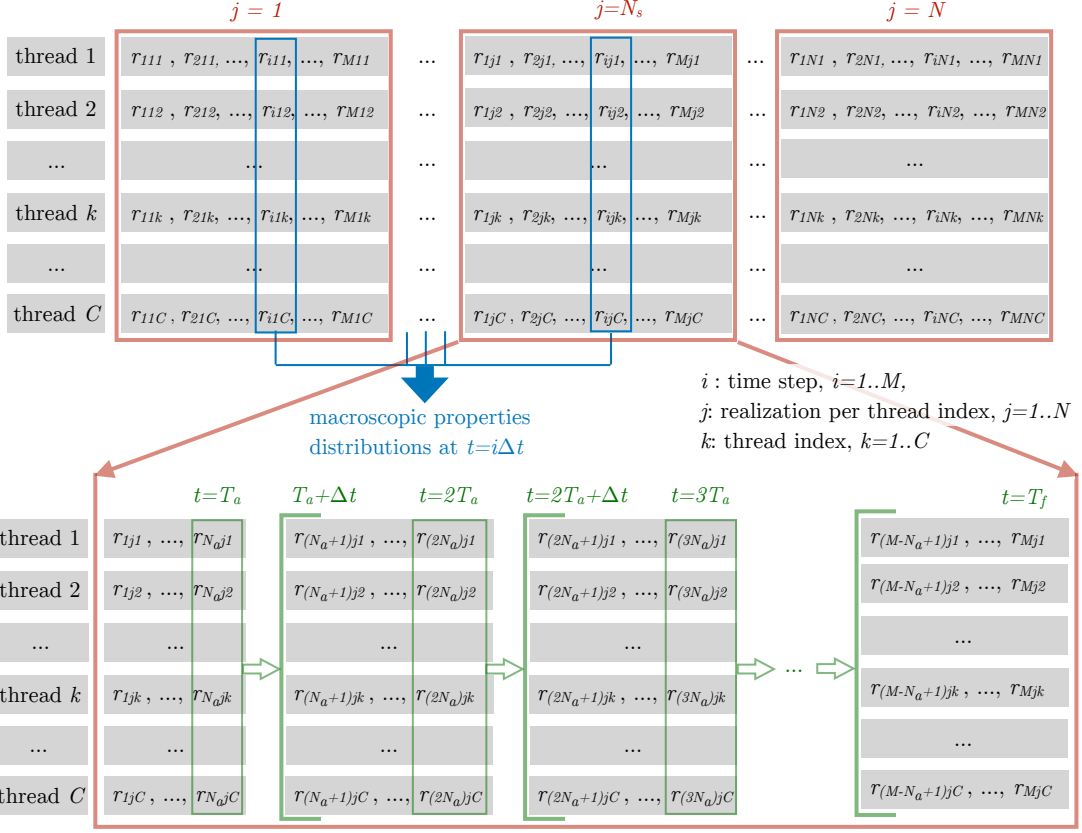


Figure 3.9: The schematic of running transient DSMC simulations on different threads. Top Diagram: Each thread runs sequentially  $N$  realizations of the transient simulation consisting of  $M$  time steps.  $N_s C$  realizations  $r_{ijk}$ ,  $j = 1..N_s$ ,  $k = 1..C$ , are averaged every output time interval,  $T_o = N_o \Delta t$ , to compute the macroscopic properties distributions at output time step  $N_o$ . The decision to include a new set of  $N_s C$  transient simulations (increment  $N$  by  $N_s$ ) is based on the relative statistical difference between macroscopic properties of the last  $NC$  realizations and the previous  $(N - N_s)C$  realizations. Note that  $N$  is an integer multiple of  $N_s$ .

Bottom Diagram: The spatio-temporal adaptivity is carried out for all threads every  $N_a = T_a / \Delta t$  time steps.  $C$  realizations  $r_{ijk}$ ,  $k = 1..C$  are averaged (by the sniffer) at  $t = T_a, 2T_a, \dots, T_f$  (corresponding to time steps  $i = N_a, 2N_a, \dots, M$ ) to estimate the macroscopic properties distributions needed for the spatio-temporal adaptivity criteria. These criteria will set the grid size and the associated temporal levels distribution for the time intervals  $[T_a + \Delta t, 2T_a]$ ,  $[2T_a + \Delta t, 3T_a]$ , ... While the sniffer carries out sampling of the microscopic properties and subsequently updates the distribution of cell sizes and temporal levels, the threads pause. Once the sniffer completes its task, the threads resume.

are calculated according to the binary elastic collision model used in the DSMC simulation. Several collision models, designed to reproduce the real flow macro-

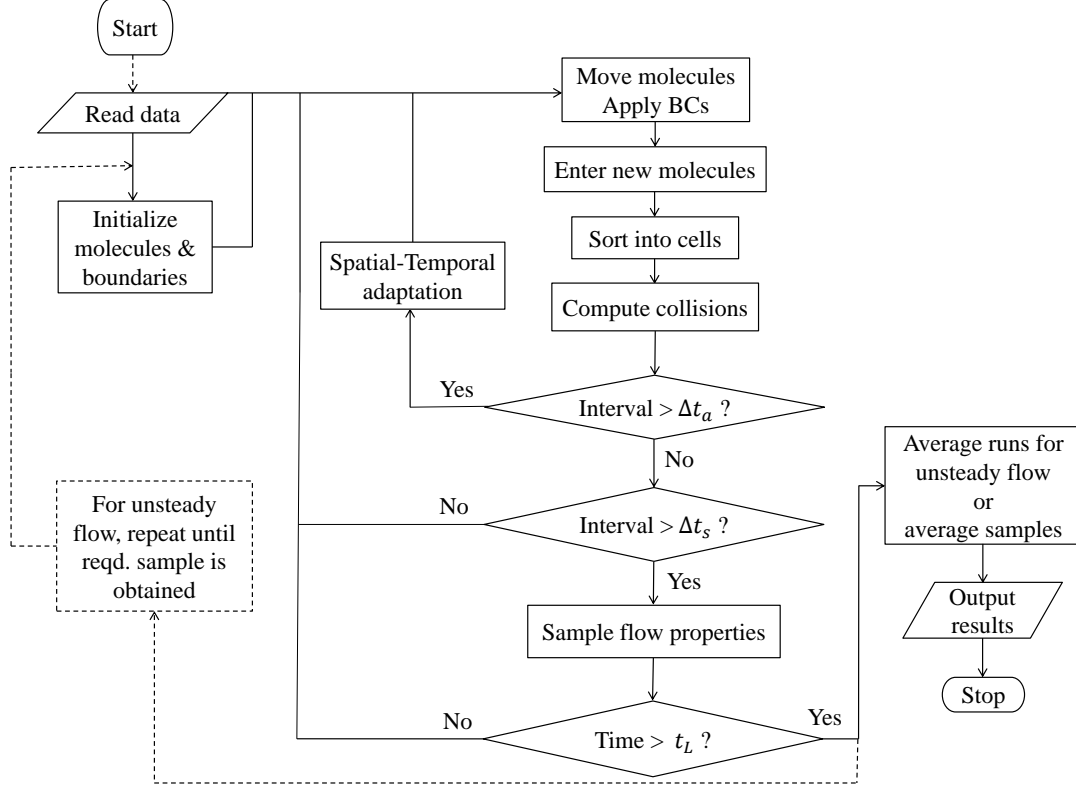


Figure 3.10: Simplified flow chart of the implemented DSMC algorithm.  $\Delta t_a$ : adaptive time;  $\Delta t_s$ : sampling time;  $t_L$ : Long time.

scopic behavior, were applied successfully to numerous DSMC simulations from micro/nano flows to hypersonic flows. These models include the inverse power law model, the hard sphere (HS) model, the variable hard sphere model (VHS), and the variable soft sphere (VSS) model [3]. In the VSS model, the mean free path is given as:

$$\lambda_{vss} = \frac{4\alpha(7-2\omega)(5-2\omega)}{5(\alpha+1)(\alpha+2)} \left(\frac{\mu}{n}\right) (2\pi m k_B T)^{-1/2} \quad (3.1)$$

$$\mu = \mu_{ref} \left(\frac{T}{T_{ref}}\right)^\omega$$

where  $m$  is the molecular mass,  $n = \rho/m$  is the number density,  $\mu_{ref}$  is the viscosity at reference temperature,  $k_B$  is Boltzmann constant,  $\alpha$  is the scattering parameter ( $\alpha = 1$  for VHS model), and  $T$  is the temperature. The viscosity index  $\omega$  is the power exponent of temperature in the viscosity law given by  $\omega = \frac{1}{2} \frac{\eta+3}{\eta+1}$ .  $\eta$  is the repulsive power exponent in the inverse power law model or the repulsion

point center model,  $F = K/r^\eta$ , where  $F$  is the force,  $K$  is constant, and  $r$  is the distance between molecules. Typical values of  $\eta$  are: 5, 7, 9, 11, 15, 21, 25, and  $\infty$ . For the hard sphere (HS) model one has  $\omega = 1/2$ , and  $\eta = \infty$ , for the variable hard sphere (VHS and also the inverse power law IPL) molecule with  $\omega = 0.75$ , one has  $\eta = 9$ , and for the Maxwellian molecule one has  $\omega = 1$ , and  $\eta = 5$ . The hard sphere model with  $\eta = \infty$  is the hardest molecule, and the Maxwellian molecule is the most soft among the molecular models under study [46]. In the HS model, the mean free path is expressed as:

$$\lambda_{\text{HS}} = \frac{1}{\sqrt{2}\pi d^2 n} \quad (3.2)$$

where  $d$  is the molecular diameter.

The mean collision time is related to the mean free path  $\lambda$  and the most probable velocity  $v_{mp}$  by

$$\tau_c = \frac{\lambda}{v_{mp}} = \lambda \times \frac{\sqrt{m}}{\sqrt{2k_B T}} \sim \frac{\lambda}{\sqrt{T}} \quad (3.3)$$

It can be seen from the above equations that the mean free path and the mean collision time are inversely proportional to the number density as:

$$\begin{aligned} \lambda &\sim \frac{T^{\alpha^*}}{n} \\ \tau_c &\sim \frac{T^{\alpha^*-0.5}}{n} \end{aligned} \quad (3.4)$$

where  $\alpha^* \in [0, \omega - 1/2]$ ;  $\alpha^* = 0$  for HS model.

Thus, for flow field having large non-uniform density and temperature variation, such as hypersonic flows over blunt bodies, will result in a significant variation of the mean free path and mean collision time in the flow field. The presence of large gradients of flow macroscopic parameters requires variable resolution of different flow regions. In a high-density region,  $\lambda$  and  $\tau_c$  would be small and small cell size and time step should be used in these regions; while in low-density areas, large cell size and time step would be used for accurate and efficient DSMC simulation. In addition to the number density, the temperature also affects the mean free path and the mean collision time. Thus, the process of regenerating the mesh and reassigning a variable time step should ideally be fully automated and consistent with the density and temperature distribution of the flow field.

Prior to the adaptation process, the DSMC simulation starts with a uniform mesh. As depicted in the bottom diagram of Figure 3.9, spatio-temporal adaptivity is carried out for all threads every spatio-temporal adaptivity time interval. This parallel process is frequently interrupted by the SNIFFER (serial) algorithm to efficiently compute the macroscopic properties from the average of the realizations from different threads and subsequently set the grid size and the associated

temporal levels distribution. The additional cost of the SNIFFER serial activity is very small and the measured parallelization efficiency of the parallelization scheme including the SNIFFER is more than 95 %.

Spatial adaptation of collision cells follows the conventional DSMC constraint on the collision cell size,  $\Delta x_c = \alpha_c \lambda$ , where  $\alpha_c$  is a user-defined collision cell size factor. The DSMC constraint on the minimum number of simulated molecules per collision cell,  $N_{min}$ , must be preserved in spatial adaptation process to disallow collision cells with too few molecules. It is concluded from previous DSMC studies [76] that it is necessary that the cell size is constrained to be less than one-third of the local mean-free-path,  $\alpha_c \in [1/4, 1/2]$ , and the number of simulated molecules per cell should exceed 20 ( $N_{min} > 20$ ) for slip flow and 10 ( $N_{min} > 10$ ) for transition flow. Spatial adaptation is done in two processes: refinement and coarsening. During the refinement process, the average of local mean free path,  $\lambda_{av}$ , and the minimum number of simulated molecules over all CPU cores is computed for each collision cell. Then, each collision cell is tested for spatial adaptivity by computing the nearest division integer  $n_d$ , given by:

$$n_d = \frac{\log\left(\frac{\Delta x_c}{\alpha_c \lambda_{av}}\right)}{\log(2)} \quad (3.5)$$

The collision cell is refined into  $(2^{n_d})^3$  new octants provided that  $n_d$  is greater than or equal to one and the minimum number of simulated molecules in a collision cell is greater than  $N_{min} \times (2^{n_d})^3$ . In the coarsening process, the average local mean free path over all CPU cores is computed in each parent cell of eight collision octants. Every parent cell with a size less than  $\alpha_c \lambda_{av}$  or contains a child with few numbers of simulated molecules will be coarsened. The entire new geometry octree data structure is then completed and simulated molecules are re-sorted into the new tree structure.

The temporal adaptation requires computing the desired time step,  $\Delta t_d$ , in every collision cell and updating the DSMC simulation flow time step. Usually, the desired time step is adapted to the minimum time between a specified fraction of the local mean collision time in each collision cell,  $\Delta t_1 = \alpha_1 \tau_c$ , and a specified fraction of the time needed for a molecule to travel a local collision cell size,  $\Delta t_2 = \alpha_2 \Delta x_c / v_{mp}$ , as follows:

$$\Delta t_d = \min\left(\alpha_1 \frac{\lambda_c}{v_{mp}}, \alpha_2 \frac{\Delta x_c}{v_{mp}}\right) \quad (3.6)$$

where  $\alpha_1$  and  $\alpha_2$  should be smaller than 1/2 in DSMC simulations [76],  $\alpha_1, \alpha_2 \in [1/3, 1/2]$ . The DSMC simulation flow time step is updated according to a user-defined criterion. One of the criteria is the one specified by Bird in 2007 [67] in which the flow time step is advanced in steps equal to the smallest value of the desired time step among all collision cells. In the present study, different

temporal levels are considered in the computational domain where the number of temporal levels,  $N_l$ , is computed from the minimum,  $\Delta t_{dmin}$ , and maximum,  $\Delta t_{dmax}$ , desired time steps as follows:

$$\frac{\Delta t_{dmax}}{\Delta t_{dmin}} = 2^{N_l} \quad (3.7)$$

Each temporal level  $l$ ,  $l = 0..N_l - 1$ , is characterized by  $2^l$  time steps. Collision cells are assigned to different temporal levels such that  $\Delta t_l = \frac{\Delta t_{dmax}}{2^l} \leq \Delta t_{dc}$ , where  $\Delta t_l$  is the time step in temporal level  $l$  and  $\Delta t_{dc}$  is the desired time step in collision cell  $c$ . After grouping collision cells into different temporal levels, only levels with a minimum of 10% collision cells are considered in temporal adaptation. The DSMC simulation flow time step is advanced in steps equal to the average of the desired time steps of all collision cells in the first temporal level. The simulation then proceeds to iterate over the different temporal levels, in descending order of the time step size, where within each loop, all cells sharing the same time step are handled. Figure 3.11 presents a flowchart of the implemented temporal adaptation algorithm. The use of different temporal levels allows better handling of the spatial dependence of the time step in the flow domain and decoupling of molecular motion and collision in the variable time step. The temporal adaptivity scheme results in spatial dependence of the time step based on the criteria expressed in equations 3.6 and 3.7. Employing a discrete set of temporal levels enable effective handling of the movement of molecules between cells of different time steps. The proposed temporal adaptation scheme considers equal time steps, the time step in a temporal level  $l$ , for the molecular movement and collision attributes (Move Molecules ( $m$ ,  $\Delta t_l$ ) and Collide Molecules ( $l$ ,  $\Delta t_l$ ) as shown in Figure 3.11). Multiple collisions are carried out at the end of each time step  $\Delta t_l$  by computing to the nearest integer the ratio of the assigned temporal level time step of a cell to its desired time step. The advantage of using different temporal levels compared to other local time stepping techniques [32, 33] is to better handle the spatial dependence of the time step in the flow domain and the decoupling of molecular motion and collision in the variable time step. The temporal adaptivity scheme permits an effective treatment for the movement of molecules that may go to a cell with a different temporal level than the one of their corresponding cell. Besides, it handles molecular collisions in each collision cell while preserving the number of possible collision pairs based. A schematic of the temporal adaptation procedure for a two-temporal levels case is presented in Figure 3.12. For the sake of clarity, the large cells correspond to the first temporal level with a time step  $\Delta t$ , and the small ones correspond to the second temporal level with a time step  $\Delta t/2$ . We follow the loop over temporal levels as depicted in Figure 3.11. During the loop over temporal level  $l = 0$ , simulated molecules within cells of temporal levels  $l = 0$ , and  $l = 1$  are allowed to move (sub-figures (a) and (b)). The molecules are sorted at the end of the move step, and collisions

are performed within cells of temporal level  $l = 0$  (sub-figure (c)). Before ending the loop over temporal level  $l = 0$ , simulated molecules moved to level  $l = 1$  (simulated molecule 2) are held stationary during the loop over the temporal level  $l = 1$ , and simulated molecules moved from level  $l = 1$  to  $l = 0$  (simulated molecule 5, 6, and 7) are reset to their initial positions (sub-figures (c) and (d)). The loop over temporal level  $l = 1$  considers an inner loop over two time steps of  $\Delta t/2$ . The same criteria is followed. Sub-figures ((d) and (e)) and sub-figures ((g) and (h)) correspond respectively to the move step within the nested loop over the first and second time steps within the loop over temporal level  $l = 1$ . Sub-figure (f) and sub-figure (i) correspond respectively to the sort/collision step within the nested loop over the first and second time steps within the loop over temporal level  $l = 1$ . Sub-figure (j) corresponds to the end of the temporal adaptation procedure. At the end of the temporal adaptation algorithm, a global sort algorithm re-sorts all molecules by their position coordinates into the appropriate collision cells. The simulation is then resumed using the new mesh and the updated temporal levels.

```

Temporal adaptation algorithm:
If (inflow/outflow boundaries exist) then
    ❖ Generate/Move molecules at flow boundaries.
    ❖ Assign FLAG to molecules entering the domain. These molecules
      don't move while iterating over temporal levels.
End if
Do  $l=0, N_l-1$ 
    Do  $j=1, 2^l$ 
        ❖ Move Molecules( $m, \Delta t_l$ ): move molecules in cells in
          temporal level  $m \geq l$  with a time step  $\Delta t_l$ .
          • Ray-Trace molecule movement
          • Carry out molecule-surface interactions
          • Update molecular position
        ❖ Sort Molecules
        ❖ Collide Molecules( $l, \Delta t_l$ ): collide molecules in cells in
          temporal level  $l$ .
          • Multiple collisions
    End Do
    ❖ Assign FLAG '*' to molecules initially at level ' $l$ ' and moved to level
      ' $m > l$ '. These molecules don't move in the next iterations of the loop.
    ❖ Retain molecules initially at level ' $m > l$ ' to their initial position.
End Do
Global Sort (loop over all cells/molecules)

```

Figure 3.11: Temporal adaptation algorithm within the DSMC code.  $N_l$ : number of temporal levels in the domain;  $2^l$ : number of time steps in temporal level  $l$ ;  $\Delta t_l$ : time step in temporal level  $l$ .



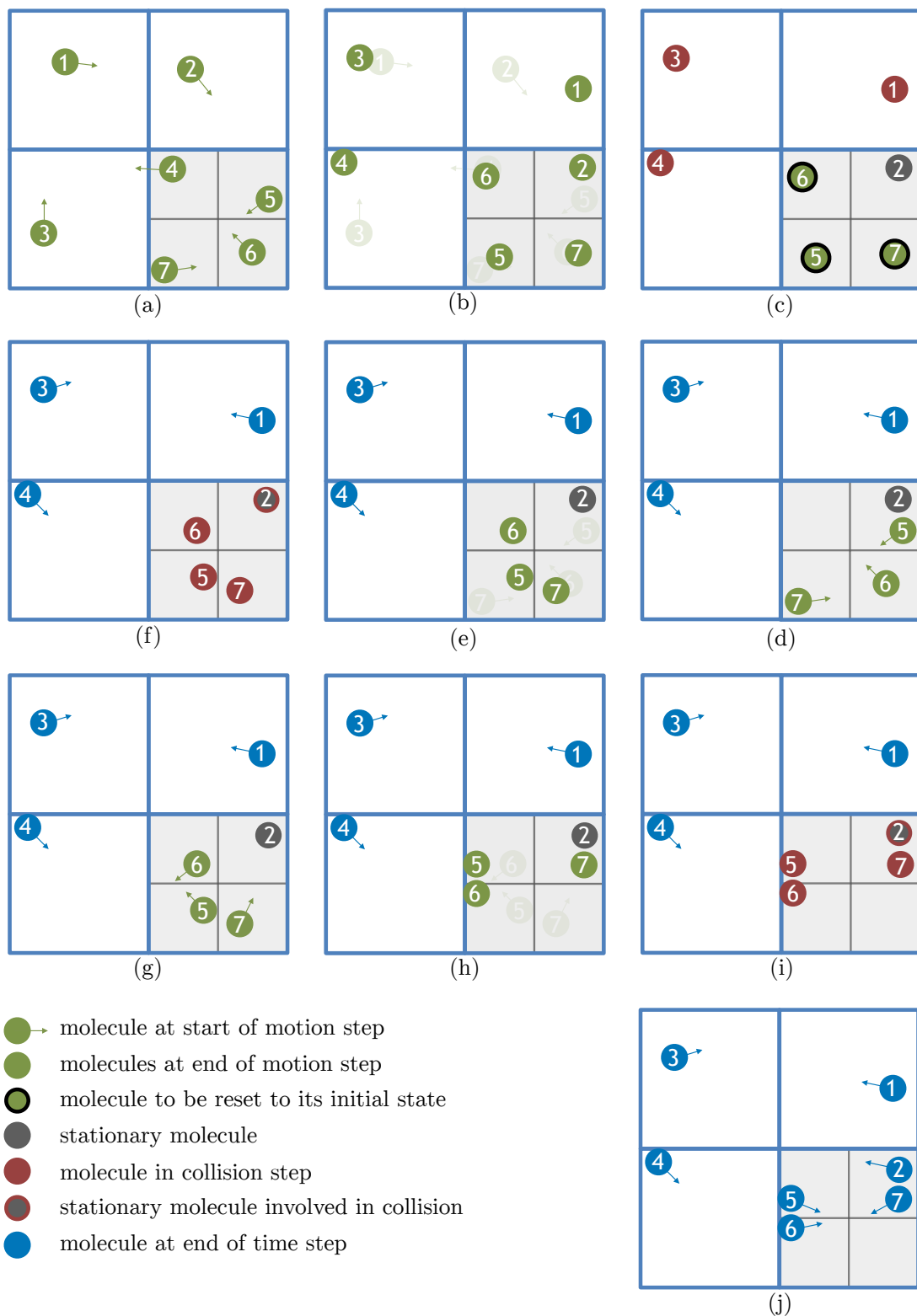


Figure 3.12: Schematic describing the temporal adaptation procedure with two temporal levels.

# Chapter 4

## Validation of The Proposed Algorithm Against Benchmark Simulations

Several time-dependent numerical simulations of rarefied gas flow in the slip and transition flow regimes are investigated to validate the proposed three-dimensional DSMC code. These simulations have been extensively applied in many applications of micro/nano-technologies. The simulations were carried out using Fortran in CentOS Linux 7 on 16 core Intel Xeon(R) CPU E5-2650 v2 running at 2.6 GHz.

### 4.1 Oscillatory Shear-Driven Couette Flow

In the following, we analyze oscillatory Couette flow which is one of the previous investigations that uses the unsteady direct simulation Monte Carlo method to analyze time-periodic rarefied gas flows. A flow of argon gas between two infinite parallel plates at a distance  $H$  apart is simulated such that the bottom plate is stationary and the top plate oscillates in a simple harmonic motion with a velocity  $U = U_0 \sin(\omega t)$  in the lateral direction. A schematic view of the problem is shown in Figure 4.1. The oscillatory Couette flow is characterized by the Knudsen  $\text{Kn}$ , Mach  $\text{Ma}$ , and Stokes  $\beta$  non-dimensional parameters. The Knudsen number is the ratio of the mean free path  $\lambda$  to the characteristic system length  $H$ ,  $\text{Kn} = \lambda/H$ . The Stokes number  $\beta$  represents the ratio of the diffusion and oscillation characteristic time scales, and it is defined as

$$\beta = \sqrt{\frac{\omega H^2}{\nu}} = \left( \frac{H^2/\nu}{1/\omega} \right)^{1/2}, \quad (4.1)$$

where  $\nu$  is the kinematic viscosity and  $\omega$  is the oscillation frequency. The Mach number is the ratio of the flow velocity to the local speed of sound  $a$ ,  $\text{Ma} = U_0/a$ , and can be used to evaluate the effects of compressibility.

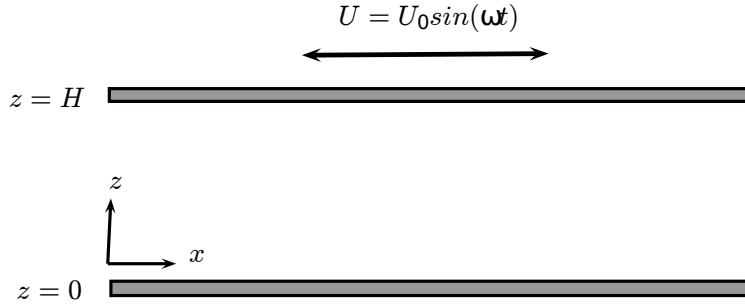


Figure 4.1: Schematic view of oscillatory Couette flow

Simulation parameters are selected based on previous work done by Park et al. [77]. The gas medium is initially at rest under standard atmospheric conditions ( $P_0 = 101325\text{Pa}$  and  $T_0 = 273\text{K}$ ). The two plates are maintained at the same temperature  $T_w = 273\text{K}$ . The oscillation amplitude of the upper plate is kept constant at  $U_0 = 100\text{m/s}$  resulting in a low Mach number,  $\text{Ma} = 0.3248$ , so that the compressibility effects are negligible. The characteristic system length and the oscillation frequency are adjusted to  $H = 0.625\mu\text{m}$ , and  $\omega = 8.096 \times 10^8\text{rad/s}$ , respectively. This results in a simulation of  $\text{Kn} = 0.1$  and  $\beta = 5$ . We utilize the Hard Sphere (HS) collision model for molecular collisions and the No-Time-Counter (NTC) scheme for collision pair selection. The horizontal plates are assumed to be fully accommodating and periodic boundary conditions are employed on the side walls at the  $x - z$  and  $y - z$  planes. For this unsteady flow, ensemble averaging over 5000 independent unsteady realizations is performed. Each unsteady realization simulates the flow over a time span long enough for the flow to reach the quasi-stationary behaviour. The macroscopic properties are computed every  $T/4$ , where  $T = 2\pi/\omega$  is the period of oscillation.

DSMC simulations consider combinations of cell size, time step, and a number of simulated molecules per cell constraints to produce physically realistic and accurate solutions. The cell size constraint states that a characteristic cell dimension is chosen such that it is some fraction of the mean free path (typically  $1/3$  to  $1/2 \lambda$ ). In our simulation, cell size is chosen as  $1/3\lambda$ , each containing two sub-cells along each direction where nearby particles located within the same computational sub-cell collide statistically. About 100 simulated molecules are randomly distributed in each cell. The time step, ( $\Delta t$ ), is chosen smaller than the mean collision time ( $\sim \lambda/v_{mps}$ ) where  $v_{mps}$  is the most probable velocity, the time period of oscillations  $\simeq 2\pi/\omega$ , and the viscous diffusion time scale  $\sim H^2/\nu$  where  $\nu$  is the kinematic viscosity. Moreover,  $\Delta t$  is chosen such that a simulated molecule resides in the same cell for at least a few time steps to allow it to interact with other molecules. In our simulation, a typical molecule is set to move about one-third of the cell dimension in one simulation time step. This ensures that its information can be adequately distributed throughout the computation domain.

The analytical solution to this problem is obtained by solving the Navier-Stokes equations subject to the appropriate slip-flow boundary conditions. The flow is isothermal, and simulation parameters are chosen such that compressibility and viscous heating effects are negligible (low Ma). Besides, the small amplitude lateral motion of the plate does not generate any stream-wise pressure gradients. In this limit, the momentum equation in dimensionless form reduces to the following:

$$\frac{\partial u}{\partial t} = \left( \frac{\partial^2 u}{\partial z^2} \right) \quad (4.2)$$

For isothermal flows with tangential momentum accommodation coefficient ( $\sigma_v = 1$ ) in slip flow regime, first-order velocity slip boundary conditions are applicable and given by:

$$\begin{cases} u(0, t) - C_1 \text{Kn} \frac{\partial u(0, t)}{\partial z} = 0 \\ u(1, t) + C_1 \text{Kn} \frac{\partial u(1, t)}{\partial z} = \sin(\omega t) \end{cases} \quad (4.3)$$

where  $C_1$  is the modified slip coefficient given by:

$$C_1 = 1.298 + 0.718 \tan^{-1}(-1.175 \text{Kn}^{0.586}) \quad (4.4)$$

Indeed, the analytical solution of the above equations is that of a one-dimensional boundary-value problem of heat conduction with non-homogeneous boundary conditions of the first kind. This problem is solved with the integral transform (Fourier transform) technique [78]. It follows that the general solution of the oscillatory flow problem considered is:

$$u(z, t) = \sum_{n=1}^{\infty} e^{-\alpha \lambda_n^2 t} K(\lambda_n, z) \left[ \int_0^t e^{\alpha \lambda_n^2 t'} A(\lambda_n, t') dt' \right] \quad (4.5)$$

where

$$A(\lambda_n, t') = \frac{K(\lambda_n, 1)}{C_1 \text{Kn}} \quad (4.6)$$

and  $K(\lambda_n, z)$  is the Kernel corresponding to non-homogeneous boundary conditions of the first kind given by:

$$K(\lambda_n, z) = \sqrt{2} \frac{\lambda_n \cos(\lambda_n z) + H_1 \sin(\lambda_n z)}{\left[ (\lambda_n^2 + H_1^2) \left( 1 + \frac{H_2}{\lambda_n^2 + H_2^2} \right) + H_1 \right]^{1/2}} \quad (4.7)$$

and the eigenvalues,  $\lambda_n$ , are positive roots of the equation:

$$\tan \lambda = \frac{\lambda(H_1 + H_2)}{\lambda^2 - H_1 H_2} \quad (4.8)$$

where  $H_1 = H_2 = \frac{1}{C_1 \text{Kn}}$ .

In the work done by Park et al. [77], a semi-analytical/empirical model that

is applicable for quasi-steady flows ( $\beta \leq 0.25$ ) in the entire Knudsen regime, and for any Stokes number flow in the slip flow regime ( $\text{Kn} \leq 0.1$ ), is developed. A velocity response of the form  $u(z, t) = \Im\{U(z)\exp(j\omega t)\}$  is expected, where the symbol  $\Im$  denotes the imaginary part of a complex expression. Thus, in the quasi-steady limit, The time-dependent velocity distribution is obtained as follows [77]:

$$u(z, t) = \Im \left[ \left( U_0 \frac{\sinh(\sqrt{j}\beta Z) + \sqrt{j}\beta C_1 \text{Kn} \cosh(\sqrt{j}\beta Z)}{(1 + j\beta^2 C_1^2 \text{Kn}^2) \sinh(\sqrt{j}\beta) + 2\sqrt{j}\beta C_1 \text{Kn} \cosh(\sqrt{j}\beta)} \right) \exp(j\omega t) \right], \quad (4.9)$$

where  $Z = z/L$ . The shear stress at the oscillating plate is given by:

$$\begin{aligned} \tau_{xz} &= \left. \frac{du(z, t)}{dz} \right|_{z=L} \\ &= \Im \left[ \left( \frac{U_0}{L} \sqrt{j}\beta \frac{\cosh(\sqrt{j}\beta) + \sqrt{j}\beta C_1 \text{Kn} \sinh(\sqrt{j}\beta)}{(1 + j\beta^2 C_1^2 \text{Kn}^2) \sinh(\sqrt{j}\beta) + 2\sqrt{j}\beta C_1 \text{Kn} \cosh(\sqrt{j}\beta)} \right) \exp(j\omega t) \right], \end{aligned} \quad (4.10)$$

DSMC results of the velocity distribution at different times of an oscillation period within the time-periodic state are shown in Figure 4.2. The results are in good agreement with the published data in [77], and with the analytical solutions (equations (4.5) and (4.9)). Departure of the analytical solution from that predicted by DSMC is expected due to the fact that it employs a first order slip boundary condition at solid walls. Comparisons between the shear stress obtained from DSMC simulations and the analytical solution are presented in Figure 4.3 at different times.

## 4.2 Impulsive Started Couette Flow

Transient behavior of argon gas between two parallel, diffusely reflecting plates, each at temperature  $T = 273\text{K}$  and transverse parallel velocity of  $U_0 = 100\text{m/s}$ , is studied. The characteristic length scale of the flow system is adjusted to  $H = 0.2976\mu\text{m}$  for the simulation of a flow at  $\text{Kn} = 0.21$ . Unsteady sampling of macroscopic properties and the viscous shear stress is performed at time  $t = 16.2\epsilon^{-1}$ , where  $\epsilon$  denotes the molecular collision frequency. The viscous shear stress  $\tau$ , derived using molecular gas dynamics [3], is defined as the negative of the pressure tensor with the static pressure subtracted from the normal components. It is written in tensor notation as:

$$\boldsymbol{\tau} \equiv \tau_{ij} = -(\overline{\rho v'_i v'_j} - \delta_{ij} p), \quad (4.11)$$

where  $\rho$  is the mass density, and  $\delta_{ij}$ , the Kronecker tensor, is defined as:

$$\delta_{ij} = 1(i = j), \delta_{ij} = 0(i \neq j), \quad (4.12)$$

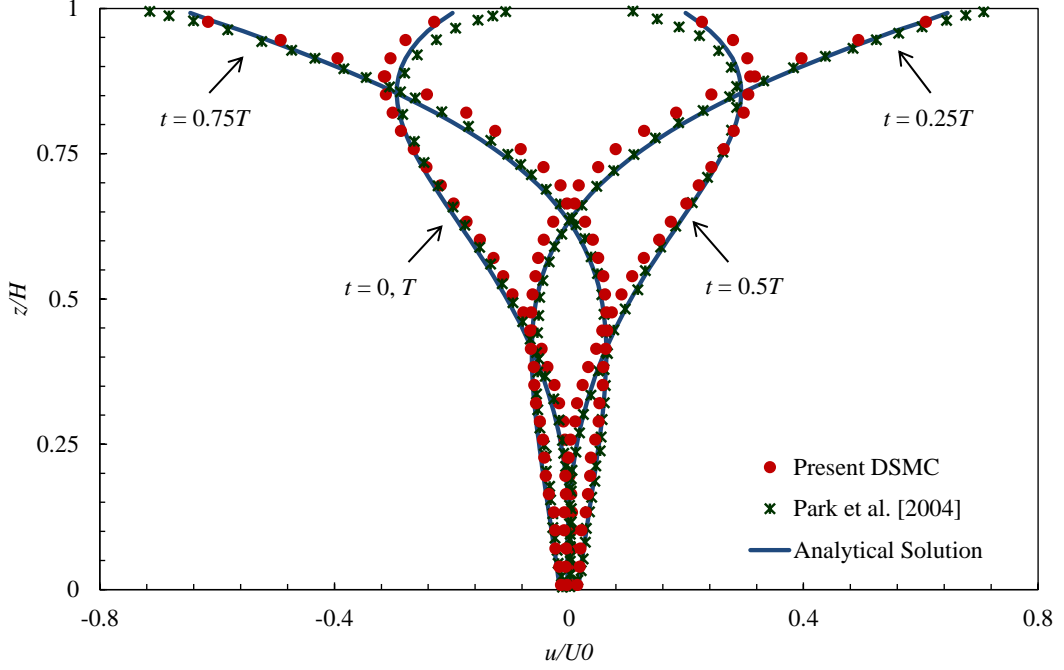


Figure 4.2: Normalized velocity profile for the shear-driven oscillatory Couette flow at  $\text{Kn} = 0.1$ ,  $\text{Ma} = 0.3248$ , and  $\beta = 5.0$ .

$v'_i$  and  $v'_j$  are the components of the velocity  $\mathbf{v}'$  of a molecule relative to the stream velocity, i.e., the thermal velocity. The mean or hydrostatic pressure  $p$ , is defined as the average of the three normal components of the pressure tensor  $p_{ij} = \rho \overline{v'_i v'_j}$ , that is:

$$p = \frac{1}{3} \rho (\overline{v'^2_x} + \overline{v'^2_y} + \overline{v'^2_z}) = \frac{1}{3} \rho \overline{v'^2} \quad (4.13)$$

The heat flux is defined as:

$$\mathbf{q} = \frac{1}{2} \rho \overline{v'^2 \mathbf{v}'} + n \overline{\epsilon_{int} \mathbf{v}'} \quad (4.14)$$

where,  $\epsilon_{int}$  is the internal (vibrational or rotational) energy related to one molecule.

To examine the accuracy of our numerical DSMC simulations of the velocity and stress fields, a comparison of the DSMC data with previous ones by Hadjiconstantinou [79], and with the exact solution of Navier-Stokes equation for second slip flow is shown in Figure 4.4. Comparison of the results show fairly good agreement.

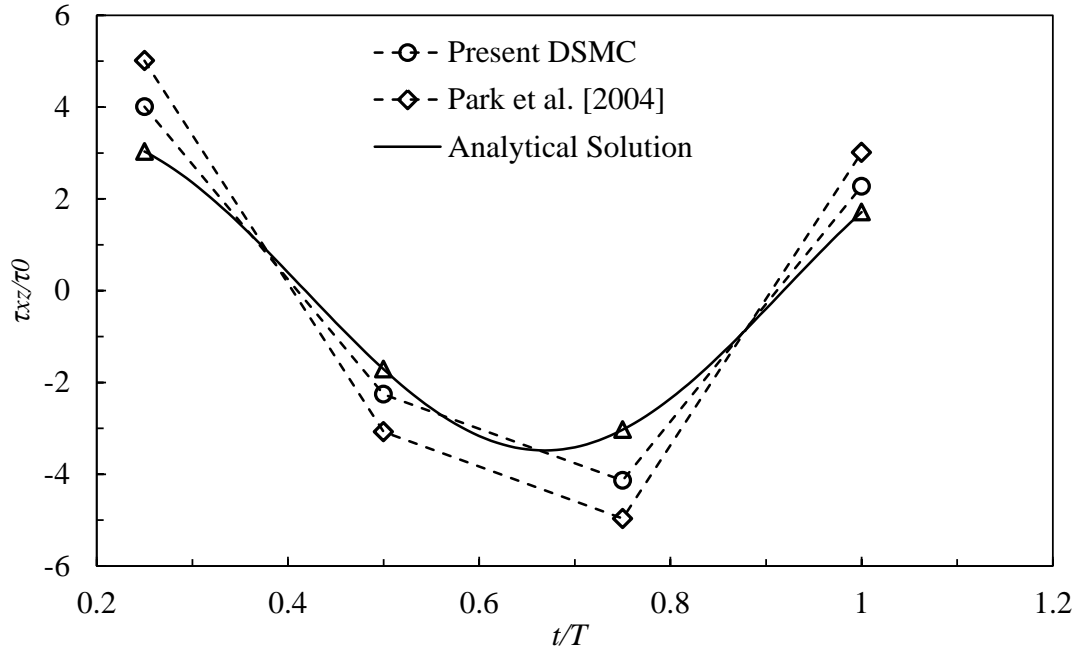


Figure 4.3: Normalized wall shear stress for the shear-driven oscillatory Couette flow at different times.  $\text{Kn} = 0.1$ ,  $\beta = 5.0$ , and  $\tau_0 = U_0/H$ .

### 4.3 Thermal Couette Flow

In the following Couette-flow test, Argon gas, initially at rest and at temperature  $T = 273\text{K}$ , is contained in the region between two stationary plates, which held at temperatures  $173\text{K}$  and  $373\text{K}$ , respectively. The system is then allowed to reach a steady state. Figure 4.5 represents steady-state temperature profiles across the channel at different Knudsen numbers ( $\text{Kn} = 0.1, 1.0$ ). A very good agreement is obtained with previous results by Olson et al. [27].

### 4.4 Poiseuille Flow

The pressure-driven Poiseuille gas flow in a rectangular microchannel between two parallel plates is considered. The channel axis is along the  $x$  direction. The plates, of length  $L$  along the flow direction ( $x$ ), are separated by a distance  $H$ , where  $L/H \gg 1$ . The flow is two dimensional in the  $x - z$  plane; the channel dimension,  $W$ , along the  $y$  direction is much larger than  $H$ ;  $W \gg H$ . Since the flow is two dimensional in the  $x - z$  plane, the computational domain consists of one grid cell in the  $y$  direction. The following boundary conditions were imposed: (i) solid-walls are assumed to be perfectly accommodating, (ii) periodic boundary conditions are enforced at the  $x - z$  planes on either size of the domain, and (iii) the Maxwellian reservoir method is used for the inflow/outflow boundary condi-

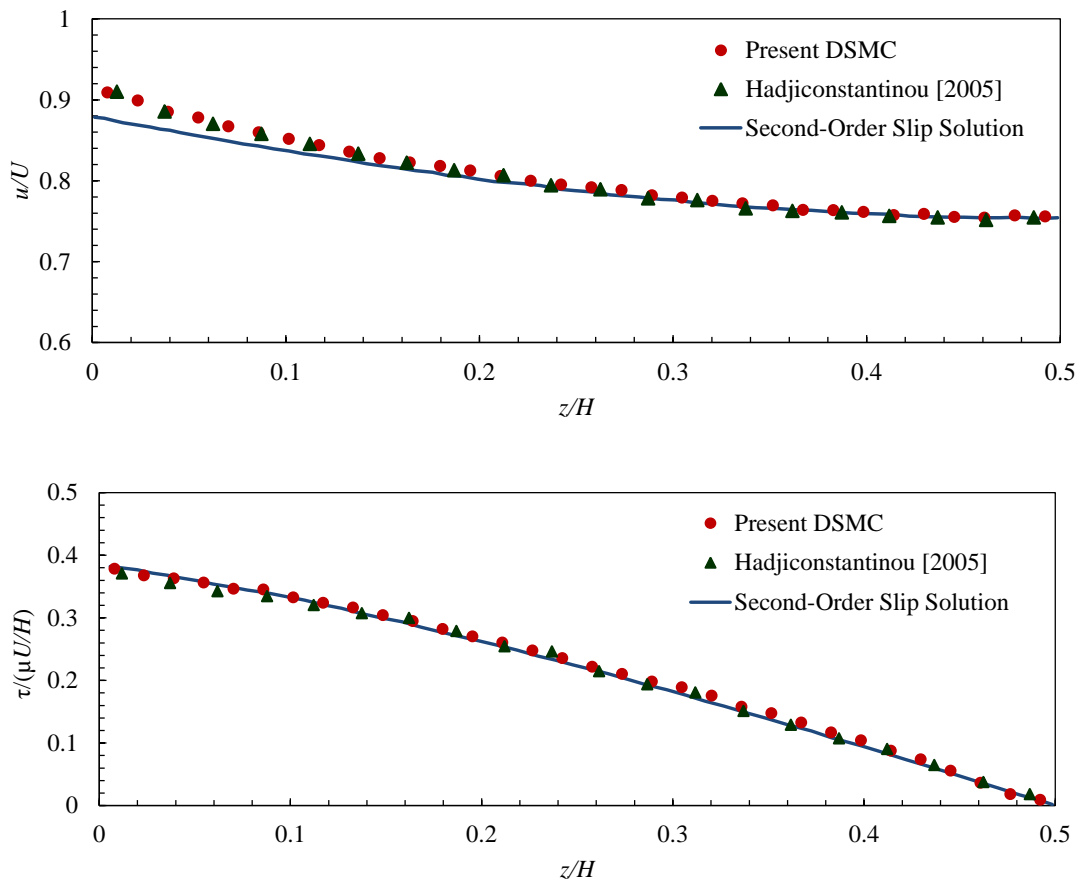


Figure 4.4: Velocity (top) and Stress (bottom) fields for the impulsive start Couette flow for  $\text{Kn} = 0.21$  at times  $16.2\epsilon^{-1}$ .



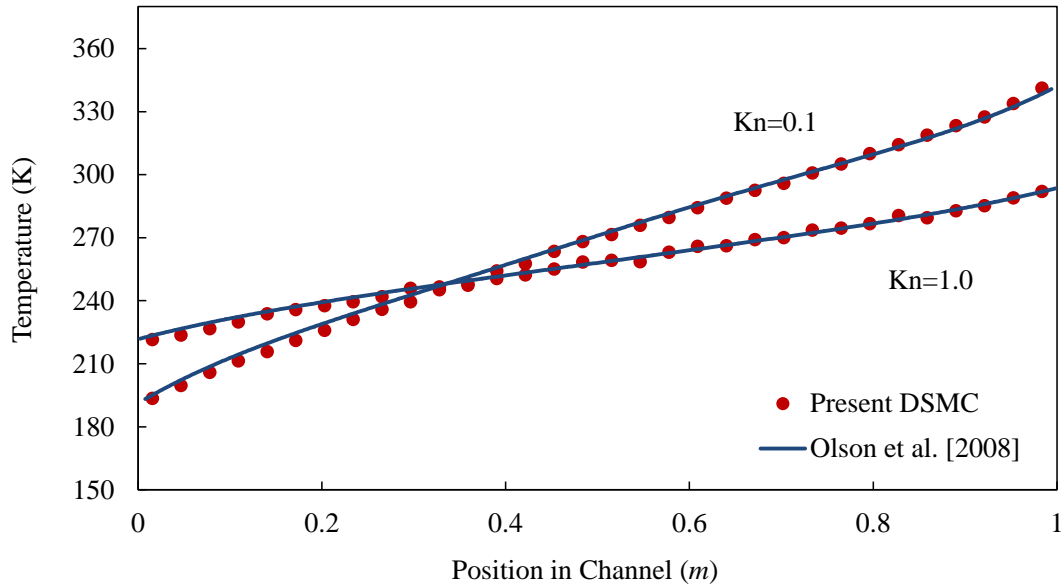


Figure 4.5: Temperature profile in a thermal Couette flow problem

tions at the inlet and outlet of the channel. The simulated fluid is argon gas (VHS gas) initially at ambient conditions. The channel length is  $L = 5 \mu m$ , the channel aspect ratio is  $L/H = 5$ , the inlet flow stream temperature is  $T_{in} = 300$  K, the inlet to exit pressure ratio,  $P_{in}/P_e = 3$ , and the inlet pressure is  $P_{in} = 160.839$  kPa. To assess the spatial adaptivity scheme in our DSMC algorithm, this flow is simulated with and without dynamic grid adaptation. For the non-adaptive simulation, the grid spacing was chosen to be  $\Delta x \sim 1/5\lambda_0$ , where  $\lambda_0$  is the mean free path at the initial conditions. For the adaptive simulation, the grid spacing is computed as  $\Delta x \sim \lambda/3$ , where at each adaptation time interval,  $\lambda$  for each cell is computed using Eq. (3.1), where  $T$  and  $n$  for the cell are averaged over all the cores. Figure 4.6 shows the steady state centreline pressure distribution using the proposed DSMC algorithm with and without grid adaptation. Our results are in good agreement with the simulated pressure profile by Wu et al. [80], with the pressure profile predicted using the adaptive scheme closer to that predicted by Wu et al. [80]. The figure also shows the flow field refined mesh upon spatial adaptation where the size of an inlet collision cell is adapted to half the size of a collision cell at the exit;  $\lambda_{in}/\lambda_e \sim P_{in}/P_e$ . With  $P_{in}/P_e = 3$  and since the octree divides a cell by integer powers of 2,  $\lambda_{in}/\lambda_e$  was set to  $1/2$ .

The transient solution and spatial adaptivity sampling intervals are presented in Table 4.1, among other parameters. The table also shows that at the end of the simulation, the number of cells in the adaptive simulation increased from 13056 to 48084, which is comparable to that used in the non-adaptive case. The computational cost of the adaptive simulation is, however, less by a factor of 7.7.

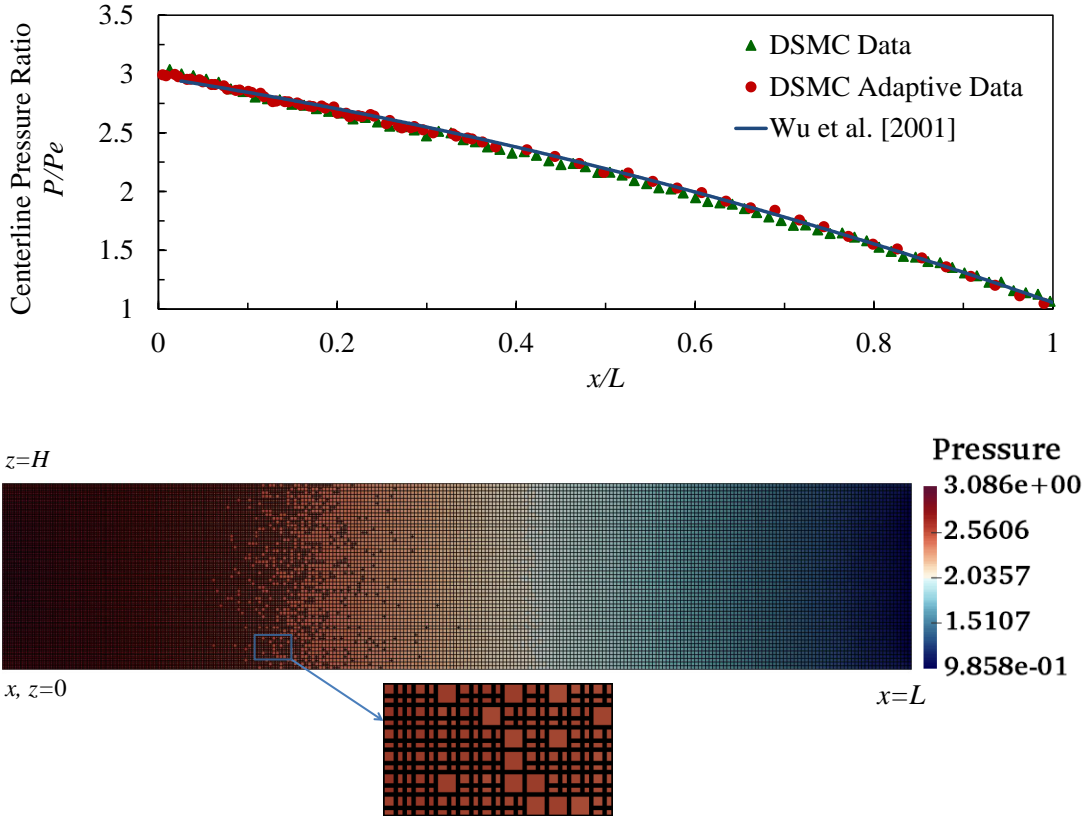


Figure 4.6: Centreline pressure distribution (top figure) and flow field refined mesh for a microchannel in the slip flow regime.

Table 4.1: Comparison of elapsed time per DSMC realization for non-adaptive and adaptive Poiseuille flow.

	<i>Non-Adaptive</i>	<i>Adaptive</i>
Time step	14.48 <i>ps</i>	28.96 <i>ps</i>
Sampling time	55.3 <i>ns</i>	55.3 <i>ns</i>
Transient solution sampling interval	27.45 <i>ns</i>	27.45 <i>ns</i>
Spatial adaptivity sampling interval		18.299 <i>ns</i>
Total no. of time steps	3800	1900
Total Time per Realization per core	45910.5 <i>s</i>	5931.76 <i>s</i>
Initial no. of Cells	52224	13056
Final no. of Cells	52224	48084

## 4.5 Slider Bearing Problem

Figure 4.7 shows a typical three-dimensional flat slider gas bearing configuration. A channel is formed by a moving horizontal bottom surface and a stationary slightly inclined surface whose length is  $L = 5 \mu\text{m}$  and height at the trailing edge is  $H_o = 50 \text{ nm}$ . The simulation used argon hard sphere particles initially at ambient conditions. Thus, the Knudsen number at the exit is  $\text{Kn}_o = \lambda/H_o = 1.25$ . The lower surface moving at a speed of  $U = 25 \text{ m/s}$  and the bearing number is  $\Lambda = 61.6$  (defined as  $\Lambda = 6\mu UL/p_a H_o^2$ ;  $p_a$  is the ambient pressure and  $\mu$  is the viscosity of the gas). As for the boundary condition, the solid-walls are assumed to be perfectly accommodating, periodic boundary conditions are enforced at the  $x-z$  planes on either side of the 3D domain, and the Maxwellian reservoir method is used for the inflow/outflow boundary conditions at the inlet and outlet. Our

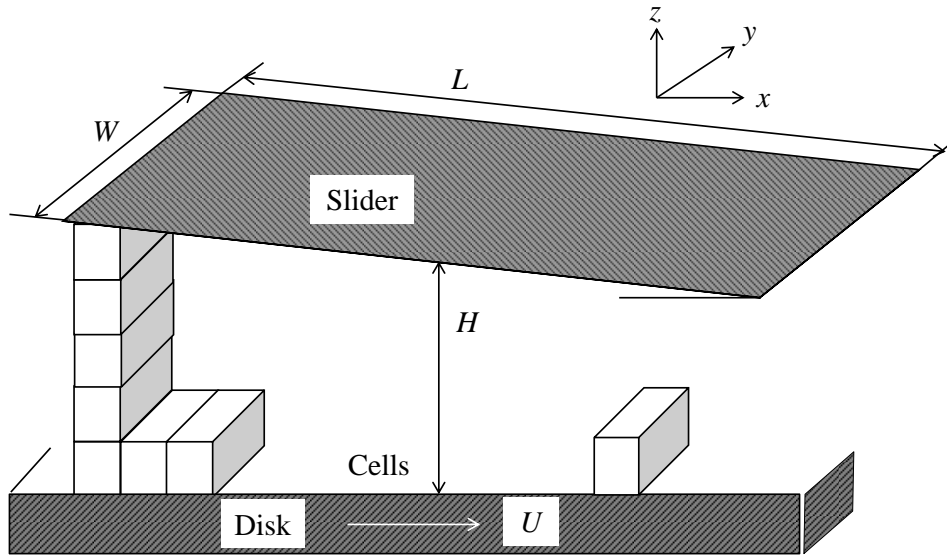


Figure 4.7. Schematic of the slider bearing geometry,  $L = 5 \mu\text{m}$ ,  $H_o = 50 \text{ nm}$ ,  $U = 25 \text{ m/s}$ .

DSMC results are in excellent agreement with the previously published DSMC solution by Garcia et al. [81].

Table 4.2 shows the simulation time per time step per core distributed among the particles movement, sorting and collision time steps. It can be observed that the molecular motion in the slider bearing flow takes the longest time. This is due to the cost of the ray-tracing operations using the hybrid mesh constructed near the inclined surface.

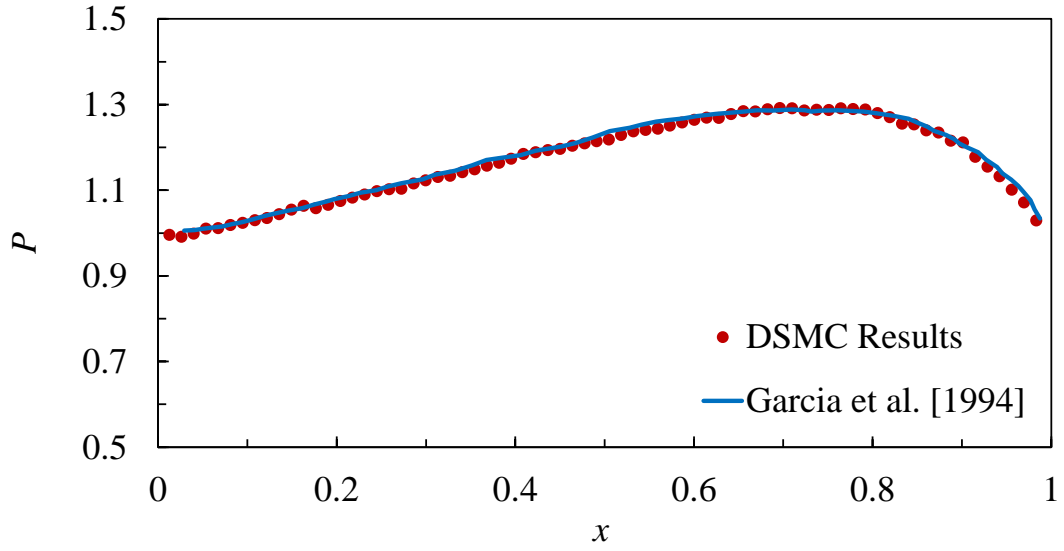


Figure 4.8. Slider bearing normalized pressure profile for  $Kn_o = 1.25$ ,  $\Lambda = 61.6$ ,  $Ma = 0.08$ .

Table 4.2. Elapsed time(s) of DSMC processes at flow sampling time step for benchmark simulations.

	<i>Non-Adaptive Poiseuille</i>	<i>Adaptive Poiseuille</i>	<i>Slider Bearing</i>
Move Molecules	3.269	0.917	4.240
Sort Molecules	2.165	0.729	0.084
Perform Collisions	0.354	0.215	0.019

## 4.6 Hypersonic Flows

To further demonstrate the utility of this code, we present two simulations of hypersonic flows with large density gradients. Two benchmark test cases are considered, 2D hypersonic flow over a cylinder and 3D hypersonic flow over a flat-nosed cylinder.

### 4.6.1 Hypersonic Flow Past a Flat-Nosed Cylinder

Hypersonic flow of argon gas at a temperature of 100K, a number density of  $1 \times 10^{21} m^{-3}$ , and a velocity of  $1000 m/s$  over a flat-nosed cylinder with a radius of  $0.01m$  is considered for the analysis. This corresponds to a stream Mach number  $Ma$  of 5.37 and a Knudsen number  $Kn$ , based on the diameter of the cylinder, of 0.0474. The simulated flow field is  $0.04m$  in the axial  $x$ -direction, and  $0.03m$  in the  $y$ - and  $z$ - directions. The total length of the cylinder is  $0.02m$  and the centre of its flat face is located at  $x = 0.02$ ,  $y = 0$ , and  $z = 0$ . The free stream flow boundary conditions are set at left ( $yz$ -plane), backward ( $y = 0.03$ ), and top boundary ( $z = 0.03$ ) of the computational domain using standard method and by providing free stream velocity and free stream temperature. Vacuum condition is imposed at the outlet boundary ( $x = 0.04$ ). Periodic boundary condition is imposed at the bottom ( $xy$ -plane), and the in front boundary ( $xz$ -plane) of the computational domain. Diffuse wall boundary condition is applied at the cylinder wall by incorporating a half-range Maxwellian distribution determined by the wall temperature and velocity. The temperature at the cylinder wall is considered as 300K.

This problem has been considered by Bird [3] as a 2D axisymmetric problem and a 3D quarter-section model with symmetry boundary conditions by Scanlon et al. [6]. To assess the spatial adaptivity scheme in our DSMC algorithm, this flow is simulated with and without dynamic grid adaptation. For the non-adaptive simulation, the grid spacing was chosen to be  $\Delta x \sim 1/2\lambda_0$ , where  $\lambda_0$  is the mean free path at the initial conditions. For the adaptive simulation, the initial grid size is  $\Delta x \sim \lambda_0$  and the grid spacing is computed as  $\Delta x \sim \lambda/2$ , where at each adaptation time interval,  $\lambda$  for each cell is computed using Eq.(3.1), where  $T$  and  $n$  for the cell are averaged over all the cores. Figure 4.9 shows the flow field refined mesh upon spatial adaptation where the size of the grid cells near the flat face of the cylinder is adapted to 1/4 the size of the initial grid. This illustrates that the high-density region near the flat face of the cylinder is clearly captured with much finer mesh distribution. The transient solution and spatial adaptivity sampling intervals are presented in Table 4.3, among other parameters. The table also shows that at the end of the simulation, the number of cells in the adaptive simulation increased from 18,432 to 136,893 which is comparable to that used in the non-adaptive case. The computational cost of the adaptive simulation is, however, less by a factor of 4.3. Besides, the percentage of time

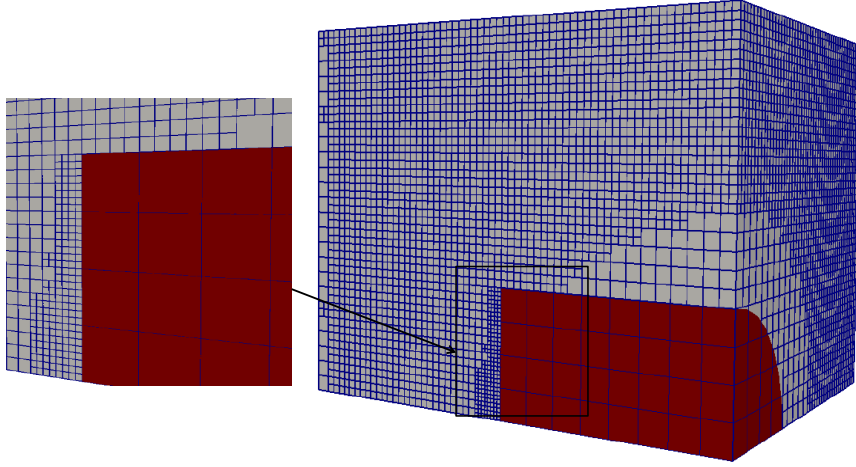


Figure 4.9. Flow Field Refined Mesh for a Flat-Nosed Cylinder.

spent by the sniffer algorithm for 15 different CPUs is less than 2% of the total time per realization per core. This reflects the high performance and efficiency of the implemented parallelization method. The temperature and density contours obtained are plotted over the results from Bird’s DSMC2A code [3], and that from `dsmcFoam` by Scanlon et al. [6] for comparison as shown in figures 4.10, 4.11. The contours show very good agreement.

Table 4.3. Comparison of elapsed time per DSMC realization for non-adaptive and adaptive hypersonic flow past a flat-nosed cylinder.

	<i>Non-Adaptive</i>	<i>Adaptive</i>
Time step	1.55 $\mu s$	1.55 $\mu s$
Sampling time	0.98 $ms$	0.98 $ms$
Transient solution sampling interval	0.196 $ms$	0.196 $ms$
Spatial adaptivity sampling interval		0.244 $ms$
Total no. of time steps	633	633
Total Time per Realization per core	102440.0 $s$	23719.0 $s$
Percentage of Time Spent by Sniffer	0.18%	1.1%
Initial no. of Cells	147456	18432
Final no. of Cells	147456	136893

### 4.6.2 Hypersonic Flow Over a Cylinder

Mach-10 hypersonic cylinder flow is a well-known 2D-benchmark problem [67, 7, 82]. Figure 4.12 shows the sketch of a Mach-10 (2634.1m/s) flow of argon gas at a temperature of 200K and a number density of  $4.274 \times 10^{20} m^{-3}$  past a circular cylinder with a fully diffusive surface at a temperature of 500K. The

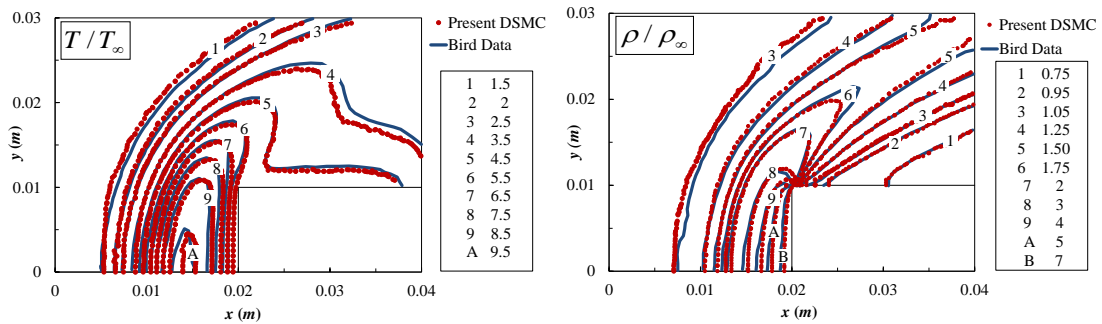


Figure 4.10. Temperature and density contours for hypersonic flow past a flat-nosed cylinder. A comparison of the results in this work with those computed in previous work done by Bird [3].

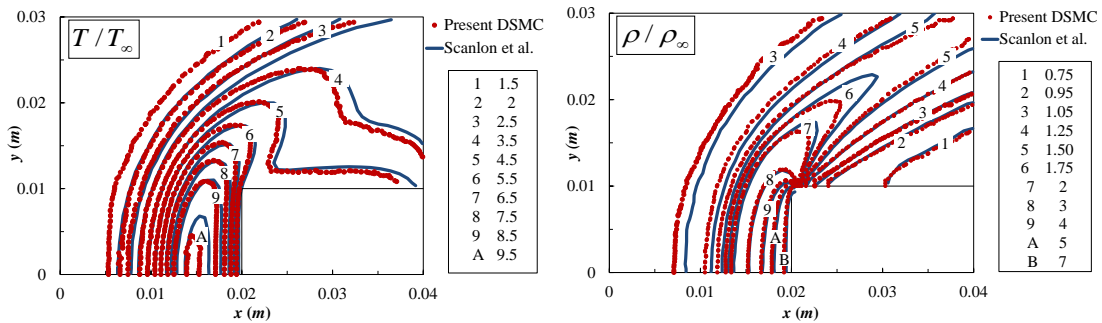


Figure 4.11. Temperature and density contours for hypersonic flow past a flat-nosed cylinder. A comparison of the results in this work with those computed in previous work done by Scanlon et al. [6].

corresponding free-stream Knudsen number is 0.01 based on the free-stream mean free path ( $\lambda_\infty = 0.003m$ ) and the diameter of the cylinder ( $D = 0.3048m$ ).

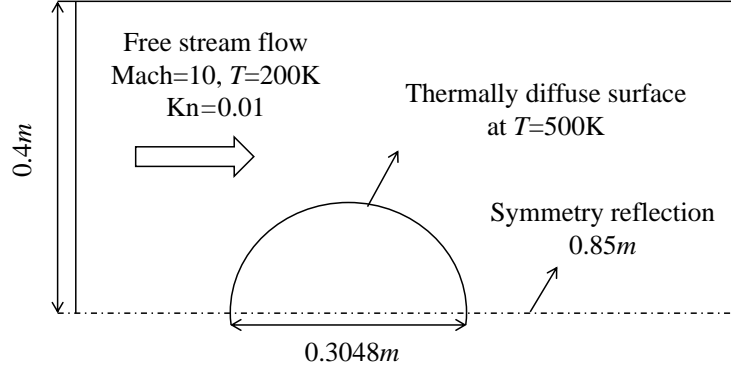


Figure 4.12. Sketch of the computational domain of argon hypersonic flow over a cylinder at  $Kn_\infty = 0.01$ ,  $Ma_\infty = 10$ ,  $T_\infty = 200$  K,  $n_\infty = 4.274 \times 10^{20}$  particles/m<sup>3</sup>.

As part of the validation process, the spatio-temporal adaptivity scheme is applied to this problem due to the rapid variations in the local molecular mean free path and the mean collision time presented. The computational grid is generated initially with a cell size of approximately  $(1 - 2) \lambda$  based on free-stream conditions, and 100 initial particles per cell. As the flow evolves within the simulated domain, the parameters  $\lambda$  and  $\tau_c$  are iteratively computed for each cell. All DSMC requirements are checked, i.e., the cell size smaller than the local mean free path, the time step smaller than the local mean collision time, and a number of simulated molecules around 20 – 30 molecules. If within any cell these conditions are not satisfied the spatio-temporal adaptation algorithm is called, and the parameters  $\Delta x_c$  and  $\Delta t$  are modified. In doing so, the cell size is adapted to  $0.25\lambda$  and the desired time step is adapted to  $0.333\tau_c$ . The total number of cells in the simulation domain changed from 7,680 to 10,480, the time step  $\Delta t$  changed from  $3.176\mu s$  to  $1.42\mu s$  and  $0.71\mu s$  within grid cells of the first and second temporal levels, respectively. Figure 4.13 compares contours of temperatures obtained using our DSMC algorithm with those reported previously using the virtual mesh refinement (VMR) module [7]. Figures 4.14, 4.15 show the density and temperature distribution along a vertical line just before the cylinder ( $x = 0.205m$ ) and in the wake region ( $x = 0.6m$ ), respectively. The results are very close to the benchmark.



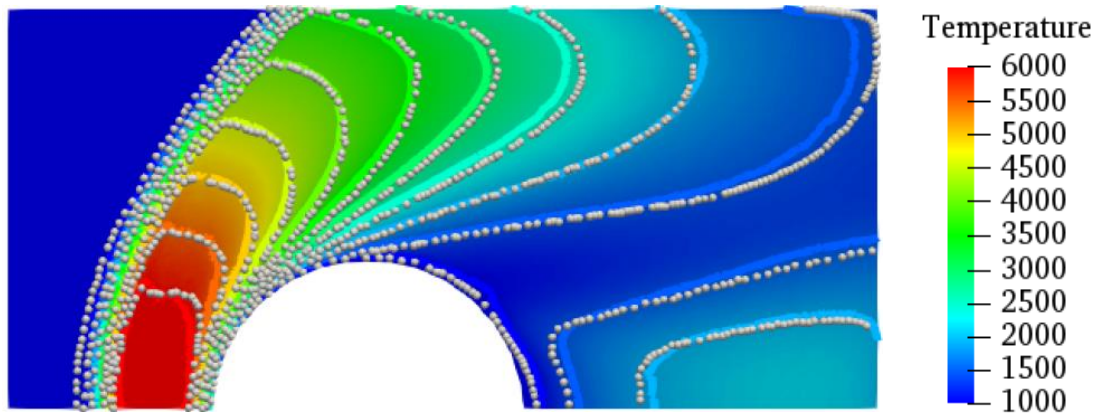


Figure 4.13. Contours of temperature of Mach-10 hypersonic flow past a circular cylinder; colored lines are DSMC data; the grey spheres are VMR data [7].

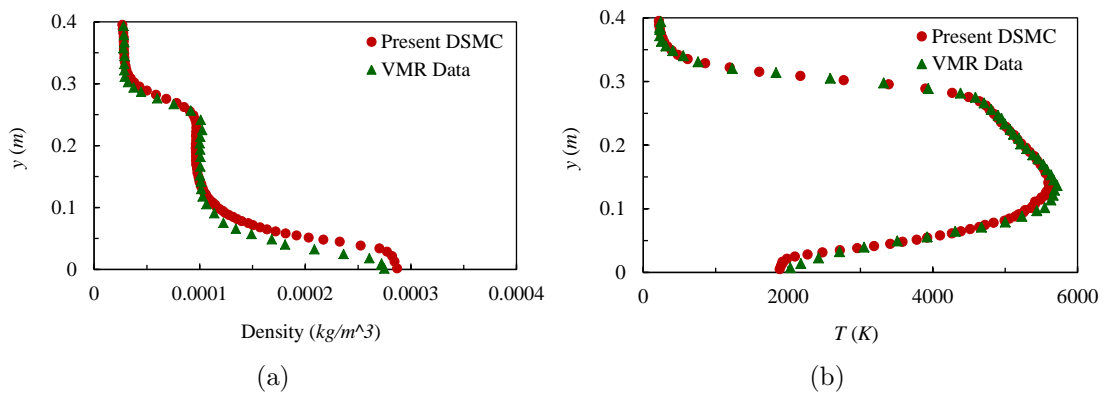


Figure 4.14. Density and Temperature distribution along a vertical line before the cylinder ( $x=0.205$  m). (a) Density (b) Temperature.

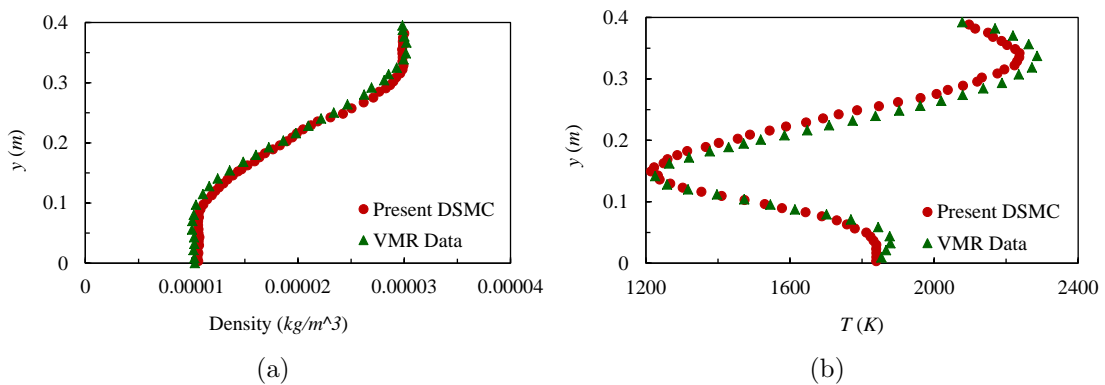


Figure 4.15. Density and Temperature distribution along a vertical line in the wake region ( $x=0.6$  m). (a) Density (b) Temperature.

# Chapter 5

## Conclusion and Future Work

In this work, a new parallel adaptive multi-scale DSMC algorithm is proposed and implemented to simulate unsteady rarefied flows in complex geometries. A hierarchical octree-based Cartesian grid is generated and a cut-cell method, where the triangulated solid boundary surface intersects the grid cells, is adopted. The selected geometry model is characterized by its low memory storage requirements compared to the use of non-Cartesian meshes, its ability to develop an efficient ray-tracing particle movement scheme, and its flexibility to develop a fully dynamic three-dimensional spatial and temporal adaptive scheme that maintains DSMC constraints consistent with the local variations of flow field properties. The proposed algorithm employs a novel spatio-temporal adaptive scheme based on the macroscopic averages of local flow properties. It also incorporates a new parallelization method based on running parallel unsteady DSMC simulations simultaneously and independently over multicores. Results for two- and three-dimensional benchmark test cases evidence the accuracy and robustness of the newly developed DSMC algorithm.

Future work should implement other aspects related to multi-dimensional flows involving gas mixtures, axially symmetric flows, moving boundary flows, and flows past multiple boundaries. In addition, this work could be extended to account for the rotational and vibrational degrees of freedom in diatomic and polyatomic gas. We expect to showcase the efficiency and accuracy of the algorithm in predicting unsteady flows encountered in transient three-dimensional problems in several micro- and nano-electromechanical systems (MEMS/NEMS) applications.

This work provides further insight on multiple applications for future research directions. In particular, the simulation of multi-physics problems, as fluid-structure interaction (FSI) problems that involve fluid flows interacting with movable and/or deformable solids or structures is increasingly needed in diverse applications. These include micro-mechanical devices, dynamic instabilities in structural engineering, and applications in biomechanics and medicine; to mention just a few. Despite the high attention, there is still a lack of established

computational methods which offer accuracy, flexibility, and efficiency, allowing to model and simulate general problems in this inherently multi-disciplinary field [83]. The simulation of the complete fluid-structure interaction problem could be done by integrating the proposed DSMC solver for the representation of the fluid into a three-dimensional finite element solver on the structural side. These solvers could be coupled iteratively by using their outputs. Outputs for the DSMC solver could then be inputs for the finite element solver and vice versa.

The proposed algorithm can also be developed to simulate multi-scale simulations of rarefied gas flows involving both continuum and rarefaction regions, and spanning a wide range of Knudsen number regimes. An interesting issue in such flows is the transition from the continuum to the rarefied regime and vice versa. The DSMC method can model the entire Knudsen number regime accurately but is computationally expensive to model the whole domain. This motivates the use of hybrid continuum-particle simulation methods, which can couple the Navier-Stokes equation to a DSMC solver [84, 85]. A different approach to handle continuum in very low Knudsen number regions could be by isolating these regions/collision cells from the rest of the simulation domain, enforcing local thermodynamic equilibrium, and re-initializing these cells from equilibrium.

Another possible application to be addressed is the simulation of heat transfer from microscale hot-wires sensors placed in a fluid flow. Thermal flow-meters are characterized by their high sensitivity and simplicity among others; however, their most significant problem is the high dependency on the thermo-physical properties of the working medium. They exhibit a dependency on the thermal conductivity, the specific heat capacity, the density, and the dynamic viscosity. A recent application of micro-thermal flow sensors is to investigate heat transfer through a periodic bunch of microcylinders arranged in a regular array. This could be used to measure the thermal conductivity and get its profile as a function of temperature, consequently, have a unique signature of the gas.

# Bibliography

- [1] M. Gad-el Hak, “The fluid mechanics of microdevices—the Freeman scholar lecture,” Journal of Fluids Engineering, vol. 121, no. 1, pp. 5–33, 1999.
- [2] G. A. Bird, Molecular gas dynamics. Clarendon Press, 1976. 76373657.
- [3] G. A. Bird, Molecular Gas Dynamics and the Direct Simulation of Gas Flows. No. v. 1, Clarendon Press, 1994. 94003873.
- [4] H. M. Cave, K. C. Tseng, J. S. Wu, M. C. Jermy, J. C. Huang, and S. P. Krumdieck, “Implementation of unsteady sampling procedures for the parallel direct simulation Monte Carlo method,” Journal of Computational Physics, vol. 227, pp. 6249–6271, 6/1 2008.
- [5] M. G. Coutinho, Guide to dynamic simulations of rigid bodies and particle systems. Springer Science & Business Media, 2012.
- [6] T. Scanlon, E. Roohi, C. White, M. Darbandi, and J. Reese, “An open source, parallel DSMC code for rarefied gas flows in arbitrary geometries,” Computers & Fluids, vol. 39, no. 10, pp. 2078–2089, 2010.
- [7] C. Su, K. Tseng, J. Wu, H. Cave, M. Jermy, and Y. Lian, “Two-level virtual mesh refinement algorithm in a parallelized DSMC code using unstructured grids,” Computers & Fluids, vol. 48, no. 1, pp. 113–124, 2011.
- [8] E. Oran, C. Oh, and B. Cybyk, “Direct simulation Monte Carlo: recent advances and applications,” Annual Review of Fluid Mechanics, vol. 30, no. 1, pp. 403–441, 1998.
- [9] N. A. Diab and I. A. Lakkis, “Investigation of the squeeze film dynamics underneath a microstructure with large oscillation amplitudes and inertia effects,” Journal of Tribology, vol. 138, no. 3, p. 031704, 2016.
- [10] M. B. Gerdroodbary, D. Ganji, M. Taeibi-Rahni, and S. Vakilipour, “Effect of Knudsen thermal force on the performance of low-pressure micro gas sensor,” The European Physical Journal Plus, vol. 132, no. 7, p. 315, 2017.

- [11] A. Frangi et al., Advances in multiphysics simulation and experimental testing of MEMS, vol. 2. Imperial College Press, 2008.
- [12] E. Roohi, “DSMC simulations of nanoscale and microscale gas flow,” in Encyclopedia of Microfluidics and Nanofluidics, pp. 681–693, Springer, 2015.
- [13] S. Dietrich and I. D. Boyd, “Scalar and parallel optimized implementation of the direct simulation Monte Carlo method,” Journal of Computational Physics, vol. 126, no. 2, pp. 328–342, 1996.
- [14] G. LeBeau, “A parallel implementation of the direct simulation Monte Carlo method,” Computer Methods in Applied Mechanics and Engineering, vol. 174, no. 3-4, pp. 319–337, 1999.
- [15] M. Ivanov, A. Kashkovsky, S. Gimelshein, G. Markelov, A. Alexeenko, Y. A. Bondar, G. Zhukova, S. Nikiforov, and P. Vaschenkov, “SMILE system for 2D/3D DSMC computations,” in Proceedings of 25th International Symposium on Rarefied Gas Dynamics, St. Petersburg, Russia, pp. 21–28, 2006.
- [16] G. Bird and M. Capitelli, “The DS2V/3V program suite for DSMC calculations,” in AIP conference proceedings, vol. 762, pp. 541–546, AIP, 2005.
- [17] D. Gao, C. Zhang, and T. Schwartzentruber, “A three-level Cartesian geometry-based implementation of the DSMC method,” in 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, p. 450, 2010.
- [18] A. Klothakis and I. Nikolos, “Modeling of rarefied hypersonic flows using the massively parallel DSMC kernel SPARTA,” in 8th Int. Congress Computational Mechanics, 2015.
- [19] G. B. Macpherson, N. Nordin, and H. G. Weller, “Particle tracking in unstructured, arbitrary polyhedral meshes for use in CFD and molecular dynamics,” International Journal for Numerical Methods in Biomedical Engineering, vol. 25, no. 3, pp. 263–273, 2009.
- [20] G. Bird, M. Gallis, J. Torczynski, and D. Rader, “Accuracy and efficiency of the sophisticated direct simulation Monte Carlo algorithm for simulating noncontinuum gas flows,” Physics of Fluids, vol. 21, no. 1, p. 017103, 2009.
- [21] V. I. Kolobov, R. R. Arslanbekov, V. V. Aristov, A. A. Frolova, S. A. Zabelok, M. Mareschal, and A. Santos, “Unified flow solver for transient rarefied-continuum flows,” in AIP Conference Proceedings, vol. 1501, pp. 414–421, AIP, 2012.

- [22] V. Kolobov, R. Arslanbekov, V. Aristov, A. Frolova, and S. A. Zabelok, “Unified solver for rarefied and continuum flows with adaptive mesh and algorithm refinement,” Journal of Computational Physics, vol. 223, no. 2, pp. 589–608, 2007.
- [23] S. A. Zabelok, V. I. Kolobov, R. R. Arslanbekov, M. Mareschal, and A. Santos, “GPU accelerated kinetic solvers for rarefied gas dynamics,” in AIP Conference Proceedings, vol. 1501, pp. 429–434, AIP, 2012.
- [24] S. Popinet, “Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries,” Journal of Computational Physics, vol. 190, no. 2, pp. 572–600, 2003.
- [25] M. Ivanov, G. Markelov, and S. Gimelshein, “Statistical simulation of reactive rarefied flows-numerical approach and applications,” in 7th AIAA/ASME Joint Thermophysics and Heat Transfer Conference, p. 2669, 1998.
- [26] D. Gao, C. Zhang, and T. E. Schwartzentruber, “Particle simulations of planetary probe flows employing automated mesh refinement,” Journal of Spacecraft and Rockets, vol. 48, no. 3, pp. 397–405, 2011.
- [27] S. E. Olson and A. J. Christlieb, “Gridless DSMC,” Journal of Computational Physics, vol. 227, no. 17, pp. 8035–8064, 2008.
- [28] R. Arslanbekov, V. Kolobov, J. Burt, and E. Josyula, “Direct simulation Monte Carlo with octree Cartesian mesh,” in 43rd AIAA Thermophysics Conference, p. 2990, 2012.
- [29] K. C. Kannenberg and I. D. Boyd, “Strategies for efficient particle resolution in the direct simulation Monte Carlo method,” Journal of Computational Physics, vol. 157, no. 2, pp. 727–745, 2000.
- [30] M. A. Gallis, J. Torczynski, D. Rader, and G. A. Bird, “Convergence behavior of a new DSMC algorithm,” Journal of Computational Physics, vol. 228, no. 12, pp. 4532–4548, 2009.
- [31] A. C. J. Wade, D. Baillie, and P. Blakie, “Direct simulation Monte Carlo method for cold-atom dynamics: Classical Boltzmann equation in the quantum collision regime,” Physical Review A, vol. 84, no. 2, p. 023612, 2011.
- [32] M. Laux, “Local time stepping with automatic adaptation for the DSMC method,” in 7th AIAA/ASME Joint Thermophysics and Heat Transfer Conference, p. 2670, 1998.

- [33] G. Cai, W. Su, and F. Hou, “Theoretical development for DSMC local time stepping technique,” Science China Technological Sciences, vol. 55, no. 10, pp. 2750–2756, 2012.
- [34] H. Samet, The Design and Analysis of Spatial Data Structures. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990.
- [35] C. Zhang and T. E. Schwartzentruber, “Robust cut-cell algorithms for DSMC implementations employing multi-level Cartesian grids,” Computers & Fluids, vol. 69, pp. 122–135, 2012.
- [36] B. G.A., “Approach to translational equilibrium in a rigid sphere gas,” Physics of Fluids, vol. 6, pp. 1518–1519, oct 1963. Provided by the SAO/NASA Astrophysics Data System.
- [37] 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, 1970. 70077285.
- [38] L. Boltzmann, Lectures on gas theory. Berkeley: University of California Press, 1964. ID: 556023.
- [39] C. Cercignani, The Boltzmann Equation and Its Applications. Springer-Verlag New York, 1988.
- [40] F. John, J. E. Marsden, and L. Sirovich, The Boltzmann Equation and Its Applications. Springer, -01-01 1988. doi: pmid:.
- [41] J. C. Maxwell, The Scientific Papers of James Clerk Maxwell, vol. 2. Cambridge University Press, 2011.
- [42] G. E. P. Box and M. E. Muller, “A note on the generation of random normal deviates,” The Annals of Mathematical Statistics, vol. 29, no. 2, pp. 610–611, 1958.
- [43] H. Akhlaghi and E. Roohi, “A novel algorithm for implementing a specified wall heat flux in DSMC: Application to micro/nano flows and hypersonic flows,” Computers & Fluids, vol. 127, pp. 78–101, 2016.
- [44] E. Roohi, V. SHARIATI, C. White, et al., “Evaluation of wall heat flux boundary condition in DSMC implemented in OPENFOAM,” in MIGRATE-2017 2nd MIGRATE Workshop & Summer School, 2017.
- [45] M. Gad-el Hak, MEMS: introduction and fundamentals. CRC press, 2005.
- [46] C. Shen, Rarefied gas dynamics: fundamentals, simulations and micro flows. Springer Science & Business Media, 2006.

- [47] A. J. Lofthouse, L. C. Scalabrin, and I. D. Boyd, “Velocity slip and temperature jump in hypersonic aerothermodynamics,” Journal of thermophysics and heat transfer, vol. 22, no. 1, p. 38, 2008.
- [48] M. Ikegawa and J. Kobayashi, “Development of rarefied gas flow simulator using the direct simulation Monte Carlo method(1 st report, 2-D flow analysis with the pressure conditions given at the upstream and downstream boundaries).,” TRANS. JAPAN SOC. MECH. ENG.(SER. B)., vol. 54, no. 507, pp. 3057–3060, 1988.
- [49] J.-S. Wu, F. Lee, and S.-C. Wong, “Pressure boundary treatment in micro-mechanical devices using the direct simulation Monte Carlo method,” JSME International Journal Series B Fluids and Thermal Engineering, vol. 44, no. 3, pp. 439–450, 2001.
- [50] R. P. Nance, D. B. Hash, and H. Hassan, “Role of boundary conditions in Monte Carlo simulation of microelectromechanical systems,” Journal of Thermophysics and Heat Transfer, vol. 12, no. 3, pp. 447–449, 1998.
- [51] M. Wang and Z. Li, “Simulations for gas flows in microgeometries using the direct simulation Monte Carlo method,” International Journal of Heat and Fluid Flow, vol. 25, no. 6, pp. 975–985, 2004.
- [52] W. Liou and Y. Fang, “Implicit boundary conditions for direct simulation Monte Carlo method in MEMS flow predictions,” In other words, vol. 2, p. 7, 2000.
- [53] C. White, M. K. Borg, T. J. Scanlon, and J. M. Reese, “Accounting for rotational non-equilibrium effects in subsonic DSMC boundary conditions,” in Journal of Physics: Conference Series, vol. 362, p. 012016, IOP Publishing, 2012.
- [54] E. Farbar and I. D. Boyd, “Subsonic flow boundary conditions for the direct simulation Monte Carlo method,” Computers & Fluids, vol. 102, pp. 99–110, 2014.
- [55] D. L. Whitfield, “Three-dimensional unsteady Euler equation solutions using flux vector splitting,” 1983.
- [56] J. Yang, J. Ye, J. Zheng, I. Wong, C. Lam, P. Xu, R. Chen, and Z. Zhu, “Using direct simulation Monte Carlo with improved boundary conditions for heat and mass transfer in microchannels,” Journal of Heat Transfer, vol. 132, no. 4, p. 041008, 2010.
- [57] C. R. Lilley, C. R. Lilley, and M. N. Macrossan, “Methods for implementing the stream boundary condition in DSMC computations,” International



Journal for Numerical Methods in Fluids, vol. 42, pp. 1363; 1363–1371; 1371, -08-30 2003. doi: 10.1002/fld.603 pmid:.

- [58] A. Prez and J. A. Morigo, “Computational efficiency of the inflow boundary conditions in DSMC simulation,” AIP Conference Proceedings, vol. 1501, no. 1, pp. 601–608, 2012. <http://aip.scitation.org/doi/pdf/10.1063/1.4769597>.
- [59] M. W. Tysanner, M. W. Tysanner, and A. L. Garcia, “Nonequilibrium behaviour of equilibrium reservoirs in molecular simulations,” International Journal for Numerical Methods in Fluids, vol. 48, pp. 1337; 1337–1349; 1349, -08-30 2005. doi: 10.1002/fld.983 pmid:.
- [60] E. Roohi and S. Stefanov, “Collision partner selection schemes in DSMC: From micro/nano flows to hypersonic flows,” Physics Reports, vol. 656, pp. 1–38, 2016.
- [61] G. A. Bird and G. A. Bird, “Direct simulation of gas flows at the molecular level,” Communications in applied numerical methods, vol. 4, pp. 165; 165–172; 172, -03-01 1988. doi: 10.1002/cnm.1630040205 pmid:.
- [62] T. Tokumasu, T. Tokumasu, and Y. Matsumoto, “Dynamic molecular collision (DMC) model for rarefied gas flow simulations by the DSMC method,” Physics of fluids (1994), vol. 11, pp. 1907; 1907–1920; 1920, -07-01 1999. doi: 10.1063/1.870053 pmid:.
- [63] G. A. Bird, Monte-Carlo Simulation in an Engineering Context, pp. 239–255. Rarefied Gas Dynamics, Parts I and II, American Institute of Aeronautics and Astronautics, 01/01; 2017/04 1981. 17; M1: 0; doi:10.2514/5.9781600865480.0239.0255.
- [64] K. Koura and H. Matsumoto, “Variable soft sphere molecular model for inversepowerlaw or LennardJones potential,” Physics of fluids.A, Fluid dynamics, vol. 3, pp. 2459; 2459–2465; 2465, -10-01 1991. doi: 10.1063/1.858184 pmid:.
- [65] K. Koura and H. Matsumoto, “Variable soft sphere molecular model for air species,” Physics of fluids.A, Fluid dynamics, vol. 4, pp. 1083; 1083–1085; 1085, -05-01 1992. doi: 10.1063/1.858262 pmid:.
- [66] J. Fan, “A generalized soft-sphere model for Monte Carlo simulation,” Physics of fluids (1994), vol. 14, pp. 4399; 4399–4405; 4405, -12-01 2002. doi: 10.1063/1.1521123 pmid:.
- [67] G. A. Bird, “Sophisticated DSMC, Notes Prep. a Short Course DSMC07 Meet. St. Fe, USA.,” September 2007.

- [68] “CEA/DEN, EDF R&D and OPEN CASCADE, SALOME: the open source integration platform for numerical simulation,” 2015. URL: <https://www.salome-platform.org/> [accessed July 2018].
- [69] T. Akenine-Möller, “Fast 3D triangle-box overlap testing,” in ACM siggraph 2005 courses, p. 8, ACM, 2005.
- [70] J. M. Burt, E. Josyula, and I. D. Boyd, “Novel Cartesian implementation of the direct simulation Monte Carlo method,” Journal of thermophysics and heat transfer, vol. 26, no. 2, pp. 258–270, 2012.
- [71] A. Y. Chang, “A Survey of Geometric Data Structures for Ray Tracing,” 2001.
- [72] A. S. Glassner, “Space subdivision for fast ray tracing,” IEEE Computer Graphics and Applications, vol. 4, pp. 15 – 24, October 1984. DOI: 10.1109/MCG.1984.6429331.
- [73] J.-S. Wu and Y.-Y. Lian, “Parallel three-dimensional direct simulation Monte Carlo method and its applications,” Computers & Fluids, vol. 32, no. 8, pp. 1133 – 1160, 2003.
- [74] K. C. Kannenberg, “Computational methods for the direct simulation Monte Carlo technique with application to plume impingement,” 1998. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2016-05-15.
- [75] S. Dietrich and I. D. Boyd, “Scalar and Parallel Optimized Implementation of the Direct Simulation Monte Carlo Method,” Journal of Computational Physics, vol. 126, no. 2, pp. 328 – 342, 1996.
- [76] Z.-X. Sun, Z. Tang, Y.-L. He, and W.-Q. Tao, “Proper cell dimension and number of particles per cell for DSMC,” Computers & Fluids, vol. 50, no. 1, pp. 1–9, 2011.
- [77] J. H. Park, P. Bahukudumbi, and A. Beskok, “Rarefaction effects on shear driven oscillatory gas flows: A direct simulation Monte Carlo study in the entire Knudsen regime,” Physics of Fluids, vol. 16, no. 2, pp. 317–330, 2004.
- [78] M. N. Ozisik, Boundary value problems of heat conduction. Courier Corporation, 2013.
- [79] N. G. Hadjiconstantinou, “Validation of a second-order slip model for dilute gas flows,” Microscale Thermophysical Engineering, vol. 9, no. 2, pp. 137–153, 2005.

- [80] J.-S. Wu and K.-C. Tseng, “Analysis of micro-scale gas flows with pressure boundaries using direct simulation Monte Carlo method,” Computers & Fluids, vol. 30, no. 6, pp. 711–735, 2001.
- [81] F. J. Alexander, A. L. Garcia, and B. J. Alder, “Direct simulation Monte Carlo for thin-film bearings,” Physics of Fluids, vol. 6, no. 12, pp. 3854–3860, 1994.
- [82] B. Goshayeshi, E. Roohi, and S. Stefanov, “DSMC simulation of hypersonic flows using an improved SBT-TAS technique,” Journal of Computational Physics, vol. 303, pp. 28–44, 2015.
- [83] H. J. Bungartz, M. Mehl, and M. Schäfer, Fluid structure interaction II: modelling, simulation, optimization, vol. 73. Berlin: Springer, 2010.
- [84] N. Hadjiconstantinou, “Discussion of recent developments in hybrid atomistic-continuum methods for multiscale hydrodynamics,” Technical Sciences, vol. 53, no. 4, 2005.
- [85] A. Donev, J. B. Bell, A. L. Garcia, and B. J. Alder, “A hybrid particle-continuum method for hydrodynamics of complex fluids,” Multiscale Modeling & Simulation, vol. 8, no. 3, pp. 871–911, 2010.