SIMULATION OF ENGINEERED NANOSTRUCTURED THIN FILMS

by

JASON CHEUNG

A thesis submitted to the
Department of Physics, Engineering Physics and Astronomy
in conformity with the requirements for
the degree of Master of Science (Engineering)

Queen’s University
Kingston, Ontario, Canada
March 2009

Copyright © Jason Cheung, 2009
Abstract

The invention of the Glancing Angle Deposition (GLAD) technique a decade ago enabled the fabrication of nanostructured thin films with highly tailorable structural, electrical, optical, and magnetic properties. Here a three-dimensional atomic-scale growth simulator has been developed to model the growth of thin film materials fabricated with the GLAD technique, utilizing the Monte Carlo (MC) and Kinetic Monte Carlo (KMC) methods; the simulator is capable of predicting film structure under a wide range of deposition conditions with a high degree of accuracy as compared to experiment. The stochastic evaporation and transport of atoms from the vapor source to the substrate is modeled as random ballistic deposition, incorporating the dynamic variation in substrate orientation that is central to the GLAD technique, and surface adatom diffusion is modeled as either an activated random walk (MC), or as energy dependent complete system transitions with rates calculated based on site-specific bond counting (KMC). The Sculptured Nanostructured Film Simulator (SNS) provides a three-dimensional physical prediction of film structure given a set of deposition conditions, enabling the calculation of film properties including porosity, roughness, and fractal dimension. Simulations were performed under various growth conditions in order to gain an understanding of the effects of incident angle, substrate rotation, tilt angle, and temperature on the resulting morphology of the
thin film. Analysis of the evolution of film porosity during growth suggests a complex growth dynamic with significant variations with changes in tilt or substrate motion, in good agreement with x-ray reflectivity measurements. Future development will merge the physical structure growth simulator, SNS, with Finite-Difference Time-Domain (FDTD) electromagnetics simulation to allow predictive design of nanostructured optical materials.
Acknowledgments

Without the guidance and encouragement of the following people, this thesis could not have been completed. To all my friends whom I have met throughout my university life, the memories and friendships will never be forgotten. To my family in the Queen’s Chinese Catholic Community, with whom I have shared countless joy and laughter. You have all deepened my spiritual life by showing me love, humility, faith, and compassion. To Chelsea Elliott, for her help with thesis editing; and to Cristina Buzea, for her helpful suggestions and discussions. To my labmates Tim Brown, Jian Yang, and Alex Kerkau, for their support and advice, as well as the occasional game of table tennis. To Dr. Towner, for his stimulating discussion and help on the Finite Difference Time Domain simulations. I would like to extend my gratitude and appreciation to Dr. Kevin Robbie, who gave me this opportunity to work on and study the fascinating world of nanostructured materials, and for his guidance and discussions throughout the thesis. To Sharon Liu, for her emotional support and encouragement. To my parents and sister, who supported me throughout my life and provided me with the opportunity to succeed and the freedom to do what I am interested in.
Abbreviations

AFM  Atomic Force Microscopy
BD   Ballistic Deposition
CN   Coordination number
DFT  Density Functional Theory
EDP  Electron Density Profile
EMA  Effective Medium Approximation
EW   Edwards-Wilkinson
FDTD Finite Difference Time Domain
FM   Frank-van der Merwe
GLAD Glancing Angle Deposition
HPCVL High Performance Computing Virtual Laboratory
KMC  Kinetic Monte Carlo
KPZ  Kardar-Parisi-Zhang
MC   Monte Carlo
MD   Molecular Dynamics
MPI  Message Passing Interface
NN   Nearest Neighbour
NNN  Next Nearest Neighbour
Open MP  Open Multi-Processing
PEC  Perfect Electric Conductor
PMC  Perfect Magnetic Conductor
PML  Perfectly Matched Layer
PVD  Physical Vapor Deposition
QMC  Quantum Monte Carlo
RE  Rate Equation
RMS  Root Mean Square
RR  Random Relaxation
RSOS  Restricted Solid On Solid
SEM  Scanning Electron Microscopy
SK  Stranski-Krastanov
SNS-MC  Sculptured Nanostructured Films Simulator - Monte Carlo
SNS-KMC  Sculptured Nanostructured Films Simulator - Kinetic Monte Carlo
SOS  Solid On Solid
STM  Scanning Tunneling Microscopy
SZM  Structure zone model
TST  Transition State Theory
UHV  Ultra High Vacuum
UPML  Uniaxial Perfectly Matched Layer
VFIGS  Virtual Film Growth Simulator
VW  Volmer-Weber
WV  Wolf-Villiain
XRR  X-ray Reflectivity
# Table of Contents

Abstract i

Acknowledgments iii

Abbreviations iv

Table of Contents vi

List of Tables viii

List of Figures ix

Chapter 1: Introduction 1

1.1 Introduction .................................................. 1
1.2 General overview ........................................... 3
1.3 Finite Difference Time Domain method ....................... 5
1.4 Motivation and scope of thesis ............................... 6

Chapter 2: Growth Models and Processes ......................... 7

2.1 Nanostructure growth with GLAD .......................... 9
2.2 Growth processes ........................................... 16
2.3 Deposition model ............................................ 19
2.4 Growth models .............................................. 21
2.5 Surface diffusion model .................................... 25
2.6 Desorption model .......................................... 29
2.7 Atomic bindings ............................................. 29

Chapter 3: Physical Structure Simulation ....................... 31

3.1 Metropolis Monte Carlo algorithm ........................ 33
3.2 Monte Carlo Simulation ..................................... 34
3.3 Kinetic Monte Carlo Simulation ............................ 37
3.4 Surface properties .......................................... 41
List of Tables

2.1 Diffusion processes .............................................. 27
2.2 Activation energy for Au(111) on graphite [1] .................. 28

5.1 Wave form code in SNS. ......................................... 66
5.2 Substrate code in SNS ............................................ 67
List of Figures

2.1 Geometry of the GLAD system ........................................... 9
2.2 Effect of material variation on the film density as a function of vapor flux angle. [2] ...................................................... 12
2.3 SEM images of film deposited at oblique angles with various materials:
   a) Ag $\alpha=86^\circ$, b) Al $\alpha=83^\circ$, c) Al-N, d) As$_2$S$_3$ $\alpha=80^\circ$, e) As$_2$Se$_3$, f) Au $\alpha=80^\circ$, g) Be, h) Be, i) Co $\alpha=85^\circ$, j) Cr $\alpha=87^\circ$, k) Cu $\alpha=75^\circ$, l) Mg $\alpha=75^\circ$, m) Ni $\alpha=87^\circ$, n) Pt $\alpha=87^\circ$, o) Sb$_2$Se$_3$ $\alpha=80^\circ$, p) Si $\alpha=83^\circ$, q) SiO $\alpha=87^\circ$, r) TiO$_2$ $\alpha=80^\circ$, s) YSZ $\alpha=45^\circ$, t) WO$_3$ $\alpha=80^\circ$ [3] ....... 14
2.4 Thin films fabricated with GLAD system. a) Si on Si(100) thin film with slanted columns, deposited at $\alpha = 83^\circ$, (b) Si on Si(100) columnar thin film, deposited at $\alpha = 83^\circ$, (c) Si spirals, (d) Si rugate film, (e) Si rugate with antireflection layer on glass substrate, (f) Si square helix, deposited at $\alpha = 80^\circ$, (g) Cu on Si(100) zig-zag, (h) Si square spiral on glass, deposited at $\alpha = 83^\circ$, (i) Si envelope corrected rugate on glass substrate. [4] ................................................................. 15
2.5 Illustration of a) Ballistic Deposition model and b) Random Relaxation model ......................................................................... 20
2.6 Solid-on-solid model [5]. ............................................................ 22
4.1 Position of the electric and magnetic field components in a unit cell of the Yee space lattice. [6] ...................................................... 53
4.2 PEC on the top surface of the Yee cube, where the E-field on the PEC surface are set to zero ..................................................... 58
5.1 KMC program structure ............................................................... 63
5.2 Substrate profiles 1-13. ............................................................... 68
5.3 Simulated deposition on 13 prepatterned substrates. .................. 69
5.4 Density profile illustration ......................................................... 70
5.5 Effect of deposition on prepatterned substrates. Deposited at $\alpha = 85^\circ$ on non-rotating substrate. Substrate pre-patterned structures are described in Table 5.2 and shown in Figure 5.3. Note that height corresponds to the distance from the substrate surface. ............... 71
5.6 Controllable nucleation growth on pre-patterned substrates (a) & (b) Cylinder substrate (type 7, radian=1, 0.1), (c) & (d) Empty cone substrate (type 11, radian=1, 0.1), (e) Plane substrate, with incidence angle $\alpha = 85^\circ$ and rotational rate $\phi' = 1000$ particles/rev. First two rows show the top and cross sectional view of the film, third row shows the cross sectional view of the substrates. 

5.7 Density profiles of films deposited at $\alpha = 85^\circ$ with fast rotation (phi speed=1000 particles/rev) to substrate 7 and substrate 11.

5.8 Growth evolution on concave substrate with rotation: $\alpha = 65^\circ$–$70^\circ$ with phi speed = 10000 particles / rev. (a) thickness = 34 (b) thickness = 72 (c) thickness = 200.

5.9 $\alpha = 85^\circ$ with no rotation on convex substrate.

5.10 Simulated surface and cross sectional views of non-rotating films at various incidence angles, $\alpha$.

5.11 Effect of deposition angle alpha of non-rotating films on a) film density b) RMS roughness.

5.12 Deposition angle effect: density profile of non-rotating substrate for $\alpha=0^\circ$ to $88^\circ$, $d(\alpha)=1$.

5.13 Electron density profile for a) $\alpha = 40^\circ$, b) $\alpha = 60^\circ$, c) $\alpha = 70^\circ$ [7].

5.14 Effect of deposition angle on density of KMC simulated film on Cu at 125K. Open circle: simulation results at 125K; Open square: simulation at 300K; Filled square: experimental data at 300K [8].

5.15 Simulated surface and cross sectional views of helix films, with pitch=100, phi period = 30000 particle / rev, at various incidence angles, $\alpha$.

5.16 Density profile of helix films, with pitch=100, phi period = 30000 particle / rev at various incidence angles, $\alpha$.

5.17 a) Density as a function of incidence angle b) RMS roughness as a function of incidence angle, for helix film with pitch=100, phi period = 30000 particle / rev at various incidence angles.

5.18 Simulated surface and cross sectional view of helix films, with pitch=100 and $\alpha=80^\circ$, at various phi periods, $T_\phi$. Below show the cross sectional view of SiO$_2$ films grown at comparable rotational speeds. [9].

5.19 Density profile of helix films, with pitch=100 and $\alpha=80^\circ$, at various phi periods.

5.20 a) Density as a function of phi period b) RMS roughness as a function of phi period, for helix films with pitch=100 and $\alpha=80^\circ$, at various phi periods.

5.21 Simulated surface and cross sectional view of zig zag films, with zig zag height=100 at various incidence angles, $\alpha$. 

---

x
5.22 Density profile of zig zag films, with zig zag height=100 at various incidence angles, inset shows density variation as a function of incidence angle for zig zag films.

5.23 (a) Density as a function of incidence angle (b) RMS roughness as a function of incidence angle, for zig zag films with zig zag height=100 at various incidence angles.

5.24 (a) SEM image of a 12 period rugate filter with sinusoidal variation of refractive index fabricated with GLAD, (b) A 12 period rugate film simulated with VFIGS (with input parameters γ=0.6×4/9 and S=20).

5.25 Simulated surface and cross sectional view of rugate films, alpha varies sinusoidally between 63° and 86°, rugate height=100 at various rotation speed (thickness/rev), φ′.

5.26 Density profile of rugate films, alpha varies sinusoidally between 63° and 86°, rugate height=100. Inset shows the phase shift amplitude with respect to φ′=100.

5.27 a) Density as a function of rotation period b) RMS roughness as a function of rotation period for rugate films, alpha varies sinusoidally between 63° and 86° rugate height=100.

5.28 Simulated surface and cross sectional views of non-rotating films, with α=70° at various number of random walk steps.

5.29 Density profile of non-rotating films, with α=70° at various number of random walk steps.

5.30 a) Density as a function of number of random walk steps, b) RMS roughness as a function of number of random walk steps, for non-rotating films with α=70°.

5.31 Simulated surface and cross-sectional views of non-rotating films, with a) α=75°, b) α=80°, c) α=85°, d) α=87°, e) α=88°, and f) α=89°.

5.32 SNS-MC and SNS-KMC density comparison for non-rotating substrate (MC: nwalk=0, T=10, KMC: E_diff=0.1 dE=0.1, T=10) for α=75°, 80°, 85°, 87°, 88°, 89°.

5.33 Roughness evolution profile of KMC simulated film, non-rotating with α=75°, 80°, 85°, 87°, 88°, 89°.

5.34 (a) Rectangular pulse propagation using scalar wave equation with S=1, and pulse width = 40. (b) Rectangular pulse propagation using scalar wave equation with S=0.99, and pulse width = 40.

5.35 (a) Gaussian pulse propagation using scalar wave equation with S=1, and pulse width = 40 cells, (b) Gaussian pulse propagation using scalar wave equation with S=0.99, and pulse width = 40.
5.36 Visualization of the E and H field as it hits the simulated perfect electric conductor (Ez = 0) on the right boundary. Note that Ez is reflected after it hits the right boundary and continues to propagate in the x direction, where E is in the z direction while H remains in y direction.

5.37 Visualization of the Ez and Hy field in the presence of a simulated perfect magnetic conductor at the right boundary. The perfect magnetic conductor reflects the incident Hy wave in a 1D plane wave propagation simulation. The H field is now in +y direction and the E field remains in the +z direction.

A.1

xii
Chapter 1

Introduction

1.1 Introduction

Since the invention of the scanning tunneling microscope in the 1980s, developments in the synthesis and study of nanomaterials have sparked great interest due to their remarkable structural, electronic, optical, and magnetic properties. Nanotechnology is the science of engineering of functional systems at the molecular level. It is the method of controlling matter at a near atomic scale to produce advanced materials and devices. For example, tetrahexahedral (24-facet) platinum nanocrystals improve efficiency of hydrogen production for fuel cells [11]; nano-magnetic materials, such as nanostructured Fe-Co, exhibit superior magnetic properties that promise applications in computer chips and storage [12]; nano-biomedical sensors improve glucose monitoring for diabetics as well as cancer diagnostics and detection [13]; advances in nanotube filters promise a high quality yet cost-effective way to provide drinking water to developing countries [14]; nano-molecular machines deliver medicine and
repair cells [15]. Nanostructured thin films have many potential scientific applications, such as thermal barrier coatings, dental adhesives, optical polarizers/rotators, chemical sensors, biocompatible prosthetic coatings, chiral catalysts, and optical interference filters. This list gives only a small glimpse of the emerging applications that nanotechnology promises.

The electronic, magnetic, and optical properties of materials are highly sensitive to the nanostructure of the material. Precise control of the nanostructure in terms of the morphology, porosity, and crystal structure is required to significantly improve the properties of engineered nanostructured materials. Over the past two decades, nanoparticles, quantum dots, and nanocomposite materials have attracted much attention due to their tunable optical properties. The optical properties are tuned by tailoring the size, shape, and chemical composition of the materials [16]. For example, the absorption spectra of CdSe quantum dots can be varied by altering quantum dots size through control of the concentration of the electrolyte [17], the emission spectra of InP, InAs and CdSe nanocrystals can be tuned from 400 to 2000 nm by tailoring their sizes [18]. Nanostructured thin films fabricated with Glancing Angle Deposition (GLAD) have the potential to precisely tune the optical properties needed for many applications. Ultimately, development of a synthesis and simulation model will allow accurate design of nanostructured thin films, thereby optimizing material properties and minimizing manufacturing cost. This thesis focuses on 3-Dimensional Monte Carlo (MC) and Kinetic Monte Carlo (KMC) physical structure simulations of nanostructured thin films fabricated with the GLAD technique, and subsequently uses the Finite Difference Time Domain (FDTD) method to simulate the electromagnetic response. Accurately predicting the final film morphologies and optical response
characteristics from various growth conditions will ultimately allow us to engineer and optimize the design parameters to produce films with desirable optical characteristics.

Computational simulations of thin films offer a unique way of controlling and characterizing the properties of thin films. In addition, they offer a platform to test hypotheses and models of atomic processes. In order to understand how the properties of nanostructured thin films are changed by the choice of depositing materials, substrate, and deposition condition, molecular models of the deposition process were studied and incorporated into the simulation. The simulated models and properties can be directly compared with results from experiment, and the continual improvement in the simulation model will lead to greater control of the properties of fabricated films. The reliability of the results from simulation depends upon the fundamental models upon which the simulator is based. However, molecular growth processes that take place in real systems are not yet well understood. This study reviews various deposition models, surface relaxation models, and simulation techniques that have been developed, and then simulation results are compared to recent experiments.

1.2 General overview

1.2.1 Monte Carlo method

The Monte Carlo method has been used since 1953 to study the transport, adsorption and diffusion dynamics, surface morphology, crystal growth, and material properties in surfaces, nanostructures and nanoparticles [19]. These pioneering works [20] [21] [22] have stimulated extensive research into engineered nanostructured growth with specific material properties that have a wide range of applications in semiconductor
CHAPTER 1. INTRODUCTION

devices, opto-electronics, solar cells, biomedicine, etc. A virtually unlimited variety of three-dimensional nanostructures can be fabricated using the GLAD method. Nanostructured materials have morphological features smaller than one micron in at least one dimension, and as such their electrical, optical and magnetics properties are highly dependent on their shape and molecular crystal structure. The ability to precisely control the nanostructure can yield materials with very interesting and unique properties.

The MC method is a widely used computational algorithm for simulating physical systems with stochastic characteristics. For example the MC method is used to solve coupled differential equations for radiation fields, in molecular modeling, for simulation of annealing for protein structure [23], and for carrier transport in semiconductors. For large scale systems with many variables and degrees of freedom, the Monte Carlo method is efficient because it exhibits convergence of \( \frac{1}{\sqrt{n}} \) [24], independent of dimension, where \( n \) is the number of points generated, as long as the function is well-behaved. The convergence rate \( \frac{1}{\sqrt{n}} \) indicates that to in order to half the error estimate, four times the number of samples has to be evaluated. Other MC methods include direction-simulation MC [25] [26] [27], dynamic MC [28], kinetic MC [1] [29], quasi-MC [30], and Markov Chain MC [31].

The MC method for simulating particle transport is a statistical approach to solving the linearized Boltzmann equation. The essence of the MC method of ballistic deposition is that the trajectory of individual particles is followed in an analog fashion in the simulation space. Particles undergo a series of events, where each particle is generated at a random position above the growing film and follows a straight line trajectory until it strikes the film surface, then the particle undergoes surface
diffusion or a random walk to minimize surface potential before becoming permanently incorporated in the growing film surface.

1.2.2 Kinetic Monte Carlo method

The Kinetic Monte Carlo method simulates the dynamic evolution of a physical system quantitatively by considering the rate of processes associated with the formation of the system. KMC simulation has been used to explore phase transition kinetics of epitaxial crystal growth [32][33]. The advantage of the KMC method is that it is computationally inexpensive for large scale models, and is able to simulate specific kinetic processes with a rate table. The rates for the kinetic processes can be calculated from Molecular Dynamics (MD) or Density Functional Theory (DFT). For simulating GLAD nanostructured thin film growth, three major processes are considered during nucleation and growth: ballistic deposition, adatom diffusion, and adatom desorption. Chapter 5 gives a more detailed explanation of the KMC models, algorithm, and the procedure of the thin film growth simulation program.

1.3 Finite Difference Time Domain method

The Finite Difference Time Domain method is a computational electrodynamics modeling technique suitable for simulating the optical properties of nanostructured thin films [6]. It employs the finite difference approximation to solve the time dependent Maxwell’s equations. Chapter 4 gives a detailed review of the FDTD method, and subsequently the result of a 1D FDTD simulation program is presented in Chapter 5.
1.4 Motivation and scope of thesis

The goal of this thesis is to develop a Monte Carlo simulator to accurately simulate GLAD nanostructured thin films that will aid in the design of thin films with specific optical properties. A smaller focus is placed on simulating the dynamic evolution of nanostructured films using the KMC method and to gain a better understanding of the atomic processes during deposition. Various simulations and growth models that govern thin film growth were studied and incorporated into the our own simulators: SNS-MC (Sculptured Nanostructure Films Simulator - Monte Carlo) and SNS-KMC (Sculptured Nanostructured Films Simulator - Kinetic Monte Carlo). Deposition conditions such as deposition rate, surface energy, tilt and rotation rate, angles, temperature, type of substrate, and flux, were used as inputs to generate the final structure. The relationship between the deposition conditions and the resulting film properties such as its density, roughness, and fractal dimension were investigated and compared with experiments.
Chapter 2

Growth Models and Processes

The primary objective of Monte Carlo modeling is the investigation of the equilibrium properties of interacting systems. It is useful to consider the relationship between the equilibrium ensemble and the dynamic surface diffusion models. The equilibrium state is determined by the long time average \((t \to \infty)\) of the microscopic dynamics with a defined energy function.

Monte Carlo simulation investigates a dynamical system by studying the properties of particular models to determine an approximate model for the system. In general, the dynamic properties of a growing surface can be described by a stochastic Langevin equation [34]:

\[
\frac{\partial h(x, t)}{\partial t} = v \nabla^2 h(x, t) - K \nabla^4 h + \lambda_1 (\nabla h)^2 + \lambda_2 \nabla^2 (\nabla h)^2 + \eta(x, t) \tag{2.1}
\]

The first term in equation 2.1 accounts for the growth by surface relaxation, where \(v\) is the surface tension coefficient, the second term accounts for the hydrodynamic
smoothing of the surface, where K is a constant, the third and fourth term accounts for the directional velocity of the particles to the tangent plane of the surface, where \( \lambda_1 \) and \( \lambda_2 \) are constant, \( h(x,t) \) is the height of the surface at the position \( x \) of a d-dimensional substrate at time \( t \), and the fifth term accounts for the fluctuations, where \( \eta(x,t) \) is a stochastic gaussian noise term with a zero average, which satisfies:

\[
\langle \eta(x,t) \eta(x',t') \rangle = 2D \delta(x-x') \delta(t-t')
\]  

(2.2)

where \( D \) is the stochastic noise strength, and \( \delta(x) \) is the dirac delta function.

Extensive research has been done to evaluate the critical scaling exponents using various growth models such as the Family model [35], the Restricted Solid on Solid (RSOS) model [36], the Krug model [37], and the Wolf-Villain (WV) model [38]. The scaling relation characterizes the dynamic of the roughening process, Family and Viscek [39] shows that in the absence of any characteristic length scale, the interface width is a power function of time. The scaling relation is defined as: \( w(L,t) \sim L^\alpha f \left( \frac{t}{L} \right) \), where \( L \) is the system size, \( t \) is time, \( w(L,t) \) is the interface width, \( \alpha \) is the roughness exponent, and \( \beta \) is the growth exponent. A scaling law links the roughness exponent to the growth exponent, such that \( z = \frac{\alpha}{\beta} \).

The Family model follows the Edward-Wilkinson universality class and the resulting critical exponents are \( \alpha = 1/2, \beta = 1/4 \), and \( z=2 \) in \( d=1+1 \), while the RSOS follows the Kardar-Parisi-Zhang (KPZ) universality class [40] and the critical exponents are \( \alpha = 1/2, \beta = 1/3 \), and \( z=3/2 \) in \( d=1+1 \). Each growth model describes the behavior of the particles after they randomly deposit on the surface, resulting in differing universality classes with characteristic power law scaling of the surface. For the Family model, particles diffuse to a nearest neighbour (NN) site at lower height.
For the RSOS mode, a particle is only accepted if the RSOS condition $|\delta h| = 0.1$ is satisfied. For the WV model, particles diffuse to a NN sites with more NN bonds. The next section gives a more detailed explanation of some of these models.

### 2.1 Nanostructure growth with GLAD

Nanostructured thin films have many practical application in semiconductors, photovoltaics, magnetics, electronics, and optoelectronic devices. In order to make nano-materials with specific properties, a full understanding of the atomic and molecular processes in epitaxial growth, and what parameters influences the final properties of the film, is needed. A general overview of the thin film growth process employing the GLAD method is presented below followed by theoretical models of the deposition process.

![Geometry of the GLAD system](image)

Figure 2.1: Geometry of the GLAD system

The GLAD deposition technique, as shown in Figure 2.1, produces engineered structured thin films by varying the tilt ($\alpha$), and rotation ($\phi$) during the deposition
process. In physical vapor deposition (PVD), the source material is heated with a high energy electron beam (5 - 10 kV) until a suitable vapor pressure is reached. The ultrahigh vacuum (UHV) GLAD system operates at $10^{-9}$ Torr, where the mean free path between molecular collision is large compared to chamber dimensions, therefore it can be assumed that there is no intermediate collision with gas molecules and that depositing atoms travel ballistically (i.e. without scattering) between the source and the substrate.

The principal control that makes GLAD powerful is the variation of both the substrate rotation and tilt angle during deposition. Atomic shadowing and surface diffusion compete with each other to produce films with unique structure and properties. Atomic shadowing occurs when an area of film growth prevents incoming flux from directly depositing in the existing film’s geometric shadow, however, atoms may still diffuse to the shadowed area. GLAD films are deposited at near-glancing incidence, where the effect of atomic shadowing is intensified, producing films that are extremely porous. Furthermore, the directionality of the incoming flux induces asymmetric shadowing resulting in films that exhibit anisotropy properties. For vapor incident at an angle $\alpha > 70^\circ$ from the substrate normal, the increase in the shadowing effect leads to a rapid increase in porosity with the polar angle, resulting in isolated columns developing in the film. Tait [41] studied the relationship between the column tilt angle, $\beta$, relative to the polar angle, $\alpha$, and derived an expression that gives good results for vapor incidence angle $\alpha > 70^\circ$:

$$\beta = \alpha - \arcsin \left[ \frac{1 - \cos \alpha}{2} \right]$$  \hspace{1cm} (2.3)

Equation 2.3 is applied to films that exhibit columnar structure, and that belong
to zone 1 in the structured zone model (SZM) (as described in the next section). This equation assumes that the inclined columns are capped with a hemispherical top, diffusion effects are minimal, and that there is sufficient distance between the flux and the source to produce a highly columnated vapor flux. Other factors that may affect the final film properties and structure such as surface contamination, deposition rate, and deposition materials, are not taken into account in the Tait equation.

The effect of material variation on film density as a function of flux angle is shown in Figure 2.2. SEM images of obliquely deposited films in Figure 2.3 illustrate the wide range of morphology with various materials, demonstrating that the morphology is a result of the material’s physical, chemical, electrical and atomic properties, as well as the deposition condition in which the films are fabricated. Tait [41] derived a model that relates the deposition angle to the film density, \( \rho \), based on the relationship between column spacing to column width:

\[
\rho = 2 \cdot \frac{\cos(\alpha)}{1 + \cos(\alpha)}
\]

(2.4)
Figure 2.2: Effect of material variation on the film density as a function of vapor flux angle. [2]

By dynamically varying the tilt, $\alpha$, and rotation, $\phi$, angles during deposition, the GLAD technique can be used to fabricate a large variety of complex nanostructured thin films with tunable optical, electrical, and magnetic properties. Some examples of nanostructures that can be produced with GLAD are shown in Figure 2.4. Interesting and complex 3D nanostructures can be produced by precisely manipulating the substrate during deposition. For instance, GLAD is capable of fabricating a film with a helical superstructure and an underlying zig zag substructure, or a film with a sub 3D structure (sphere, pyramid, cube) with partial shielding during deposition. The morphological characteristics of the structure, such as column angle, column diameter, column spacing, density, and pitch, are functions of the deposition angle and rotation rate. The substrate orientation in $\alpha$ and $\phi$ can be controlled based on
mathematical functions of the position as a function of height. The technique for fabricating various types of structure is shown by Robbie [2]. The resulting films demonstrate the power law scaling of fractal structures, and exhibit scale invariant morphological characteristics such as RMS roughness. [42].
Figure 2.3: SEM images of film deposited at oblique angles with various materials: a) Ag $\alpha=86^\circ$, b) Al $\alpha=83^\circ$, c) Al-N, d) As$_2$S$_3$ $\alpha=80^\circ$, e) As$_2$Se$_3$, f) Au $\alpha=80^\circ$, g) Be, h) Be, i) Co $\alpha=85^\circ$, j) Cr $\alpha=87^\circ$, k) Cu $\alpha=75^\circ$, l) Mg $\alpha=75^\circ$, m) Ni $\alpha=87^\circ$, n) Pt $\alpha=87^\circ$, o) Sb$_2$Se$_3$ $\alpha=80^\circ$, p) Si $\alpha=83^\circ$, q) SiO $\alpha=87^\circ$, r) TiO$_2$ $\alpha=80^\circ$, s) YSZ $\alpha=45^\circ$, t) WO$_3$ $\alpha=80^\circ$ [3].
Figure 2.4: Thin films fabricated with GLAD system. a) Si on Si(100) thin film with slanted columns, deposited at $\alpha = 83^\circ$, (b) Si on Si(100) columnar thin film, deposited at $\alpha = 83^\circ$, (c) Si spirals, (d) Si rugate film, (e) Si rugate with antireflection layer on glass substrate, (f) Si square helix, deposited at $\alpha = 80^\circ$, (g) Cu on Si(100) zig-zag, (h) Si square spiral on glass, deposited at $\alpha = 83^\circ$, (i) Si envelope corrected rugate on glass substrate. [4]
2.2 Growth processes

The study of growth processes is of interest for engineering of nanostructured thin film with functional properties, to control surface roughness, and to determine an atomistic model to simulate the growth dynamics. This chapter reviews existing models that describe the dynamic scaling behavior of surface processes. There are two general approaches to solving the dynamics of the growth processes, one is to use discrete models to approximate surface dynamics using computer simulations, the other approach is to describe the dynamic processes using stochastic differential equations as pioneered by Edwards-Wilkinson (EW) [43] and Kardar-Parisi-Zhang (KPZ) [44].

When an atom first deposits on the substrate, nucleation effects dominate and surface growth is largely dependent on the interface energy and the crystal structure of the atoms. As new atoms arrive on the coated surface, surface diffusion, atomic shadowing and other growth mechanisms take place. Atoms diffuse on the potential energy surface and bond together to form structures on an increasingly rough surface. Atoms can diffuse on the surface if they can overcome the energy barrier from within the potential well of their bonding interaction with the surface. The potential energy surface for a small number of atoms can be calculated accurately with molecular dynamics or density functional theory. As the number of atoms increase, one has to rely on approximation models to simulate the interaction potential on the surface.

An adatom is similar to an electron in that it tends to relax to a lower energy orbital, just as lower energy electron orbitals fill before higher energy orbitals. Adatoms diffuse to more stable positions with a higher numbers of interatomic bonds, and resulting in a net relaxation to defect sites. Since the energy barrier is less at defect sites
and step edges, an adatom is more likely to move into these positions. The diffusion length depends on the diffusion rate, initial energy, deposition angle, binding energy of the materials, and temperature.

The average adatom diffusion length, $\Lambda$, is given by:

$$\Lambda = \sqrt{D \cdot \tau_m}$$

(2.5)

$$D = D_o \exp \left( -\frac{E_h}{k \cdot T_f} \right)$$

(2.6)

where $\tau_m$ is the mobility lifetime (s), $D_o$ is the intrinsic diffusivity ($m^2/s$), $T_f$ is the film temperature (K), and $E_h$ is the hopping energy (J). $E_h, \tau_m, T_f$ vary depending on the location, local geometry, temperature, and chemical potential. Generally, materials with a higher melting point temperature have a higher activation energy barrier, thus lower diffusion length.

The structure zone model (SZM) provides a quantitative framework to describe nanostructured film growth [45]. There are three structured zones categorized by the ratio of the film temperature to the material melting temperature, $T_m$. This ratio gives a rough measure of the diffusion length. In zone 1, where $\frac{T}{T_m} < 0.3$, the atomic shadowing effects dominate, and the adatom diffusion effect is small, producing films with columnar structure. For zone 2, where $0.3 \leq \frac{T}{T_m} \leq 0.5$, adatom diffusion dominates over the shadowing effect, and finally in zone 3, where $\frac{T}{T_m} > 0.5$, bulk diffusion takes place, producing films that are dense and more uniform. The SZM model, however, only works for flux angle $\alpha < 50^\circ$. Since the vapor flux arriving at the substrate will have an effective temperature of the boiling point of the material, and
the substrate is kept at a constant temperature, a localized increase in temperature can increase the probability of adatom diffusion and bulk diffusion. For glancing angle deposition, the atomic shadowing effect is enhanced, and as such atomic and crystal structure effects are amplified in the surface structure on the film. Brown [46] produced copper films with a pyramidal structure that is primarily the result of the underlying crystal structure of copper. Brown and Buzea [47] produced films using various materials and diverse nanostructures such as zig zag, helix, tilted nanocolumn, cauliflower, and square spiral with the GLAD system.

The angular distribution of the impinging atoms is assumed to be gaussian distributed; this is important for a realistic simulation. Given that the separation between the source and the substrate is 31 cm [48], the typical angular spread for the GLAD deposition system is between 4° to 6°. If the impinging atoms have sufficient kinetic energy, they may desorb from the surface or cause a slight increase in the localized temperature on the surface and induce diffusion of the surrounding atoms.

Surface roughness and film density depend on the density of island nucleation. As substrate temperature increases, atomic density increases, roughness decreases and island nucleation size increases, all due to higher surface mobility.

An understanding of the atomic formation dynamics and equilibrium morphology under different conditions is essential in engineering nanostructures. Some of the processes that lead to nanostructure formation include:

- Deposition
- Diffusion across plane
- Diffusion along step edge
• Association of an adatom to a step edge

• Dissociation of an adatom from step edge

• Diffusion up or down an island

• Desorption from plane

Any attempt at producing a realistic thin film growth simulation must take into account all possible processes and the rate of these processes must be known. The following sections of this chapter describe some of these major processes and growth models associated with film growth.

2.3 Deposition model

Several discrete growth models are generally used to describe the physical deposition system [5]. The behavior that influences the growth and nucleation to the final microstructures is explained in atomistic terms. Surface processes such as adsorption, desorption, diffusion of atoms on surfaces, and binding energy between atoms are studied in detail and are adapted to the simulation algorithm.
2.3.1 Ballistic deposition

Figure 2.5: Illustration of a) Ballistic Deposition model and b) Random Relaxation model

For the ballistic deposition (BD) model, a particle sticks to the first nearest neighbour (NN) or next nearest neighbour (NNN) site along its trajectory. The difference between random relaxation (RR) and the BD model is that for the BD model, the particle sticks to the first particle it encounters, while for RR, the particle arrives on the surface first, then relaxes to a lower energy position. Figure 2.5a shows the a BD model with the NN sticking rule. Here a particle is generated at a random position above the surface, then the particle travels ballistically toward the surface and stick to the first site along its trajectory that has an occupied NN. For the RR model, particle A and B are generated randomly above the surface and are deposited on the top of the column under them. In contrast to the BD model, the height of the interface at a given point does not depend on the height of the neighboring columns.
2.4 Growth models

2.4.1 Kardar-Parisi-Zhang Growth model

The KPZ equation is the minimal Langevin equation, which is the extension to the Gaussian linear Edward-Wilkinson equation with non-linear terms describing non-conservative surface growth processes [5].

\[
\frac{\partial h (x,t)}{\partial t} = \nu \nabla^2 h + \frac{\lambda}{2} (\nabla h)^2 + \eta (x,t)
\]  

(2.7)

Where \( h(x,t) \) is a height variable at time \( t \), \( \nu \) and \( \lambda \) are constant and \( \eta \) is Gaussian white noise. The first term on the right hand side describes the surface relaxation under a surface tension \( \nu \), and the second non linear term \( \frac{\lambda}{2} (\nabla h)^2 \) represents growth normal to the surface.

2.4.2 Eden model

There are three variants of the Eden model [49]. In the first model, a particle has equal probability to deposit on any unoccupied position on the surface. In the second model, each unoccupied site adjacent to an occupied site is represented by a bond, and the final site is chosen with equal probability. In the third Eden model, a site is chosen on any occupied site on the surface with equal probability, then the new particle site is chosen on any unoccupied surface of the chosen occupied site with equal probability. Results show that the third Eden model achieves the best scaling behavior, and model 1 and 2 result in strong finite-size effects. The scaling properties are described by the KPZ equation.
2.4.3 Solid-On-Solid model

The SOS model belongs to the KPZ universality class, and it models atoms as stacks layered on top of each other with no overhangs, and places a limit on the height difference between neighboring sites. There are two types of SOS models that give accurate scaling exponents: 1) Single step model, and 2) Restricted SOS model [36] [50].

![Solid-on-solid model](image)

Figure 2.6: Solid-on-solid model [5].

For the single step model, we can obtain the scaling parameters by mapping it to the Ising model or the lattice gas model, both are described in the next chapter. The single step model starts with a grooved surface. The added particle either adsorbs at the minimum with probability $p^+$ or desorbs at the local maxima with probability $p^-$. This limits the height difference between neighboring sites to be 1. Two updating methods can be employed: 1) Sequentially, one site is chosen at random, and absorption or desorption occurs, 2) in parallel, all absorption occurs at the local minimum.
with probability $p+$ and all desorption occurs at the local maxima with probability $p-$.

The surface topology can be described using a 2D array of height at each site, $i$. The reaction site and a process is chosen randomly, and the lattice array is updated upon completion of a process.

Assuming only first NN interactions are present, there are 5 desorption/diffusion processes ($j=1..5$) and one adsorption process ($j=6$) for each site, $i$, each with distinct probabilities according to their interaction potential, $w$, given by the site’s coordination number:

\[ \Gamma_j = \nu_d e^{-\frac{jw}{kT}}, \quad j = 1..5 \]  
\[ \Gamma_6 = \nu_d e^{-\frac{3w}{kT}} e^{-\frac{\Delta \mu}{kT}} \]  

where $\nu_d$ is the vibrational frequency (typically between $10^{13} - 10^{15}$ s$^{-1}$), $T$ is the temperature, $\Delta \mu$ is the phase transition chemical potential, and $k$ is the Boltzmann factor.

### 2.4.4 Ising model

The Ising model demonstrates ferromagnetism, showing a second order phase transition from a disordered state at high temperature to an ordered state at low temperature. [51] Each spin can be in either the up or the down state. The energy of the system is determined by its nearest neighbor pairs. With the Ising model, the interface properties can be described by a set of Ising variables $[s] = [s_1, s_2, ..., s_L]$, where $s_i = h(i) - h(i-1) = \pm 1$. At $t=0$, magnetic equilibrium exists such that the spins alternate. Growth occurs if the spins are in a $(-+)$ configuration, and the spins
transition from \((-+)\) to \((++)\) with probability \(p^+\). Similarly, atoms desorb if the spin is in a \((+++)\) configuration, and the spin transitions from \((++)\) to \((-+)\) with probability \(p^-\).

The single step model can be mapped onto the lattice gas model. Surfaces with slope \((-1)\) move right to surfaces with slope \((+1)\) with probability \(p^+\), and vice versa. Consider a system with \(L/2\) particles which are placed at every other lattice site. Growth occurs with a probability \(p^+\) when a particle moves to the right, and desorption occurs with a probability \(p^-\) when it moves to the left. Only one particle can occupy each lattice site at a time, this condition is equivalent to the height difference restriction in the single step model.

The KMC method is widely used to study crystal defects, here we present a lattice Ising model of surface defect diffusion. At high temperature, surface defects are created due to high adatom vibrations, exchange diffusion occurs where interstitials may diffuse to vacancies position in the crystal. Consider interstitials \((\sigma_i = 1)\) and vacancies \((\sigma_i = 0)\) in a lattice, their mobility can be described by:

\[
r(T) = v_d e^{-\frac{\Delta J}{kT}}
\]

where \(\Delta J\) is the activation energy between two neighboring vacancies sites at a separation distance \(r = |i - i'|\), and the binding energy at site \(i\) is given by:

\[
U_{binding}(i) = \sum_{i'} J(|i - i'|)\sigma_{i'}
\]

Furthermore, the vacancies and interstitials can recombine if they come within a distance \(p\) from each other. The tight-binding molecular dynamics method was
used to study diffusion, interstitial-vacancy recombination, and point defects volume in crystalline silicon [52]. The lattice Ising model is used to simulate surface defect systems at various temperatures and activation energies. For instance, the interstitials vs. vacancies jump ratio and the number of recombinations that occur can be simulated.

2.5 Surface diffusion model

To model diffusion, it is best to consider the potential energy surface. In order for the atoms to move, atoms must have sufficient energy to surpass an activation barrier, and the barrier energy depends on the type and nature of the atoms. As an adatom diffuses across the surface, it will be subjected to a periodic variation in potential; the magnitude of the periodic variation is called the surface diffusion energy. The SNS-KMC bond counting model assumes that only nearest neighbor interaction exists, the activation energy is calculated by:

\[ E = E_o + (CN) \cdot E_B \]  

(2.12)

Where \( E_o \) is the terrace diffusion barrier, \( CN \) is the coordination number or the number of nearest neighbor bonds in the initial position, and \( E_B \) is the binding energy of the crystal. SNS-KMC integrates a diffusion model that considers the difference in diffusion energy between the initial and final position. The diffusion rate depends on the number of bonds in the initial and final positions. If the final position has a smaller coordination number than the initial position, the diffusion rate is decreased by multiplying the rate by \( e^{0.13(\text{CN}_{\text{final}} - \text{CN}_{\text{initial}})} \) [8], where the energy difference of
0.13 \cdot \Delta CN is approximated from MD calculation for Al [53]. Ye and Hu’s [54] model assumes that atoms will diffuse to the lowest energy position in the nearest neighbor site. Zhang et al [55] assume that as the atom diffuses on the surface, it jumps to a random site according to its rate, furthermore the atoms will be stacked on if there are more than three atoms around the next jump site. Additionally if there are more than three nearest atoms at the next jump site, the atom will move up; otherwise, it will continue to diffuse to random positions until it locates a site with more than three nearest neighbor atoms.

When an adatom arrives at the surface at an oblique angle, the momentum along the surface plane is conserved and the adatom diffuses on the surface until it has exhausted its kinetic energy and relaxes to its final position. The diffusion length and path depend on the atom’s binding energy with the surface and the state of the atom before arriving at the surface [56].

The model used in the SNS-MC simulator can simulate the final structure with reasonable accuracy. In order to gain a deeper understanding of surface growth dynamics, one can obtain a physical time scale for the simulated atomic processes using KMC method. The diffusion model can be further improved by constructing a detailed catalogue for all the process rates in the system. The current diffusion model in SNS-MC considers the initial and final position of the atoms. If the final position is not occupied, and if its hopping probability is greater than a random number, the atoms will relax to that position. An extended model was developed by Claassens et al. [1], where the processes are catalogued in terms of the initial and final state of the atom. Table 2.1 shows the 16 major diffusion processes by the number of nearest neighbors in the initial and final positions, represented by $N_i$ and $N_f$ respectively. It
is noted that Schwoebel jumps refer to the event of particle diffusing up or down an island edge, therefore it is not applicable to show the initial and final coordination number.

<table>
<thead>
<tr>
<th>Process</th>
<th>$N_i$</th>
<th>$N_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step diffusion</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Dimer association</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Dimer dissociation</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Dimer-to-kink diffusion</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Step-to-kink diffusion</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Kink-to-step diffusion</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Corner dissociation</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Corner association</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Step dissociation</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Step association</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Kink dissociation</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Kink association</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Step-to-corner diffusion</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Corner-to-step diffusion</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Terrance diffusion</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Schwoebel Jumps</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Each transition represents a different process that depends on the number of bonds a diffusing atom has in its initial and final positions. The activation energy for each of the processes in Table 2.2 is calculated using Transition State Theory (TST). Xu et al. [57] performed KMC simulation of Pd clusters on MgO (100) using activation energies calculated by DFT. The energy of a large number of atomic arrangement scenarios were calculated in order to map out the fundamental energy landscape as accurately as possible and to determine which process is kinetically dominant. The DFT calculation of the activation barrier shows that the magnitude of the diffusion barrier varies significantly depending on the specific type of diffusion
that the atom undergoes. A simple linear model based on the coordination number, or the number of atoms in the nearest neighbor (NN) sites, is only suitable as a first approximation, more complex interactions can be introduced by including next nearest neighbor potential contributions.

The heuristic approach to calculate the activation energy based on the local environment has been met with some success [58]. The self learning KMC algorithm uses a shell scheme, in which a central atom is identified and the next 3 closest shells surrounding the central atoms are active. The central atom moves to a neighboring vacancy while all atoms in the 3 surrounding shells participate in the process. It is successful in predicting the most dominant kinetic processes for Cu clusters on Cu (111) as compared to standard KMC simulations. However, the frequencies differ by an order of magnitude.

<table>
<thead>
<tr>
<th>Process</th>
<th>ΔE on Au(111) (eV)</th>
<th>ΔE on Graphite (eV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 → 0</td>
<td>0.47</td>
<td>0.29</td>
</tr>
<tr>
<td>1 → 1</td>
<td>0.34</td>
<td>0.16</td>
</tr>
<tr>
<td>1 → 2</td>
<td>-0.05</td>
<td>-0.1</td>
</tr>
<tr>
<td>1 → 3</td>
<td>0.01</td>
<td>-0.28</td>
</tr>
<tr>
<td>2 → 0</td>
<td>0.57</td>
<td>0.55</td>
</tr>
<tr>
<td>2 → 1</td>
<td>0.72</td>
<td>0.41</td>
</tr>
<tr>
<td>2 → 2</td>
<td>0.34</td>
<td>0.16</td>
</tr>
<tr>
<td>2 → 3</td>
<td>0.25</td>
<td>-0.02</td>
</tr>
<tr>
<td>3 → 0</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>3 → 1</td>
<td>0.73</td>
<td>0.59</td>
</tr>
<tr>
<td>3 → 2</td>
<td>0.43</td>
<td>0.34</td>
</tr>
<tr>
<td>3 → 3</td>
<td>0.34</td>
<td>0.16</td>
</tr>
</tbody>
</table>
2.6 Desorption model

The desorption rate depends on two major factors: 1) The type of atoms and how strongly the atom is bonded to the crystal surface, and 2) local geometry. The adatom desorption rate is given by $\nu_a e^{-E_a/kT}$, where $\nu_a$ is the vibrational frequency of the atom, and $E_a$ is the characteristic desorption energy required to release an atom from the surface. The adatom lifetime before desorption is given by $\tau_a = \nu_a^{-1} e^{-E_a/kT}$. Desorption events are not significant except at elevated temperature or in the case of sputtering. For Pd deposition on MgO, it was determined from KMC simulations that even at 600K the desorption rate is only a fraction of the lowest diffusion rate [57].

2.7 Atomic bindings

The nucleation and growth structure depend strongly on the type of deposition material and the substrate. The layer-by-layer model, also called Frank-van der Merwe (FM) model [59], describes layered films in terms of their binding energy. Consider the deposition of material A onto a substrate with material B, if $\gamma_A < \gamma_B + \gamma_*$, where $\gamma_*$ is the interface energy, $\gamma_A$ and $\gamma_B$ are the binding energy for material A and B, respectively, then the film will have a layered structure due to the higher binding energy between the deposition atoms and the substrate. Films with dominant island growth arise because the deposited atoms are more strongly bound between themselves than to the substrate. The island growth model is also known as the Volmer-Weber model (VW) [60]. The layer-plus-island model, or Stranski-Krastanov (SK) model [61], is an intermediate version of the layer-by-layer and island growth model, where layering appears first and island growth is dominant in the later stage due to the increase in
binding energy as thickness increases.

To simplify the binding energy for all cluster sizes, we assumed the pair binding model in our SNS simulator, in which the binding energy of cluster size, \( j \), is given by \( E_j = b_j E_b \), where \( b_j \) is the number of lateral bonds in the cluster, and \( E_b \) is the binding energy between a pair of atoms.
Chapter 3

Physical Structure Simulation

A combination of kinetic Monte Carlo, molecular dynamics, tight-binding, transition state theory, and ab initio density functional simulations can be used to tackle the problem of gaining a full understanding of the bonding interactions of molecular growth on a variety of substrate surfaces. These methods are complementary to each other. For instance, density functional simulations can derive accurate adsorption energies and binding energies of the atoms, and these can be used as inputs for KMC simulations to understand the underlying processes of nanostructured growth under varying deposition conditions on a macroscopic scale. The KMC simulations then provide a dynamic representation of the film structures, and the simulated structures are used as inputs to FDTD simulations to calculate the optical properties of the films.

The choice of simulation method is not only driven by the accuracy of the results, but also by the length and time scale of the physical system. There are four broad categories for length scale: Electronic and Atomic (∼10⁻⁹m), Microscopic (∼10⁻⁶m),
Mesoscopic ($\sim 10^{-4}\text{m}$), and Macroscopic ($\sim 10^{-2}\text{m}$). Time scale can range from femtosecond to millisecond. Specific methods are more advantageous at certain length and time scales. At the atomic scale and time scale in the femtosecond region, Density Functional Theory (DFT) [62] and Quantum Monte Carlo (QMC) [63] must be used. At the microscopic scale, one would consider using Molecular Dynamics (MD) [64] [65], Kinetic Monte Carlo (KMC) or Monte Carlo (MC) [1]. At the macroscopic scale, one can use Rate Equation analysis (RE). DFT utilizes a quantum mechanical approach to solve the Schrodinger equation that describes the interaction between nuclei and electrons in the system. Approximation schemes such as Hartree-Fock and Born-Oppenheimer are then used to solve the many body problems.

Instead of solving the computationally intensive Schrodinger equation, molecular dynamics derives the equilibrium states of a system based on the Newtonian equation. Given the interaction potential between atoms, the evolution of the system is revealed by numerically solving for the forces between atoms undergoing a small time increment $\delta t$, and the system is then allowed to eventually relax to an equilibrium condition. However, MD is only suitable for simulations of a few thousands atoms. For GLAD nanostructure, where we wish to simulate thin films that are up to a micrometer thick, and that have adatom diffusion rates in the microsecond regime under room temperature, the KMC method should be chosen based on the length and time scale requirement. The KMC method can simulate the evolution of nanostructure growth and its resulting properties with reasonable precision.

The Monte Carlo algorithm simulates physical systems by using a stochastic method to examine all the possible states of the system. The possible system configurations are achieved through transitions with equal probabilities. KMC takes into
account the energy barrier, or the probabilities to access between possible states that
govern the evolution of a system. This is especially important in modeling surface
diffusion, where the diffusion interaction depends on the local bonding environment.
The use of transition rates translates to a real time scale that can be used to charac-
terize the system, for example, to determine the diffusion constant.

3.1 Metropolis Monte Carlo algorithm

The Metropolis MC method is an effective numerical simulation method that is used
to solve complex and multidimensional problems. The MC algorithm chooses a system
configuration at random. For example, consider a system that is at state, $I$, the
algorithm chooses a random NN site, $j$, and proposes to diffuse a particle particle
to that site. The proposed configuration, $J$, is the state where that particle diffuses
from position $i$ to position $j$. The transition probability is based on the Boltzmann
equation:

$$P_{I ightarrow J} = \begin{cases} 
    e \left( -\frac{\Delta E_{IJ}}{kT} \right) & \Delta E_{IJ} > 0 \\
    1 & \Delta E_{IJ} \leq 0 
\end{cases}$$ (3.1)

where $P$ is the transition probability, and $\Delta E$ is the energy difference between the
initial and final state. The Metropolis MC is concerned only with the initial and
final states of the system, and provides no information about the dynamic evolution
leading to the final state. For the Metropolis algorithm, the trial state is accepted
if $\Delta E \leq 0$. If $\Delta E > 0$, the trial state is accepted if $u < e \left( -\frac{\Delta E_{IJ}}{kT} \right)$, where $u$ is
a random number with a specific distribution. This process is repeated until the
system has reached thermodynamic equilibrium. The KMC algorithm incorporates the Metropolis MC scheme by time stepping through the selection process. Consider the rate equation for the forward and backward transition rate:

\[ r_{I ightarrow J} = A_{IJ} e^{-\frac{E_{IJ}}{kT}} \]  

\[ r_{J ightarrow I} = A_{IJ} e^{-\frac{E_{IJ} - \Delta G_{IJ}}{kT}} \]

The probability of choosing a particular process depends on its rate, such that:

\[ \sum_{i=1}^{k} \frac{r_i}{R} < u < \sum_{i=2}^{k-1} \frac{r_i}{R} \]

where \( u = e^{-R\Delta t} \) a random number uniformly distributed between [0, 1], and \( R \) is the sum of the rates. Once a process is chosen, the time is incremented by:

\[ \Delta t = -\frac{\ln(u)}{R} \]

Thus the time increment between processes is stochastic and depends on the rates of all available processes.

### 3.2 Monte Carlo Simulation

Suzuki et al. developed a 3D Virtual Film Growth Simulator (VFIGS) [66] to model growth of thin films with Monte Carlo ballistic deposition. This simulator enables a deeper understanding of the ballistic deposition, surface diffusion and other underlying processes that occur in real films. The SNS simulator presented in this thesis was
developed from VFIGS, and adapted specifically for the GLAD deposition technique. The SNS simulator was used to study the effect on film properties and morphology under varying deposition condition and diffusion models, and the simulation results were compared with experiments. SNS is powerful in that an extensive list of deposition parameters can be incorporated into the simulation, including control of tilt rate, $\alpha'$, rotation rate, $\phi'$, (by angle, phase, wave form, and height), custom pre-patterened substrate, alpha and phi dispersion, number of periodic layers, temperature, and energy barriers.

The KMC algorithm used in SNS-KMC will be discussed in more detail in the proceeding chapters. In summary, SNS-MC follows the ballistic deposition model, where a particle is randomly generated above the substrate, and the particle follows a straight line trajectory until it strikes the surface of the film. This is done by constantly checking the immediate region surrounding the particle. If this region contains another particle, the hopping probabilities to the immediate region are calculated, and its final position is determined by a random walk algorithm. In addition, the particle is not allowed to move to an occupied position.

SNS-MC utilizes the Metropolis and random walk algorithm to determine whether the particle moves to a particular position. For instance, let us consider a system of N interacting particles, the interaction between them is given by a potential function $U(r)$. Next, consider a random particle; we choose a random displacement $r'$ for this particle, such that the potential energy difference is $\Delta U = U(r') - U(r)$. To determine whether this particle will displace to the new position, a random number, $q$, is generated and compared to the Boltzmann factor. If $q < \exp^{-\frac{\Delta U}{k_B T}}$, the particle will move to the new position, otherwise, it will continue to undergo the random walk.
algorithm \( S \) times, where \( S \) is the number of random walk steps, until the condition is satisfied, and its final resting position is determined. The number of random walk steps is related to the surface diffusion length, which is a function of many deposition parameters such as substrate temperature, source and substrate material, chamber pressure, and deposition rate. The current SNS simulator must be calibrated with experimental results to determine the random walk steps, \( S \).

A simplified process tree for SNS is presented as follow:

1) Initialization (Read input parameter file: see Appendix A2 for an sample input file)

2) Deposition (Particle generation and trajectories determination based on deposition angle and phase)

3) Ballistic deposition (Particles follow trajectories until a neighboring atom is found in its immediate region)

4) Surface diffusion (Particle undergoes random walk, resting position based on hopping probability)

5) Desorption

6) Deposit particle (Save particle position), go to step 2

7) Output analysis data

The simulated films are shown with top and cross sectional views. The depth of deposited particles in the \( x-z \) plane for the surface view is represented by a gray scale \( 0 \leq z \leq 256 \), and the depth of the cross sectional view by \( 0 \leq y \leq 256 \). SNS requires the use of a visualization program to generate the surface and cross sectional image of the simulated films. We developed a visualizer for Windows using the GrWin visualization library [67], and for Mac we developed a Yorick visualization and film
3.3 Kinetic Monte Carlo Simulation

3.3.1 Introduction

The PVD process starts with the heating of the source, generation of vapor, interaction of the vapor atoms with any foreign atoms, interaction of the vapor with the substrate or surface atoms, diffusion of adatoms on the surface, and desorption of adatoms from the surface. Adatom diffusion occurs simultaneously with the deposition of atoms, where the process rates depend on the deposition condition as well as the material being deposited. The diffusion rate is correlated with the binding energy between the atoms as well as its geometry. A diffusion model that most accurately approximates the complex potential energy surface of the film is the focus of simulation research today.

There are several challenges for KMC growth simulations. One of the major challenges is to create a complete rate catalog of all possible surface processes with their transition probabilities. The transition probabilities can be derived from density functional theory, molecular dynamics, and transition state theory for atomic length scales. Significant research is being done to refine the approximation of the potential energy surface using semi-empirical and empirical methods such as tight binding method, interaction potentials, embedded atom methods, and effective medium theory.

Another major challenge is the time scale separation between competing processes, this can be overcome by probability weighted MC to rescale the probability
of the specific surface reactions. Consider the probability of a set of surface reaction, \( x_1, x_2, ..., x_n \), probability weighted MC adjusted the probability such that, \( x_1w_1, x_2w_2, ..., x_nw_n \), where \( w_n \) is the weight for each transition probability based on importance sampling.

### 3.3.2 Atomistic model

The dynamics of the surface processes are described by the rates of the allowed processes. The diffusion and desorption energy barrier can be obtained accurately from ab initio or Molecular Dynamics (MD) calculations. Since the energy barriers are changed based on the atomic arrangement in the lattice, all the respective diffusion rates have to be calculated. Consider a 3D cubic lattice, if the adatom is only allowed to diffuse to its nearest neighbour site, and the rate depends only on the nearest neighbors, then there are \( 2^6 = 64 \) different diffusion rates for each possible configuration that have to be calculated. A simplified KMC bond counting model has been adopted in the SNS-KMC simulator to determine the energy barriers and the resulting diffusion rates. The model assumes that the energy barrier is due to the sum of the contributions from individual atoms in the nearest neighbor sites, and that \( E_i \) varies linearly with the coordination number, as shown in Equation 2.12. Particles are deposited onto a custom substrate with a constant rate. The diffusion rate for an interstitial atom is modeled as a Boltzmann distribution:

\[
r_i(T) = \nu_o e^{-\frac{\Delta E}{kT}}
\]

where \( \nu_o \) is the vibrational frequency of the interstitial atom, \( \Delta E \) is the change in activation energy, \( k \) is the Boltzmann’s constant and \( T \) is the temperature.
3.3.3 KMC algorithm

The accuracy of simulation is directly dependent on the assumptions of the identified processes, rates, and algorithm used to model the deposition process. SNS-MC considers only one adatom as mobile at any given time, and each adatom follows the same sequential process of ballistic deposition, diffusion and relaxation. After this, the adatom is considered immobile and the next adatom begins its deposition process. The diffusion model in SNS-MC takes into account the energy in the initial and final position, however, the local height does not play a role in determination of the diffusion process.

In contrast, SNS-KMC considers the entire system during the simulation. All atoms and processes can occur at any given time, deriving a much more accurate model of the actual film growth process. The three main processes, ballistic deposition, diffusion and desorption, that were described above, along with 15 sub models in the diffusion process, are needed to take into account the different possible diffusion scenarios. The deposition rate, $F$, is defined as atoms per second, and is dependent on the tilt angle, $\alpha$. The time between deposition is given by

$$\Delta t = \frac{1}{F}$$  \hspace{1cm} (3.7)

During this time, all adatoms on the surface are mobile, therefore any specific adatom can either diffuse or desorb from the surface. The probability of a process to occur is dependent on its weighted probability:

$$p_i = \frac{r_i}{R}$$  \hspace{1cm} (3.8)
where $r_i$ is the rate of the specific process and $R$ is the total rate of the system of $k$ processes,

$$R = \sum_{i=1}^{k} r_i$$  \hspace{1cm} (3.9)

A process is chosen randomly according to

$$\sum_{i=1}^{k} r_i < u < \sum_{i=2}^{k-1} r_i$$  \hspace{1cm} (3.10)

where $u = e^{-R\tau}$ a random number uniformly distributed between $[0, 1]$. The system time for the selected process is given by:

$$\Delta t = -\frac{\ln(u)}{R}$$  \hspace{1cm} (3.11)

When $\Delta t$ exceeds the deposition time, the next particle will be deposited. The probability of which process will be chosen is determined by the relative rates.

The SNS-KMC algorithm begins with an atom generated at a random position above the surface, traveling in a ballistic trajectory, and arriving at the substrate at an angle, $\alpha$, to the normal. Each atom has several processes that it can undergo, the rates for each atom are calculated, and the selection of the specific atom and process is chosen according to equation 3.10. The time of the specific process is used to update the total system. If $\tau < \Delta t$, then the chosen process is executed and the selection process continues until the deposition time runs out.

The SNS-KMC algorithm is summarized below:

1. Initiate deposition condition (flux, angle, rates, temperature, substrate geometry)
2. Calculate deposition time, $\Delta t$

3. Generate random number and deposit atoms

4. Calculate sum of rates of all processes for each atoms

5. Randomly select a process to be executed

6. Calculate process lapse time and add to system time

7. Repeat until system time is reached or maximum number of atoms has been deposited.

The 3D lattice is represented by a 1D array of square cells, specifying the position, type and the rate of each atom. The 3D array is mapped to a 1D array using this formula: $(i,j,k)=i*\text{length}*\text{width} + j*\text{width} + k$, where length and width is the dimension in the x and y plane. This array conversion is done to minimize memory allocation and reduce memory access.

### 3.4 Surface properties

The morphological features of the film are influenced by atomic mobility, grain boundaries, crystal structure, atomic shadowing, and other factors due to the deposition conditions. There are three main types of growth models that are studied extensively[68]:

- The first type is called the Frank-van der Merwe (FM) mode, which is a layer by layer growth model that occurs in cases where diffusion is dominant;
- the second type is called the Volmer-Weber (VW) mode, which is characterized by a predominant island growth on the surface;
- the last is the Stranski-Krastanov (SK) mode, where a combination of FM and VW growth is seen on the film.
3.4.1 Density, roughness, fractal dimension

The SNS simulator generates the packing density of the simulated thin film which can be compared with gravimetric and XRR measurement. Surfaces are generally characterized by the Root Mean Square (RMS) roughness, skewness, and fractal dimension [5].

The RMS roughness is defined as:

$$w(L, t) = \sqrt{\frac{1}{L} \sum_{i=1}^{L} [h(i, t) - \bar{h}(t)]^2} \quad (3.12)$$

where $L$ is the system size and $h$ is the height of the interface. The RMS roughness is calculated at a specific interval to measure the evolution of the roughening process as a function of time. The interface roughness follows a power law scaling according to

$$w(L, t) \sim t^\beta \quad (3.13)$$

where $\beta$ is the growth exponent that characterizes the time-dependent dynamics of the roughening process. The interface roughness follows a power law increase until it reaches a saturation value, $w_{sat}$, which also follows a power law scaling,

$$w_{sat}(L) \sim L^\alpha \quad (3.14)$$

where $\alpha$ is the roughness exponent. The RMS roughness in Equation 3.12 by itself cannot completely characterize the surface because the polarity of the height value relative to the mean height is indistinguishable. The skewness can describe the asymmetry of the height distribution about the mean:
Fractal dimension is a statistical quantity to measure the self-similarity of the system. The Box Counting method [5] has been used to determine the fractal dimension of the 3D structure, $S$, and the fractal dimension, $D$, is given by

$$D_b = \lim_{h \to 0} \frac{\log N_b(k_b)}{\log (k_b)}$$

(3.16)

where $N_b(k_b)$ is the minimum number of $n$-dimensional cubes of size length $k_b$ needed to cover the set $S$. The fractal dimension, $D_b$, corresponds to the slope of the plot of $\log N_b(k_b)$ vs. $\log (k_b)$. It is noted that only the sequence in which $k_b$ converges to zero in the set $S$ needs to be considered, in the limit of $\log N_b(k_b) / \log (k_b)$. It is convenient to choose a sequence in which every $h$ converges in the set $S$, in this case we chose the powers of two sequence as the width size, such that $k_b = 2^0, 2^1, 2^2, \ldots, 2^p$. Where $p$ is an integer such that $2^p \leq \text{maxDims}(c)$, where maxDims($c$) is the maximum dimension of the lattice array. If any dimensions of the lattice array is not in the powers of two, the lattice array is padded with zeros to size $2^{p+1}$ over each dimensions. The box counting method then counts the number of 3D boxes of size $k_b$ needed to cover the space, and the fractal dimension is determined using equation 3.16.
Chapter 4

Electromagnetic FDTD Simulations

4.1 Introduction

Computer simulation of optical response of thin films will not only enhance our understanding of the optical, electrical, and magnetic properties of films under various deposition conditions, but also will provide a tool to optimize the optical response given stringent design parameters for engineering film properties. The Finite-Difference Time-Domain (FDTD) method has been proven to be accurate and robust for simulating the electromagnetic response of nanostructured thin film by numerically solving the time-dependent Maxwell’s equations.

The motivation for correctly predicting the optical response is due to the needs to engineer GLAD films with specific properties. Furthermore, the fabrication of nanostructured thin films is expensive and time consuming, thus it is not feasible to attempt the trial and error approach for designing thin film structure with a particular
Effective Medium Approximation (EMA) computes the optical properties of a complex medium by assuming layers of homogeneous mediums [69]. Consider a heterogeneous mixture of two materials. For a columnar GLAD film, the column and void are represented by their corresponding dielectric constants, $\epsilon_A$ and $\epsilon_B$. The Bruggeman approximation calculates the effective medium dielectric constant, $\epsilon_{\text{eff}}$, as:

$$\frac{\epsilon_A - \epsilon_B}{\epsilon_{\text{eff}} + L_B (\epsilon_A - \epsilon_{\text{eff}})} p_A + \frac{\epsilon_B - \epsilon_{\text{eff}}}{\epsilon_{\text{eff}} + L_B (\epsilon_B - \epsilon_{\text{eff}})} (1 - p_A) = 0$$

where $p_A$ is the filling factor of material A, and $L_A$ and $L_B$ are the depolarization tensors of material A and B that take into account the shape of the columns. However, the measured optical response yields inaccurate results as compared to the theory, since the exact geometrical arrangement is unknown and the effective layers are assumed to be symmetrical.

A numerical analysis method such as FDTD is proven to be robust when applied to Maxwell’s equations for simulating optical response. There are nearly no restrictions on the geometrical structure, its disadvantage being that it is computationally intensive and requires the use of supercomputing resources to carry out parallel computations.

FDTD divides space and time into a uniform grid of discrete points and approximate the derivatives of Maxwell’s equations by finite difference. The propagation in time uses a leap frog scheme where the E field at time $t$ is computed from the E field at time $t-\delta t$, and the H field at time $t-\delta t/2$, and vice versa for the H field. In this
way, the E and H field patterns are marched through time, offset by half of a time step, \( dt \). This method employs a special Yee grid in which different components of each vector are associated with different locations on the grid cell.

The Time Domain method can compute the response of a linear system at many frequencies with a single computation. This is done by sending a short pulse with a broad bandwidth into the structure and the resulting \( E(t) \) and \( H(t) \) fields are then Fourier transformed to yield \( E(\omega) \) and \( H(\omega) \), where the fields are obtained at each frequency component, \( \omega \). Because the computational region is finite, perfectly matched layers are used to simulate open boundaries.

## 4.2 Finite difference formulation

In this chapter an overview of the finite difference method, and the FDTD formulation of Maxwell’s equations, are presented. We will present the Yee algorithm [70] for applying the FDTD method to Maxwell’s equations in a linear, non-dispersive, non-magnetic medium. Considerations such as numerical stability, dispersion, and the boundary conditions with a Perfectly Matched Layer (PML) will be discussed. In addition, software design implementation and Matlab programs for modeling the 1-D wave propagation and 1-D FDTD method will be shown in Chapter 7.

Consider the one-dimensional scalar wave equation:

\[
\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}
\]

(4.2)

where \( u \) is a scalar function, \( c \) is a constant equal to the propagation speed of the wave. The solution of this scalar wave equation is a summation of two arbitrary
functions $F$ and $G$:

$$u(x, t) = F(x + ct) + G(x - ct) \quad (4.3)$$

representing propagating waves in the +x and -x directions. Finite difference is an expression of the form $\Delta f(x) = \frac{[f(x+h) - f(x-h)]}{2}$, which is also known as the central difference equation. Consider the Taylor series expression of $u(x + \Delta x)$:

$$u(x + \Delta x) = u + \Delta x \left( \frac{\partial u}{\partial x} \right) + \frac{(\Delta x)^2}{2} \left( \frac{\partial^2 u}{\partial x^2} \right) + \frac{(\Delta x)^3}{6} \left( \frac{\partial^3 u}{\partial x^3} \right) + ... \quad (4.4)$$

Similarly, for $u(x - \Delta x)$:

$$u(x - \Delta x) = u - \Delta x \left( \frac{\partial u}{\partial x} \right) + \frac{(\Delta x)^2}{2} \left( \frac{\partial^2 u}{\partial x^2} \right) - \frac{(\Delta x)^3}{6} \left( \frac{\partial^3 u}{\partial x^3} \right) + ... \quad (4.5)$$

Adding equation 4.4 and equation 4.5, and solving for $\frac{\partial^2 u}{\partial x^2}$, we obtain:

$$\frac{\partial^2 u}{\partial x^2} = \left[ \frac{u(x + \Delta x) - 2u(x) + u(x - \Delta x)}{(\Delta x)^2} \right] + O[(\Delta x)^2] \quad (4.6)$$

where $O[(\Delta x)^2]$ is the truncation error for the Taylor series. It approaches zero as the square of the space increment. Equation 4.6 is the second order accurate, central difference approximation, to the second partial derivative of $u$.

Let $u(x)$ be defined at discrete points along the real axis at finite intervals. Let us define the space interval, $i$, and a time interval, $n$, such that $u^n_i = (x_i, t_n)$ denotes a field at space point $x_i = i \cdot \Delta x$, and time point $t_n = n \cdot \Delta t$. Thus equation 4.6 can be rewritten as:
\[ \frac{\partial^2 u}{\partial x^2} = \left[ \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{(\Delta x)^2} \right] + O \left[ (\Delta x)^2 \right] \quad (4.7) \]

Similarly, for the partial derivative of \( t \):

\[ \frac{\partial^2 u}{\partial t^2} = \left[ \frac{u_{i}^{n+1} - 2u_i^n + u_{i}^{n-1}}{(\Delta t)^2} \right] + O \left[ (\Delta t)^2 \right] \quad (4.8) \]

Substituting equation (4.7) and equation (4.8) into the scalar wave equation, we obtain the finite difference equation of the scalar wave equation:

\[ u_{i}^{n+1} = (c\Delta t)^2 \left[ \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{(\Delta x)^2} \right] + 2u_i^n - u_{i}^{n-1} \quad (4.9) \]

An initial condition and a boundary condition need to be specified in order to obtain a unique solution for the partial differential equation (4.2) (ex. \( u_m^0 = f_1(n) \) and \( u_m^n = f_2(n) \)). Then for all subsequent field points, \( u_m^n \) for \( n = 0,1,2,3,\ldots, N \), and \( m=1,2,3,\ldots, M \), where \( N \), and \( M \) is the total number of time steps and grid points, respectively; can be computed simply from the previous time-step of \( u \).

### 4.3 Finite difference schemes

Let \( u_m^n = U(m \cdot \Delta x, n \cdot \Delta t) \), where \( n \) is the time step, and \( m \) is the position step. The discretizations are \( \Delta x \) and \( \Delta t \). The idea of the finite difference scheme is to approximate partial derivatives by a finite difference operator. Six FDTD implementation schemes are shown below.
\[
\frac{u_{m}^{n+1} - u_{m}^{n}}{\Delta t} + a \frac{u_{m+1}^{n+1} - u_{m}^{n}}{\Delta x} = 0 \quad \text{Forward-time forward space (4.10)}
\]

\[
\frac{u_{m}^{n+1} - u_{m}^{n}}{\Delta t} + a \frac{u_{m}^{n} - u_{m-1}^{n}}{\Delta x} = 0 \quad \text{Forward-time backward space (4.11)}
\]

\[
\frac{u_{m}^{n+1} - u_{m}^{n}}{\Delta t} + a \frac{u_{m+1}^{n} - u_{m-1}^{n}}{2\Delta x} = 0 \quad \text{Forward-time central space (4.12)}
\]

\[
\frac{u_{m}^{n+1} - u_{m}^{n-1}}{2\Delta t} + a \frac{u_{m+1}^{n} - u_{m-1}^{n}}{2\Delta x} = 0 \quad \text{Leapfrog scheme (4.13)}
\]

\[
\frac{u_{m}^{n+1} - \frac{1}{2}(u_{m+1}^{n} + u_{m-1}^{n})}{\Delta t} + a \frac{u_{m+1}^{n} - u_{m-1}^{n}}{\Delta x} = 0 \quad \text{Lax-Friedrich scheme (4.14)}
\]

\[
(2 - 2r) u_{m}^{n+1} - r (u_{m+1}^{n+1} + u_{m-1}^{n+1}) = (2 - 2r) u_{m}^{n} - r (u_{m+1}^{n} + u_{m-1}^{n})
\]

Crank-Nicholson scheme (4.15)

These schemes express \(u_{m}^{n+1}\) as a linear combination of \(u_{m}^{n}\) and \(u_{m}^{n-1}\). Equations 4.10 to 4.14 are explicit schemes where the value of \(u_{m}^{n}\) can be found directly from the previous time step, \(u_{m}^{n}\) and \(u_{m}^{n-1}\). Equation 4.15 is known as an implicit one-step scheme, since only \(u_{m}^{n}\) is required to find \(u_{m}^{n+1}\). In Strikwerda [71], the first 5 FDTD schemes are shown to be unstable using Von Neumann Analysis, while the Crank-Nicholson scheme is unconditionally stable given finite \(\Delta x\) and \(\Delta t\). However, this scheme requires a very large system of linear equations to be solved for each time step.
4.3.1 Convergence

The error at a certain instance in time is defined as:

$$
\epsilon = \max_m |U^n_m - U(m \cdot \Delta x, n \cdot \Delta t)|
$$

The scheme is convergent if \( \epsilon \to 0 \) as \( \Delta x, \Delta t \to 0 \) for all \( n \). For a convergent scheme, the FDTD solution must converge rapidly toward the exact solution as \( \Delta x, \Delta t \to 0 \). \( U^n_m \) must be bounded as \( \Delta t \to 0 \) and \( n \to \infty \).

4.3.2 Stability

The FDTD scheme is stable if for \( T \geq 0 \), there exists a constant such that

$$
|U^n_m| \leq C_T \sum_{j=0}^{J} ||U^j_m||, \ 0 \leq n \cdot \Delta t \leq T
$$

where \( |U^n_m| \leq [\Delta x \sum_m |U^n_m|^2]^{\frac{1}{2}} \), and \( J \) is the number of time steps used in the scheme subtracted by 1. Equation 4.17 shows that if \( T \) is finite, \( C_T \) will be finite and the value will be bounded at all time steps.

4.4 Maxwell’s equations

The differential form of Maxwell’s equations is as follow:
CHAPTER 4. ELECTROMAGNETIC FDTD SIMULATIONS

\[ \frac{\partial \vec{B}}{\partial t} = -\nabla \times \vec{E} - \vec{M} \quad (4.18) \]

\[ \frac{\partial \vec{D}}{\partial t} = -\nabla \times \vec{H} - \vec{J} \quad (4.19) \]

\[ \nabla \cdot \vec{D} = \rho \quad (4.20) \]

\[ \nabla \cdot \vec{B} = 0 \quad (4.21) \]

Where \( \vec{E} \) is the electric field (V/m), \( \vec{D} \) is the electric flux density (coulombs/m\(^2\)), \( \vec{H} \) is the magnetic field (A/m), \( \vec{B} \) is the magnetic flux density (webers/m\(^2\)), \( \vec{J} \) is the electron current density (A/m\(^2\)), \( \vec{M} \) is the equivalent magnetic current density (V/m\(^2\)), \( \rho \) is the total charge density (coulombs/m\(^3\)). For linear, isotropic and non-dispersive materials,

\[ \vec{D} = \epsilon \cdot \vec{E} = \epsilon_r \epsilon_o \vec{E} \quad (4.22) \]

\[ \vec{B} = \mu \cdot \vec{H} = \mu_r \mu_o \vec{H} \quad (4.23) \]

Where \( \epsilon \) is the electrical permittivity (farads/m), \( \epsilon_r \) is the relative permittivity, \( \epsilon_o \) is the free space permittivity (8.854x10\(^{-12}\) farads/m), \( \mu \) is the magnetic permeability (henrys/m), and \( \mu_r \) is the relative permeability (dimensionless scalar), and \( \mu_o \) is the free space permeability (4\( \pi \times 10^{-7} \) henrys/m).

For linear, isotropic, nondispersive, lossy materials, Maxwell’s equations become,

\[ \frac{\partial \vec{H}}{\partial t} = -\frac{1}{\mu} \nabla \times \vec{E} - \frac{1}{\mu} \left( \vec{M}_{source} + \sigma^* \vec{H} \right) \quad (4.24) \]

\[ \frac{\partial \vec{E}}{\partial t} = -\frac{1}{\epsilon} \nabla \times \vec{H} - \frac{1}{\epsilon} \left( \vec{J}_{source} + \sigma \vec{E} \right) \quad (4.25) \]
where $\sigma$ is the electric conductivity (siemens/m), $\sigma^*$ is the equivalent magnetic loss (ohms/m), $\vec{M}_{\text{source}}$ is the source of the magnetic field, and $\vec{J}_{\text{source}}$ is the source of the electric field.

These two coupled differential equations can then be resolved into components for simpler calculation. Clearly, the time derivative of the H-field is dependent on the curl of the E-field and the stored value of the H-field, and similarly for the E-field. The FDTD method numerically solves these two equations by sequentially time stepping to solve for the E-field and H-field, yielding an approximation of the partial differential equation.

\section*{4.5 Finite-Difference expression for Maxwell’s equations in three dimensions with Yee lattice}

The Yee algorithm [70] is commonly used to solve FDTD equations. Figure 4.1 shows the location of the H-field and E-field for a unit cell in the computational region. In the Yee cube, each H-field component is surrounded by four E-field components, and similarly, each E-field component is surrounded by four H-field components.

For simplicity, let us assume our model is a linear, isotropic, non-dispersive dielectric with no magnetic charge or magnetic current ($\rho = 0, M = 0$). Maxwell’s curl equation becomes:

\begin{align}
\frac{\partial H_x}{\partial t} &= -\frac{1}{\mu} \left( \frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} \right) \\
\frac{\partial H_y}{\partial t} &= -\frac{1}{\mu} \left( \frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x} \right)
\end{align}  

\hspace{1cm} (4.26) \hspace{1cm} (4.27)
Figure 4.1: Position of the electric and magnetic field components in a unit cell of the Yee space lattice. [6]

\[
\frac{\partial H_z}{\partial t} = -\frac{1}{\mu} \left( \frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} \right)
\]  
(4.28)

\[
\frac{\partial E_x}{\partial t} = \frac{1}{\varepsilon} \left( \frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} - \sigma E_x \right)
\]  
(4.29)

\[
\frac{\partial E_y}{\partial t} = \frac{1}{\varepsilon} \left( \frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} - \sigma E_y \right)
\]  
(4.30)

\[
\frac{\partial E_z}{\partial t} = \frac{1}{\varepsilon} \left( \frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} - \sigma E_z \right)
\]  
(4.31)

The central difference equation is applied to Maxwell’s coupled equations, and introduces material variation with respect to spatial position in the model. The materials’ electromagnetic response in the model are represented by their conductivity and permeability. The Finite Difference equations for a space region with a finite number of distinct mediums are as follow:
CHAPTER 4. ELECTROMAGNETIC FDTD SIMULATIONS

\[ H_x^{n+\frac{1}{2}}_{i,j,k} = D_a(m) H_x^{n-\frac{1}{2}}_{i,j,k} + D_b(m) \left( E_y^n_{i,j,k+\frac{1}{2}} - E_y^n_{i,j,k-\frac{1}{2}} - E_z^n_{i,j,k+\frac{1}{2}} + E_z^n_{i,j,k-\frac{1}{2}} - M_{source_x}^n_{i,j,k} \Delta \right) \] (4.32)

\[ H_y^{n+\frac{1}{2}}_{i,j,k} = D_a(m) H_y^{n-\frac{1}{2}}_{i,j,k} + D_b(m) \left( E_x^n_{i,j,k} - E_x^n_{i,j,k-1} - E_x^n_{i,j,k+1} + E_x^n_{i,j,k+1} - M_{source_y}^n_{i,j,k} \Delta \right) \] (4.33)

\[ H_z^{n+\frac{1}{2}}_{i,j,k} = D_a(m) H_z^{n-\frac{1}{2}}_{i,j,k} + D_b(m) \left( E_x^n_{i,j,k} - E_x^n_{i,j-1,k} - E_y^n_{i,j,k} - E_y^n_{i,j,k+1} + E_y^n_{i,j,k+1} - M_{source_z}^n_{i,j,k} \Delta \right) \] (4.34)

\[ E_x^{n+1}_{i,j,k} = C_a(m) E_x^n_{i,j,k} + C_b(m) \left( H_z^{n+\frac{1}{2}}_{i,j,k} - H_z^{n+\frac{1}{2}}_{i,j-1,k} - H_y^{n+\frac{1}{2}}_{i,j,k} + H_y^{n+\frac{1}{2}}_{i,j,k-1} - J_{source_x}^{n+\frac{1}{2}}_{i,j,k} \Delta \right) \] (4.35)

\[ E_y^{n+1}_{i,j,k} = C_a(m) E_y^n_{i,j,k} + C_b(m) \left( H_x^{n+\frac{1}{2}}_{i,j,k} - H_x^{n+\frac{1}{2}}_{i,j-1,k} - H_z^{n+\frac{1}{2}}_{i,j,k} + H_z^{n+\frac{1}{2}}_{i,j,k-1} - J_{source_y}^{n+\frac{1}{2}}_{i,j,k} \Delta \right) \] (4.36)

\[ E_z^{n+1}_{i,j,k} = C_a(m) E_z^n_{i,j,k} + C_b(m) \left( H_y^{n+\frac{1}{2}}_{i,j,k} - H_y^{n+\frac{1}{2}}_{i,j-1,k} - H_x^{n+\frac{1}{2}}_{i,j,k} + H_x^{n+\frac{1}{2}}_{i,j,k-1} - J_{source_z}^{n+\frac{1}{2}}_{i,j,k} \Delta \right) \] (4.37)

The above equations 4.32 to 4.37 form the general basis for 3D FDTD simulation.
In a typical calculation within a grid cell, the H field component at \( n+1 \) is calculated from previous step at \( n \) and \( n-1 \), and then the E field component at \( n+1 \) are calculated. This process is repeated until the last time step is reached. The updating coefficients, \((C_a, C_{b1}, C_{b2}, D_a, D_{b1}, D_{b2})\), in equation 4.38 to 4.43, are introduced to take into account the magnetic charge and current present at each grid cell in this model.

Updating coefficients for the E-field component:

\[
C_a (m) = \left( \frac{1 - \frac{\sigma}{2\epsilon}}{1 + \frac{\sigma}{2\epsilon}} \right) \tag{4.38}
\]

\[
C_{b1} (m) = \left( \frac{\Delta t}{\epsilon \Delta x} \right) \frac{1}{1 + \frac{\sigma \Delta t}{2\epsilon}} \tag{4.39}
\]

\[
C_{b2} (m) = \left( \frac{\Delta t}{\epsilon \Delta y} \right) \frac{1}{1 + \frac{\sigma \Delta t}{2\epsilon}} \tag{4.40}
\]

Updating coefficients for the H-field component:

\[
D_a (m) = \left( \frac{1 - \frac{\sigma}{2\mu}}{1 + \frac{\sigma}{2\mu}} \right) \tag{4.41}
\]

\[
D_{b1} (m) = \left( \frac{\Delta t}{\mu \Delta x} \right) \frac{1}{1 + \frac{\sigma \Delta t}{2\mu}} \tag{4.42}
\]

\[
D_{b2} (m) = \left( \frac{\Delta t}{\mu \Delta y} \right) \frac{1}{1 + \frac{\sigma \Delta t}{2\mu}} \tag{4.43}
\]

For a cubic lattice, \( \Delta x = \Delta y = \Delta z = \Delta \), therefore \( \Delta 1 = \Delta 2 = \Delta \), \( C_{b1} = C_{b2} \), and \( D_{b1} = D_{b2} \). We define an integer array \( m = \text{Mediums} \ (i,j,k) \) that stores an integer pointer for each set of field vector components equation. The integer array will have a finite number of elements \( M \), which correspond to the number of distinct materials in the lattice.
4.6 Numerical dispersion

Maxwell’s curl equations, as defined by equations 4.24 and 4.25, cause dispersion of the simulated waves modes in the computational domain. The phase velocity is a function of the wavelength, direction of propagation and the grid discretization. Due to numerical dispersion, the phase velocity can deviate from the speed of light, and can lead to nonphysical solutions such as broadening, ringing of pulsed waveforms, and partial cancellation of scattered waves.

The choice of grid discretization can affect the phase velocity of the numerical waves, and also the numerical error. To ensure numerical stability, the Courant stability factor defines a upper bound such that,

\[ S = \frac{cdt}{\Delta} \leq 1 \quad (4.44) \]

The grid sampling density is defined as:

\[ N_\lambda = \frac{\lambda_o}{\Delta} \quad (4.45) \]

For practical purpose, to limit the effect of numerical dispersion, the grid size should be [6],

\[ \Delta x; \Delta y; \Delta z \leq \frac{\lambda_o}{10} \quad (4.46) \]

where \( \lambda_o \) is the highest wavelength expected in the model.
4.7 Numerical stability

Taflove [6] shows that the Courant stability condition requires that the time discretization to be bounded relative to the spatial discretization such that:

\[
\Delta t < \frac{1}{c\sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}}}; \quad c = \frac{1}{\sqrt{\mu \cdot \varepsilon}} \quad (4.47)
\]

This stability condition must be satisfied for convergence of the solution. However, limitations exist for this Courant condition. This condition assumes an infinite domain or that the absorbing boundary condition is ideal, it does not take into account discontinuities in the dielectric, and it applies only to linear dielectric. Regardless, it is useful as a guideline for the limit of the time increments, and for various dielectrics.

4.8 Boundary condition

4.8.1 Perfect electric conductor

The boundary condition for a Perfect Electric Conductor (PEC) requires the tangential component of the E field to be zero at the boundary. Thus, to model a PEC, the tangential E field is set to 0 for all \( t \geq 0 \) where the PEC is located. A representation of a PEC on the top surface of the Yee cube is shown in Figure 4.2, if we wish to place the PEC on top of the surface of the cube, we require:

\[
E^n_x|_{i,j,k+1} = E^n_x|_{i,j+1,k+1} = E^n_y|_{i,j,k+1} = E^n_y|_{i+1,j,k+1} = 0 \quad (4.48)
\]
4.9 FDTD algorithm with simulated structure

The SNS nanostructured thin film simulator was used to simulate complex morphologies under various deposition conditions, and the simulated films will be used in conjunction with FDTD modeling of optical response. This section describes the design of a 3D FDTD algorithm to simulate the electromagnetic response of nanostructured thin films. The advantage with FDTD is that one can obtain the response over all wavelengths by taking the Fourier transform of the field response of a delta function. Perfectly matched layer (PML) such as the Uniaxial Perfectly Matched Layer (UPML) [72] is often introduced at the simulation boundaries to simulate inhomogeneous media with dispersions and losses, in order to eliminate reflection and to simulate an infinite simulation space. A 1D-FDTD algorithm is shown below:

*FDTD Implementation*
1. Define problem parameters (particle position, permittivity, permeability, conductivity, grid size, source pulse)

2. Define sample material by index

3. Define sample material in simulated films

4. Initialize grid cells and UPML

5. Initialize all E and H field components to 0

6. Calculate updating constant \((C(m), D(m))\)

7. Initialize media pointer arrays

8. Initialize incident source (Gaussian or plane wave)

9. Time stepping loop \((1:nmax)\) \(t = t + n \Delta t\)
   
   (a) Solve for \(E_y\) at every point
      
      i. Update \(E\) field at interior grid
      
      ii. Update \(E\) field at right boundary
   
   (b) Solve for \(H_z\) at every point
   
   (c) Output time domain value to file
   
   (d) Update FFT

10. Output frequency domain value to file
4.10 Parallelization Approach

A parallelized FDTD algorithm requires an understanding of domain decomposition. Domain decomposition is the method of dividing the grid into smaller sub domains efficiently, and assigning them to Message Passing Interface (MPI) processes. Each processor is assigned a MPI process by dividing N MPI processes up into n, m, p processes along the x, y, z axis, respectively. The N processes are then mapped onto the grid, and processes that requires exchange of ghost cells are located physically close to each other. Ghost cells are copies of sub-domain cells that exist in another sub-domain, they allow message passing between processes and field updates to be completed. Each process is then assigned a region’s location and size in which to compute in the grid. Finally, the results are combined together at the end of the simulation. For process assignment, MPI_Dims_create can be used to divide processes in a grid, and it returns the number of processors along the three axes. MPI_Cart_coords function determines the coordinates for the process. Since the electric and magnetic field calculations require field values from surrounding points, ghost cells are required to enable field updates between sub domains. HPCVL is a shared memory multiprocessor supercomputer that is capable of running hybrid parallel programs utilizing both the MPI and OpenMP standard. A parallelized FDTD algorithm using OpenMP/MPI is shown below:

```
InitializeParam();
MPI Domain decomposition;
OPENMP InitializeFields();
for (t=0; t<tmax; t++) {
    for (i,j,k to imax, jmax, kmax) {
```
MPI exchange boundary magnetic fields with neighbours;
OpenMP update Electric field using magnetic field;
Initialize $E(1) = \text{source}(t)$
MPI exchange boundary electric fields with neighbours;
OpenMP update Magnetic field using electric field;
OpenMP update fields at boundaries;
OpenMP apply PML and UPML;
}
}
Output field values to file;
Chapter 5

Results and Discussion

5.1 Overview

This chapter presents the SNS simulation results for the investigation of nanostructured thin films under various deposition conditions using the Monte Carlo, Kinetic Monte Carlo, and Finite Difference Time Domain methods in order to gain insights into growth behavior as well as to develop the capability of the simulator to engineer different nanostructures with precise properties. What is the effect on the morphology when the incidence angle, $\alpha$, varies sinusoidally with time under constant rotation? How does the atomic interaction affect the growth mode and the resulting structure? The answers to these questions are addressed with simulations, and are compared with experimental results.
5.2 Physical structure simulations

5.2.1 Simulation parameters

Figure 5.1: KMC program structure
The program structure of the SNS-KMC simulator is shown in Figure 5.1. The SNS-MC program structure is the same as the SNS-KMC program structure except that the choice of a surface event (diffusion or re-emission) is omitted. Each particle in the simulation is defined by two attributes: spatial coordinates \((x, y, z)\) and the velocities of directional cosines: \((V_x, V_y, V_z)\). The particle is generated at a random position above the substrate and projected in a ballistic trajectory until it reaches another particle. For MC simulation, the particle will then undergo a random walk algorithm. To determine whether a particle displaces to the new position, a random number, \(q\), is generated and compared to the Boltzmann factor. The particle will relax to an allowed position if the condition, \(q < p[m]/total\), where "total" is the cumulative hopping probability and \(p[m]\) is the hopping probability at a random allowed position, \(m\). This process is iterated \(S\) times, where \(S\) is the number of random walk steps.

The hopping probability is given by:

\[
p_{i \rightarrow f} = e^{-E_{surf} \cdot T \cdot \gamma (N_f - N_i)} \tag{5.1}
\]

where \(p_{i \rightarrow f}\) is the transition probability from the initial and final position, \(T\) is the temperature of the substrate, \(E_{surf}\) is the plane diffusion energy, \(N_i\) and \(N_f\) are the coordination numbers of the particle in the initial and final position, and \(\gamma\) is a constant that corresponds to the ratio of surface energy and the temperature, in this case we assumed \(\gamma = 4/9\) to be consistent with Kaminska’s simulations [66].

The overall simulation is defined by the following parameters:

- Simulation dimension(mesh.L, mesh.M, mesh.N)
- Number of defined layer(s)
CHAPTER 5. RESULTS AND DISCUSSION

• Number of periodic layer(s)

• Height of each layer

• Alpha and phi dispersion

• Substrate tilt control parameter (minimum alpha, maximum alpha, alpha phase, wave form type, period of alpha(height per period))

• Substrate rotation control parameter (minimum phi, maximum phi, phi phase, wave form type, period of phi (height per period))

• Random number seed

• Maximum film height

• Upper limit of the number of generated particles

• Substrate profile (substrate code, substrate height, number of protrusion, radian)

• Number of random walk steps

• Substrate temperature

• Face energy

• Corner energy

SNS-KMC requires the following additional deposition input parameters:

• Deposition speed

• Energy barrier for terrace diffusion
- Energy barrier for additional CN
- Desorption energy barrier

<table>
<thead>
<tr>
<th>Code</th>
<th>Wave Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sine Wave</td>
</tr>
<tr>
<td>2</td>
<td>Triangular Wave</td>
</tr>
<tr>
<td>3</td>
<td>Rectangular Wave</td>
</tr>
<tr>
<td>4</td>
<td>Line wave (phi_period = height of full rotation)</td>
</tr>
<tr>
<td>5</td>
<td>Step wave (define number of sides in the phi.min parameter, phi_period = zig height)</td>
</tr>
<tr>
<td>6</td>
<td>2-Step wave (angle 1, angle 2, period 1)</td>
</tr>
<tr>
<td>7</td>
<td>Sine pulsed wave (min angle, max angle, npeak, period)</td>
</tr>
<tr>
<td>8</td>
<td>Filewave (720 y values defining the wave form for alpha or phi for one period)</td>
</tr>
</tbody>
</table>

### 5.2.2 Morphologies of thin films

The equilibrium film morphology and properties of nanostructure under various deposition conditions are investigated. The resulting nanostructure that forms as the incidence tilt $\alpha$, and the azimuthal angle $\phi$ are varied in time, both in a variety of manner and rate, as well as the influence of temperature, will be presented in the subsequent sections. The incidence tilt rate, $\alpha'$, and azimuthal rate, $\phi'$, are described in the input file as: min angle, max angle, phase, wave type, and period. There are seven predefined wave types as listed in Table 5.1. Alternatively, the angle variation in time can be user generated and read into the program by creating a file with a list of 720 angle values. The sine pulsed wave is a gaussian enveloped sinusoidal function that is often used to enhance pass band transmittance for rugate filters. The simulator is capable of depositing on a variety of predefined substrate structures; Figure
5.2 shows the thirteen predefined substrates in SNS. The plan and cross sectional views are constructed by determining the maximum height of the surface of the 3D simulation space, the heights are then assigned a gray scale, varying from white at the top of the film to black at the bottom.

The substrate type is defined using the code in Table 5.2. Substrate features such as size and shape can also be defined. The substrate profiles in Figure 5.2 have a simulation size of 256 x 256 x 200, the substrate parameter: substrate height, number of protrusions, and radian are 10, 10, 1, for substrate type 0, 3-8, 10-13. The radian is changed to 0.1 for substrate type 1,2, 9.

<table>
<thead>
<tr>
<th>Substrate type</th>
<th>Code</th>
<th>Fractal dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plane substrate</td>
<td>0</td>
<td>2.000 ± 0.000</td>
</tr>
<tr>
<td>$\sin^2$ substrate along x</td>
<td>1</td>
<td>2.264 ± 0.392</td>
</tr>
<tr>
<td>$\sin^2$ substrate along y</td>
<td>2</td>
<td>2.264 ± 0.392</td>
</tr>
<tr>
<td>Triangular trench substrate along x</td>
<td>3</td>
<td>2.333 ± 0.456</td>
</tr>
<tr>
<td>Triangular trench substrate along y</td>
<td>4</td>
<td>2.333 ± 0.456</td>
</tr>
<tr>
<td>Trench substrate along x</td>
<td>5</td>
<td>2.333 ± 0.456</td>
</tr>
<tr>
<td>Trench substrate along y</td>
<td>6</td>
<td>2.333 ± 0.456</td>
</tr>
<tr>
<td>Cylinder substrate</td>
<td>7</td>
<td>2.317 ± 0.433</td>
</tr>
<tr>
<td>empty cylinder substrate</td>
<td>8</td>
<td>2.287 ± 0.397</td>
</tr>
<tr>
<td>Square cylinder substrate</td>
<td>9</td>
<td>2.333 ± 0.456</td>
</tr>
<tr>
<td>Empty square cylinder substrate</td>
<td>10</td>
<td>2.282 ± 0.400</td>
</tr>
<tr>
<td>Cone substrate</td>
<td>11</td>
<td>2.317 ± 0.433</td>
</tr>
<tr>
<td>Empty cone substrate</td>
<td>12</td>
<td>2.333 ± 0.456</td>
</tr>
<tr>
<td>Wave substrate</td>
<td>13</td>
<td>2.333 ± 0.456</td>
</tr>
</tbody>
</table>
Figure 5.2: Substrate profiles 1-13.
5.2.3 Results of deposition on pre-patterned substrates

Figure 5.3: Simulated deposition on 13 prepatterned substrates.

Deposition onto pre-patterned substrates controls nucleation growth by forcing islands to assume the shape of the patterned substrate, which is crucial to develop photonic devices that require uniform, controlled structure. Simulations on patterned substrates are performed to investigate the morphology and changes in growth behavior. Figure 5.3 shows films deposited at $\alpha = 85^\circ$ with no rotation. Figure 5.4 shows an illustration on the density profile calculation, where $Z$ is the total film thickness, $z$ refers to a specific height above the substrate, the areal density is calculated as 2D packing fraction at a specific height, $z$. The graph on the right shows an example of
The density profile of the films in Figure 5.5 shows little noticeable change in the overall density profile except in the nucleation stage where the effect of shadowing is present on prepatterned substrate surfaces. However, film growth on some prepatterned substrates is clearly more uniform and structured as compared to films deposited on a plane substrate.
CHAPTER 5. RESULTS AND DISCUSSION

Figure 5.5: Effect of deposition on prepatterned substrates. Deposited at $\alpha = 85^\circ$ on non-rotating substrate. Substrate pre-patterned structures are described in Table 5.2 and shown in Figure 5.3. Note that height corresponds to the distance from the substrate surface.

Substrate growth on non planar substrates that show more prominent structural control was investigated further. Simulations on a cylinder substrate and an empty cone substrate (with radian=0.1, 1), are shown in Figure 5.6. It is evident that film growth on prepatterned substrates exhibits a more uniform and regular array of columns at nucleation as compared to deposition on plane substrates. Prepatterned substrates provide a regular seeding area for nucleation and eliminate some randomness in island formation. For substrate rotation at 1000 particles/rev, the film
structure reverts back to the type of growth that is observed on films deposited on planar substrates as the films grow.

Figure 5.6: Controllable nucleation growth on pre-patterned substrates (a) & (b) Cylinder substrate (type 7, radian=1, 0.1), (c) & (d) Empty cone substrate (type 11, radian=1, 0.1), (e) Plane substrate, with incidence angle $\alpha = 85^\circ$ and rotational rate $\phi' = 1000$ particles/rev. First two rows show the top and cross sectional view of the film, third row shows the cross sectional view of the substrates.

Comparison of the density profile shows a more uniform nucleation growth at the early stage for deposition on prepatterned substrates. However, as the film grows further, all five films exhibit similar density, as shown in Figure 5.7.
5.2.4 Effect on convex substrate

The effect of depositing onto a convex substrate is an interesting study because the convex surface allows more accessible space for film growth, as well as providing a preferential diffusion direction for adatoms [73]. Figure 5.8 shows the film evolution for deposition onto convex substrate. Larger columns grow on the apex of the convex areas of the substrate substrate, dominating column growth at the trenches. For deposition with no rotation at $\alpha = 85^\circ$, as shown in Figure 5.9, column growth on
the convex surface completely shadows the growth at the trenches, suggesting that convex pre-patterning can be used to control void size and location on the film.

Figure 5.8: Growth evolution on concave substrate with rotation: $\alpha = 65^\circ - 70^\circ$ with phi_speed = 10000 particles / rev. (a) thickness = 34 (b) thickness = 72 (c) thickness = 200.

Figure 5.9: $\alpha = 85^\circ$ with no rotation on convex substrate.
5.3 Effect on density and roughness for non-rotating films at various angles of incidence

Figure 5.10 shows ten films deposited onto non-rotating planar substrates at $\alpha = 45^\circ$ to $88^\circ$, with a simulation dimension of 256 x 256 x 100. The surface and cross sectional images for $\alpha < 45^\circ$ are not shown since the films are near bulk density with no internal porosity. Substrate temperature, surface energy and face energy are set to a nominal value of 1 in the simulation. Columnar structure extending at an angle from the substrate is evident for films with $\alpha > 45^\circ$, and the column angle and porosity increases as a function of incidence angle due to increasing adatom shadowing. At a glancing angle $\alpha > 80^\circ$, we observe a sharp decline in the number of columns and an increase in column extinction and column spacing as the film grows. Strong column extinction is a result of extreme adatom shadowing where smaller columns are shadowed by bigger columns until later incoming flux atoms can no longer reach them.
Figure 5.10: Simulated surface and cross sectional views of non-rotating films at various incidence angles, $\alpha$.

Figure 5.11 shows an interesting effect of deposition angle on film density and roughness for non-rotating films. The results exhibit the same trend as the Tait rule [41], and the deviation from the Tait rule can be attributed to the random variability in the simulated diffusion probability, temperature, and difference in deposition conditions. The results confirm that film density is a strong function of the angle of incidence, $\alpha$, of the vapor flux to the substrate. For films deposited with no substrate rotation, geometrical shadowing is the primary reason for the decrease in film density as a function of tilt angle $\alpha$. As the tilt angle increases, the effective surface area that is seen by the film decreases as a cosine of the tilt angle, and the overall film thickness decreases as a result. Film density is close to unity for $\alpha < 45^\circ$. From $45^\circ < \alpha < 75^\circ$, ...
the film density decreases approximately linearly as a function of increasing height. At $\alpha = 45^\circ$, an exponential rate of decrease is observed. For $\alpha > 70^\circ$, the scaling exponent becomes so large that the density remains constant for most of the volume of the film, as evident in Figure 5.12. High density is observed at the nucleation stage of growth for all of the films since no adatom shadowing effect is present. This shows that competition between the adatom diffusion and adatom shadowing effect are the primary factors that determine the final film morphology. The simulated average film density as a function of incidence angle agrees quantitatively with the Tait rule and XRR measurement, as shown in Figure 5.11a.

Figure 5.11: Effect of deposition angle alpha of non-rotating films on a) film density and b) RMS roughness.
Figure 5.12: Deposition angle effect: density profile of non-rotating substrate for \( \alpha = 0^\circ \) to \( 88^\circ \), \( d(\alpha) = 1 \).

Figure 5.12 shows the density of simulated films with deposition angles between \( 0^\circ \) and \( 88^\circ \). Upon careful observation, film density does not always decrease as a function of film thickness. The density decreases with increasing thickness for \( \alpha < 70^\circ \), and an increase in density with thickness is observed for \( \alpha > 70^\circ \).
Figure 5.13: Electron density profile for a) $\alpha = 40^\circ$, b) $\alpha = 60^\circ$, c) $\alpha = 70^\circ$ [7].

Asgharizadeh et al. [7] measured the x-ray reflectivity of silicon films deposited at various incidence angles, and the resulting Electron Density Profile (EDP) revealed that density increases with height above substrate for films deposited at $\alpha < 70^\circ$, and decreases for films deposited at $\alpha > 70^\circ$, as shown in Figure 5.13-b. While this observed experimental result is not in complete agreement with the equivalent simulated results in Figure 5.12, the results agree that a significant change in growth regime occurs near $\alpha = 70^\circ$. The calculated density from EDP shows that the density is approximately the same for thickness $< 50\text{Å}$, and the effect of shadowing only becomes prominent for thickness $> 200\text{Å}$.
Figure 5.14: Effect of deposition angle on density of KMC simulated film on Cu at 125K. Open circle: simulation results at 125K; Open square: simulation at 300K; Filled square: experimental data at 300K [8].

Surface roughness increases as the incidence angle increases for non-rotating films at various incidence angles. Figure 5.11b shows the effect of incidence angle on film roughness for non-rotating substrates. The RMS roughness, and density, for $\alpha < 40^\circ$ remains constant at 0.92, and 3, respectively. Density decreases as deposition angle increases while roughness increases. The decrease in RMS roughness for $\alpha = 88^\circ$ may be explained by finite size effects in the simulations. Wang et al. [8] compare KMC simulation of Cu films to the experimental films produced with GLAD [74]. The RMS roughness measurements from XRR and AFM on polycrystalline Cu films on Cu are in good agreement with simulation, resulting in the same order of magnitude. The relative density matches KMC simulation of Cu films for incident angle of $\alpha =$
85°. As shown in Figure 5.14, the film density remains nearly constant for α < 50°, consistent with the results in our simulated films. It is important to note that the average density vs. angle profile is only comparable with the same material. Different materials generally exhibit the same trend, however, their specific density values are different.

5.3.1 Effect of incidence angle for helix, zig zag, and rugate films

To investigate the effect of rotation rate at various incidence angles, five helix films were simulated at oblique angles incidence (α = 70°, 75°, 80°, 85° and 88°) with constant φ rotation at 30000 particles per revolution (/rev) for a simulation size of 256 x 256 x 200. Figure 5.15 shows the surface and cross sectional views of the simulated film. Figure 5.16 shows the density profile of these helix films under constant rotation. As the incidence angle is increased, both the pitch of the helix, and the number of helical columns decrease, resulting in a decrease in film density as a function of increased incidence angle. Figure 5.17 shows the effect of varying incidence angle α on simulated film density and roughness for a fixed azimuthal rotation period of 30,000 particles/rev.
CHAPTER 5. RESULTS AND DISCUSSION

Figure 5.15: Simulated surface and cross sectional views of helix films, with pitch = 100, phi period = 30000 particle / rev, at various incidence angles, $\alpha$.

Figure 5.16: Density profile of helix films, with pitch = 100, phi period = 30000 particle / rev at various incidence angles, $\alpha$. 
CHAPTER 5. RESULTS AND DISCUSSION

Figure 5.17: a) Density as a function of incidence angle b) RMS roughness as a function of incidence angle, for helix film with pitch=100, phi period = 30000 particle / rev at various incidence angles.

Next, the effect of substrate rotation speed at a constant incidence angle was investigated. Six helix films with constant $\alpha=80^\circ$ at various phi periods are shown in Figure 5.18. SiO$_2$ films with comparable rotational speed are shown for comparison [9]. Films with fast rotation at glancing angle of incidence ($\alpha > 70^\circ$) exhibit columnar growth that grows vertically from the substrate, and films with slow rotation form helical structures with the pitch related to the azimuthal rotation period. As the rotational speed increases, the helical structure transforms into a columnar structure, leading to a higher film density as the rotational velocity is increased, as shown in Figure 5.19. Silicon oxide has a melting point of 1710K, as such film growth will exhibit low adatom diffusion and low crystallinity. SiO$_2$ films and the simulated films show similar effects on morphological evolution with rotational speed. Figure 5.20 shows the effect on density and roughness of varying azimuthal rotational period, and
confirms that both density and roughness increase as the rotational period decreases.

Figure 5.18: Simulated surface and cross sectional view of helix films, with pitch=100 and $\alpha=80^\circ$, at various phi periods, $T_\phi$. Below show the cross sectional view of $SiO_2$ films grown at comparable rotational speeds. [9].
Figure 5.19: Density profile of helix films, with pitch=100 and $\alpha=80^\circ$, at various phi periods.
Figure 5.20: a) Density as a function of phi period b) RMS roughness as a function of phi period, for helix films with pitch = 100 and $\alpha = 80^\circ$, at various phi periods.

Eight zig zag films with pitch = 100 were simulated at various incidence angles, as shown in Figure 5.21. To construct a zig zag film, the substrate is azimuthally rotated 180° for every increase in film thickness of 50, keeping the incidence angle, $\alpha$, constant. In addition to a decrease in density as the incidence angle increases, the relative change in density also decreases, as shown in the inset of Figure 5.22. Figure 5.23 shows that the RMS roughness remains nearly constant for $80^\circ \leq \alpha \leq 84^\circ$, then increases sharply as incidence angle is increased up to $\alpha = 89^\circ$. In contrast, the RMS roughness increases linearly between $80^\circ \leq \alpha \leq 88^\circ$ for helix films with constant azimuthal rotation.
(a) $\alpha=80^\circ$  (b) $\alpha=82^\circ$  (c) $\alpha=84^\circ$  (d) $\alpha=85^\circ$  (e) $\alpha=86^\circ$  (f) $\alpha=87^\circ$  (g) $\alpha=88^\circ$  (h) $\alpha=89^\circ$

Figure 5.21: Simulated surface and cross sectional view of zig zag films, with zig zag height=100 at various incidence angles, $\alpha$.

Figure 5.22: Density profile of zig zag films, with zig zag height=100 at various incidence angles, inset shows density variation as a function of incidence angle for zig zag films.
The density and roughness of the zig zag films follow the same general trends as films with other morphologies, as shown in Figure 5.23. The Tait rule for predicting film density (Equation 2.5 is shown for comparison; the simulated density is consistently higher than the predicted values, but clearly follows the same trend. This can be accounted for in that Tait’s derivation assumes that the film is made from ideal columns, each with equal column widths and spacings. There are several important results that we have demonstrated thus far: 1) Density decreases as a function of incidence angle, 2) Roughness increases as incidence angle increases, 3) Fast azimuthal rotation increases overall film density, 4) For non-rotating films, density remains constant in the central region of the film for \( \alpha > 75^\circ \), but increases linearly for \( \alpha < 75^\circ \), 5) Density increases as a function of thickness as incidence angle decreases.

Figure 5.23: (a) Density as a function of incidence angle (b) RMS roughness as a function of incidence angle, for zig zag films with zig zag height=100 at various incidence angles.
5.3.2 Effect of phi rotation speed for rugate films

Optical interference filters with a continuous, sinusoidally varying index gradients are called rugate filters. The periodic index variation leads to a Bragg reflection and an optical stop band at a characteristic wavelength. Continuous refractive index variation can be achieved by controlling the GLAD film porosity to vary sinusoidally with film thickness. Kaminska et al. fabricated porous silicon rugate filters with GLAD and then compared them with Monte Carlo simulation; results were found to be in good agreement with experiment [10]. In the past, the Characteristic Matrix Method has been used to calculate the expected optical transmissivity for a given refractive index variation with relative success for a calibration of the simulated film. However, the optical response fails to capture all of the features seen in the fabricated films. Use of the FDTD technique to simulate the optical response should produce more accurate optical response characteristics for complex nanostructured films. Figure 5.24 shows an SEM image of a rugate filter fabricated with GLAD and the corresponding simulated film with VFIGS [10].

To investigate the effect of phi rotation for sinusoidally varied rugate filters, five films were simulated with $\alpha$ varying sinusoidally between $63^\circ$ - $84^\circ$, for rotational period $\phi'$ at 100, 50, 25, 15, 5 thickness/rev. As $\phi'$ decreases from 100 to 5 layers per revolution, the helix structure transforms into a cauliflower-like structure, as shown in Figure 5.25. The change in structure is important as the film’s polarization characteristics are highly structure dependent, and precise control of the structure parameters is required for advanced photonic devices.
Figure 5.24: (a) SEM image of a 12 period rugate filter with sinusoidal variation of refractive index fabricated with GLAD, (b) A 12 period rugate film simulated with VFIGS (with input parameters $\gamma=0.6 \times 4/9$ and $S=20$).

Figure 5.25: Simulated surface and cross sectional view of rugate films, alpha varies sinusoidally between $63^\circ$ and $86^\circ$, rugate height=100 at various rotation speed (thickness/rev), $\phi'$. The density profiles of the simulated rugate films shown in Figure 5.26 reveal that increased substrate rotational period causes to a shift in the density profile. The
inset in Figure 5.26 shows the phase shift with respect to the slowest rotation density profile. Films that are tilted but with no rotation produce columns that incline toward the vapor source. With substrate rotation, cylindrically symmetrical structures grow in the direction normal to the substrate, and in-plane anisotropy is eliminated with rapid rotation. Figure 5.27 shows the variation in density and roughness as a function of rotational period for the rugate films, the observed trends are similar to those seen in the helix films. The density and roughness both decrease as rotational period is increased.

Figure 5.26: Density profile of rugate films, alpha varies sinusoidally between 63° and 86°, rugate height=100. Inset shows the phase shift amplitude with respect to $\phi'=100$. 

![Figure 5.26: Density profile of rugate films, alpha varies sinusoidally between 63° and 86°, rugate height=100. Inset shows the phase shift amplitude with respect to $\phi'=100$.](image)
5.3.3 Temperature & surface energy dependency

The structure zone model (SZM) is a categorization scheme that describes the microstructure of a thin film based on the ratio of the substrate temperature to the melting point temperature of the depositing material. The three distinct structural transition zones are defined by their respective reduced temperature ratios. For zone 1 the ratio is \( \frac{T}{T_m} < 0.3 \), columnar structure dominates due to limited adatom diffusion at low temperature. GLAD exploits this limited adatom diffusion phenomena to produce columnar nanostructured thin films defined in zone 1 with tunable optical properties. Increasing the temperature increases surface diffusion, which results in an increase in column diameters and overall density. At very high temperature, significant adatom diffusion results in bulk-like films with density very close to unity.

Figure 5.27: a) Density as a function of rotation period b) RMS roughness as a function of rotation period for rugate films, alpha varies sinusoidally between 63° and 86° rugate height=100.
The number of random walk steps that a particle takes is a function of the temperature and the activation energy, such that \( P_{\text{step}} \propto e^{-\frac{E_a}{kT}} \), where \( P_{\text{step}} \) is the diffusion probability. The microstructure dependency on the number of random walk steps is investigated.

Figure 5.28: Simulated surface and cross sectional views of non-rotating films, with \( \alpha = 70^\circ \) at various number of random walk steps

Figure 5.28 shows eleven thin films deposited at \( \alpha = 70^\circ \) with no substrate rotation with various numbers of random walk steps. It is observed that the microstructure changes from dendritic to columnar structure as the temperature is increased. The density profile, as shown in Figure 5.29, clearly shows the effect of increase diffusion
on density. In general, the density increases gradually as a function of the number of random walk steps, as the film structure transitions from zone 1 to zone 3 in the SZM. The density increases as a function of temperature as expected due to the increase in diffusion rate, as shown in Figure 5.30(a). The corresponding RMS roughness in Figure 5.30(b) shows an expected decrease in roughness as the temperature increases.

Figure 5.29: Density profile of non-rotating films, with α=70° at various number of random walk steps.
Figure 5.30: a) Density as a function of number of random walk steps, b) RMS roughness as a function of number of random walk steps, for non-rotating films with $\alpha=70^\circ$.

### 5.4 KMC simulations

Figure 5.31 shows the surface and cross sectional view of six non-rotating films deposited at various flux angles using the KMC method. The in-plane diffusion energy, $E_{diff}$, differential diffusion energy, $dE$, and the desorption energy, $E_{des}$, is set to 0.2, 0.1, 1.6, respectively. We have explored thin film growth in the low diffusion limit. From Figure 5.31, it is observed that the low diffusion rate leads to a very dendritic structure. Increasing the diffusion rate should increase the diffusion length of the particles, thus increasing the density of the film.
CHAPTER 5. RESULTS AND DISCUSSION

Figure 5.31: Simulated surface and cross-sectional views of non-rotating films, with 
(a) $\alpha=75^\circ$, b) $\alpha=80^\circ$, c) $\alpha=85^\circ$, d) $\alpha=87^\circ$, e) $\alpha=88^\circ$, and f) $\alpha=89^\circ$.

Figure 5.32: SNS-MC and SNS-KMC density comparison for non-rotating substrate 
(MC: nwalk=0, T=10, KMC: $E_{\text{diff}}=0.1$ $dE=0.1$, T=10) for $\alpha=75^\circ$, 80°, 85°, 87°, 88°, 89°.
Figure 5.33: Roughness evolution profile of KMC simulated film, non-rotating with $\alpha=75^\circ, 80^\circ, 85^\circ, 87^\circ, 88^\circ, 89^\circ$.

As the deposition angle is increased from $75^\circ$ to $89^\circ$, a progressive decrease in density is observed, as shown in Figure 5.32b. KMC simulations show a similar trend in density as compare to MC simulations in the low diffusion limit, as shown in the comparison figures for MC and KMC in Fig 5.32. KMC simulations can reveal the evolution dynamics of the film, Figure 5.33 shows the roughness evolution, the RMS roughness is calculated after a fixed number of particles deposited, and is set to 1000 deposited particle for this simulation. The density increases approximately linearly as a function of particles deposited. The roughness increases at a higher rate for low
deposition angles, as shown in Figure 5.33.

The diffusion energy directly influences the film morphology and is dependent on the local environment of the particle. At this stage of the study, the SNS-KMC simulation utilized a simplified diffusion model in which the diffusion rate decreases linearly as a function of coordination number. Since the magnitude and the rate at which the diffusion energy changes strongly depends on the type of deposition material, MD or DFT should be used to calculate the diffusion rate under different local environments.

5.5 FDTD simulations

In this section, the possibility of utilizing FDTD to calculate the optical response of the simulated films is explored. Results are presented modeling 1-D wave propagation using the finite difference formulation of the scalar wave equation, and a 1D-FDTD plane wave simulation with Perfect Electric Conductor (PEC) and Perfect Magnetic Conductor (PMC). A Matlab program is designed to implement the solution of the finite difference 1D scalar wave equation, as shown in equation 4.9.

First, to examine the effect of numerical dispersion on the wavefunction, rectangular and Gaussian waves are simulated as an incident source propagating along the x-direction, as shown in Figure 5.34 and 5.35, respectively. As the Courant stability factor is varied from $S=1$ to $0.99$, the numerical dispersion effect is seen at the edge of the rectangular pulse showing considerable ringing caused by the discontinuity at the pulse edge, while there are no appreciable numerical dispersion effects observed for the Gaussian wave.
Figure 5.34: (a) Rectangular pulse propagation using scalar wave equation with S=1, and pulse width = 40, (b) Rectangular pulse propagation using scalar wave equation with S=0.99, and pulse width = 40.

Figure 5.35: (a) Gaussian pulse propagation using scalar wave equation with S=1, and pulse width = 40 cells, (b) Gaussian pulse propagation using scalar wave equation with S=0.99, and pulse width = 40.
5.5.1 1D x-directed plane wave propagation in Yee cube

A computer model of 1D x-direction plane wave propagation in a uniform Yee grid with Ez and Hy field components was constructed. This propagation model assumed free space everywhere, with the grid terminating in the Ez component in the left and right boundary to simulate a Perfect Electric Conductor (PEC). The PEC acts as a mirror that reflects the Ez wave as it reaches the right boundary. Figure 5.36 shows the visualization of the Ez and Hy field of a 0.1 ns gaussian pulse at a number of time intervals before and after the wave reaches the right boundary. The amplitude of $E_x$ is a factor of $\eta_o = \sqrt{\frac{\mu_o}{\epsilon_o}}$ less than $H_y$. The tangential component in the electric field $E_x(k)$ is zero on the PEC; the resulting Gaussian pulse, after it is reflected from the PEC, is $180^\circ$ out of phase, as expected.
CHAPTER 5. RESULTS AND DISCUSSION

(a) n=49  
(b) n=201
(c) n=240 
(d) n=395

Figure 5.36: Visualization of the E and H field as it hits the simulated perfect electric conductor (Ez = 0) on the right boundary. Note that Ez is reflected after it hits the right boundary and continues to propagate in the x direction, where E is in the z direction while H remains in y direction.

To model a Perfect Magnetic Conductor (PMC), wave propagation with the grid terminating in the Hy component in the right boundary was simulated, as shown in Figure 5.37. The pulse before and after collision with the PMC shows that after collision, $H_y$ is inverted while $E_x$ remains in the same phase.
Figure 5.37: Visualization of the Ez and Hy field in the presence of a simulated perfect magnetic conductor at the right boundary. The perfect magnetic conductor reflects the incident Hy wave in a 1D plane wave propagation simulation. The H field is now in +y direction and the E field remains in the +z direction.

Any material can be simulated as long as the permittivity, permeability, electric conductivity and magnetic conductivity are specified in each cell in the computational grid. The Perfectly Matched Layer (PML) absorbing boundary condition is often used to simulate the infinite computational domain.
Chapter 6

Conclusions and Recommendations

6.1 Conclusion

A growth simulator has been developed utilizing the Monte Carlo and kinetic Monte Carlo methods to simulate nanostructured thin films fabricated with the GLAD technique. A variety of deposition, diffusion, and desorption models are studied. The SNS simulator produces 3D virtual films that provide qualitative and quantitative measurements of films fabricated under a wide variety of deposition conditions. The effect of the incident angle, substrate rotation rate, tilt rate, and temperature, on the resulting morphology of the thin films are investigated.

Film density and roughness are shown to be highly controllable through precise control of the tilt and rotation rate during film growth. Helix, zig zag, rugate, and inclined columnar films had been simulated and analyzed under various deposition conditions. Rotational rate effects on rugate films were examined; it is determined that varying rotational velocity caused to a phase shift in the simulated density profiles, which is the result of an increase in cylindrical symmetry that arises as rotational
Simulation of zig zag films shows that an increase in deposition angle leads to a decrease in density variation. The simulation results of non-rotating films deposited at various deposition angles are consistent with x-ray reflectivity measurement [7] and show that the density decreases with increasing incidence angle, $\alpha$. The effect of atomic diffusion shifting over to the shadowing effect [7] for $\alpha > 70^\circ$ is not observed.

A comprehensive review including the theory of finite difference and the derivation of the FDTD method of numerically solving Maxwell’s equations was presented. A Matlab program has been implemented for finite difference modeling of 1D wave propagation using the scalar wave equation for demonstrating the effect of numerical dispersion. A 1D-FDTD model was constructed that demonstrated the effect of a perfect electric and magnetic conductor on the electric and magnetic fields of a propagating electromagnetic wave. The powerful SNS thin film deposition simulator, in conjunction with FDTD modeling of optical response, will be used as an effective tool for designing and fabricating engineered nanostructured thin films.

### 6.2 Recommended future work

The ultimate goal of this work is to predict the physical structure and optical response of nanostructured thin films using Monte Carlo, Kinetic Monte Carlo and Finite-Difference Time-Domain methods, as realistically as possible when compared with experiment. Density profiles of the simulated film will be compared against gravimetric and XRR measurement of density in experimental films. Consequently, KMC simulation of the physical structure will be performed, and the simulated transmission and reflection response using the FDTD method will be compared against
ellipsometry measurements of fabricated films. The physical film microstructures are then compared through images taken with the scanning electron microscope.

The film morphology dependence on the type and magnitude of the diffusion rate is not yet well understood and should be investigated using DFT to construct a rate table that specifies the diffusion energy and rate under different diffusion events and with different materials. A more accurate diffusion model that can give the best approximation to the diffusion rate, taking into account all possible diffusion events for different materials, should be studied and implemented into the SNS-KMC simulator to more realistically describe the potential energy surface of diffusing adatoms on the film surface.

Optimization and parallelization of the simulator should be performed to make the simulator more efficient in performing larger scale simulations and to eliminate finite size effects, which are most evident under low diffusion conditions. Processes with a slow reaction rate, such as desorption, may never occur if the process time exceeds the deposition time. Since the desorption rate is relatively slow and only becomes significant at high temperatures, one can consider excluding processes with very slow rates from the simulator.

In the current simulator, particles are mapped onto a cubic simulation grid; a lattice grid can be overlaid onto the simulation grid to allow preferential lattice sites to simulate materials with different crystal structures. Addition of background gas during deposition may affect film morphology through preferential adsorption and growth direction at specific lattice sites and should be taken into account in the simulator.

For FDTD electromagnetic simulation, any 1D structure with varying material
with respect to spatial position can be specified. It is envisioned that the simulated physical structure from SNS is used as the input for the FDTD simulator. The FDTD simulator will then generate the electromagnetic response of the nonlinear dielectric materials, by calculating the Fourier transform to the time evolution of the transmitted and reflected electric field, which yields the transmission and reflection spectrum of the simulated structure.
Bibliography


Appendix A

A.1 SNS source code

#include "VFiGS1.h"
#include "math.h"
#include "make_substrate.c"
#include "nrelax.c"
#include "CHGEX.c"
#include "FileExist.c"
#include "GETEX.c"
#include "bit_on.c"
#include "random.c"
#include "waves1.c"

#define roughInt 10000
int mouse_stop=1;

/*
 Purpose: Determine if position is occupied in lattice (1D binary) array.
 Input: unsigned long X, Y, Z, unsigned long *lattice, simulation_size mesh
 Output: Returns 1 if position (x,y,z) is occupied and 0 if it’s unoccupied
*/
int display_particle (unsigned long X, unsigned long Y, unsigned long Z, unsigned long *lattice,
                     simulation_size mesh)
{
    unsigned long ell, part_address;
    int particle_bit;

    ell=mesh.L/(unsigned long)32;
APPENDIX A. APPENDIX

part_address=11*(Z+mesh.M)+Y/(unsigned long)32;
particle_bit=(int)X%32;

return ((lattice[part_address]>>particle_bit)&1);
}

/*/ Purpose: First condition to determine if neighboring position is occupied
Input: unsigned long X, Y, Z, unsigned long *lattice, simulation size mesh
Output: return 1 if occupied, 0 if unoccupied
*/
int sticking_judge(unsigned long X, unsigned long Y, unsigned long Z, unsigned long *lattice,
simulation size mesh)
{
int x[3], y[3], z[3];
int i, j, k;
int bond=0;

x[0]=(int)X; y[0]=(int)Y; z[0]=(int)Z;

//Wrap around condition
x[1]=(int)X+1;
if (x[1]>(int)mesh.L)
x[1]=mesh.L;
y[1]=(int)Y+1;
if (y[1]>(int)mesh.M)
y[1]=mesh.M;
z[1]=(int)Z+1;

x[2]=(int)X-1;
if (x[2]<0)
x[2]+=mesh.L;
y[2]=(int)Y-1;
if (y[2]<0)
z[2]=(int)Z-1;

for (i=0; i<3; i++)
{
    for (j=0; j<3; j++)
    {
        for (k=0; k<3; k++)
        {
            bond=display_particle((unsigned long)x[i], (unsigned long)y[j], (unsigned long)z[k], lattice, mesh);
            if (bond) break;
        }
        if (bond) break;
    }
    if (bond) break;
}
return bond;
}

// Set lattice array to occupied
int particle_set(unsigned long X,
unsigned long Y,
unsigned long Z,
unsigned long *lattice,
simulation_size mesh)
{
    unsigned long ell, part_address;
    int particle_bit;
    ell=mesh.L/(unsigned long)32;
    part_address=ell*(Z*mesh.M+Y)+X/(unsigned long)32;
    particle_bit=(int)X%32;
    return lattice[part_address]=bit_on(lattice[part_address], particle_bit);
}

// Terminate calculation
void stop_process(int sig)
{
    int c;
    char s[512];

    printf("Do you want to exit? If so, please input Y. \n");
    c=getchar();
    if (toupper(c)=='Y')
    {
        (void) getchar();
        mouse_stop=0;
    }
    else
    mouse_stop=1;
    if (c!='\n')
        (void)gets(s);
    signal(SIGINT, stop_process);
}

/*
 * Function name: deposition
 * Purpose: Calculate the final lattice using "sticking_judge3" and "relax_particle2". Each
 * particle position is determined based on two conditions: 1) the particle is unoccupied 2)
 * The hopping probability of surrounding position. Output final lattice.
 * Split diffusion parameter into tokens:
 * diffusion_parameter -> rand_walkstep, unit_surface_energy, corner_potential
 * The probability [i] = exp(-temperature*unit_surface_energy*(6.0-(double)i/2.25)), where i
 * is the coordination number.
 * Calculate ang_flux[360]=1.0/cos(Pi*(double)i/4.0/180.0);
 * Define alphatable [720] depending on alpha control (1. sin 2. tri 3. rect wave 4. filewave)
Get max height from "make_substrate(lattice, mesh, subst_type)" based on substrate type
Use "relax_particle2" to determine final relax position, loop until upper limit of particle
generated or max substrate height is reached

Input: simulation_size mesh, depo_angle alpha, azimuth phi, char *subst_type, char *
diffusion_parameter, char *filename

Output: unsigned long gridsz = mesh.L/32 * mesh.M * mesh.N
unsigned long max_height, num_part_generated
int rand_walkstep; //number of random walk step
double temperature, unit_surface_energy, corner_potential
unsigned long lattice array [gridsz]
double probability [27] //probability of stable site in neighboring position
double angular_flux[360] //store angular flux for each of the 360 degree
double alphatable [720] //store alpha step
int phase; //alpha phase

int deposition (simulation_size mesh, depo_angle *alpha, depo_angle *phi, unsigned long max_height,
char *subst_type, char *diffusion_parameter, char *filename, int iteration, int repeat_layer,
int num_periodic)
{
unsigned long ell, gridsz, temp;
unsigned long num_part_generated=0;
unsigned long no_bond=0; //number of particle deposited
double x, y, z, x_temp, y_temp, z_temp;
unsigned long X, Y, Z;
double vx, vy, vz;
double vx_temp, vy_temp, vz_temp;
double disp_angle_alpha, disp_angle_phi;
double deposition_angle, in_plane_angle;
char angleout1[512]="", angleout2[512]="";
int i, j, k, m, n, rand_walkstep; //men_potential, hen_potential, corner_potential*/
double temperature, unit_surface_energy, corner_potential;
char *filename="", extension[512]="", logname[512]="", partset[512]="";
char *delimit="\t", *first_token, param[512]="", decomp[512]="";
double *probability = malloc(27*sizeof(double));
unsigned long subst_height;
int subst_code, no_of_deppari;
double rad;
unsigned long currHeight=0; //current deposition height
unsigned long currlayer_height=0; //height of current layer

double flux=0.0, angular_flux[360];
double** alpha_table; //store alpha step for each different layer
alpha_table = (double**)malloc(iteration*sizeof(double));
for (i=0; i<iteration; i++)
alpha_table[i]=(double*) malloc(720*sizeof(double));

double** phi_table; //store alpha step for each different layer
phi_table = (double**) malloc(iteration*sizeof(double*));
for (i=0; i<iteration; i++)
  phi_table[i]=(double*) malloc(720*sizeof(double));

int phase1, phase2; //alpha phase
double area= (double)(mesh.L*mesh.M); //surface area
int count =0; //count when deposition reach max_height/2
double davg=0.; //average density
double density; //density at height defined by densPlot
unsigned long densPlot=(unsigned long)(mesh.N/4); //define height for density slab vs time plot
double zavg=0.; // average height for calculating RMS roughness
double ravg=0.; //average roughness
int*** space; //particle position
int** zmax; // max height for each x,y
double** zs;
double** za;
double** zc; //rms height
int dim1=(int)mesh.L;
int dim2=(int)mesh.M;
int dim3=(int)mesh.N;

double rrms=0.; //rms roughness
double ra=0.; //average roughness
double ssk=0.; //surface skewness
double zs_sum=0., za_sum=0., zc_sum=0.;

//check if max_height is greater than simulation size height
int total_layer_height=0;
if (max_height > (mesh.N)) {
  max_height = mesh.N-3;
  for (i=0;i<iteration; i++){
    total_layer_height+=alpha[i].layer_height;
    repeat_layer = (int)(max_height/total_layer_height);
  }
}

//3d array
space = (int***) malloc(dim1*sizeof(int**));
for (i=0; i<dim1; i++){
  space[i]=(int**) malloc(dim2*sizeof(int*));
  for (j=0; j<dim2; j++)
    space[i][j]=(int*) malloc(dim3*sizeof(int));
}

//2d double
zs = (double**) malloc(dim1*sizeof(double*));
for (i=0; i<dim1; i++)
  zs[i]=(double*) malloc(dim2*sizeof(double));

za = (double**) malloc(dim1*sizeof(double));
for (i=0; i<dim1; i++)
  za[i]=(double *) malloc(dim2*sizeof(double));

zc = (double**) malloc(dim1*sizeof(double*));
for (i=0; i<dim1; i++)
  zc[i]=(double*) malloc(dim2*sizeof(double));

// 2d int
zmax = (int**) malloc(dim1*sizeof(int*));
for (i=0; i<dim1; i++)
  zmax[i]=(int*) malloc(dim2*sizeof(int));

// if (space==NULL)
{ printf(" Error . Not enough memory! \n"); return 0; }
// if (zs==NULL)
{ printf(" Error . Not enough memory! \n"); return 0; }
// if (zmax==NULL)
{ printf(" Error . Not enough memory! \n"); return 0; }

// initialize space, zmax, zs array
for (i=0;i<dim1;i++){
  for (j=0;j<dim2;j++) {zmax[i][j]=0; zs[i][j]=0. ;}
}
for (i=0;i<dim1;i++){
  for (j=0;j<dim2;j++){
    for (k=0;k<dim3;k++) {space[i][j][k]=0;}}}

// if((flog=fopen(logname,"wb")==NULL)
// exit(EXIT_FAILURE);

strcpy(decomp, subst_type);
first_token=strtok(decomp, delimite);
subst_code=atoi(first_token);
first_token=strtok(NULL, delimite);
subst_height=atol(first_token);
first_token=strtok(NULL, delimite);
no_of_deppari=atoi(first_token);
first_token=strtok(NULL, delimite);
rad=atof(first_token);

strcpy(param, diffusion_parameter);
first_token=strtok(param, delimite);

unsigned long *lattice=malloc(mesh.L/(unsigned long)32 * mesh.M * mesh.N * sizeof(unsigned long) );
if(lattice==NULL){ printf(" Error . Not enough memory! \n"); return 0; }

e1l=mesh.L/(unsigned int)32;
gridsize=e1l*mesh.M*mesh.N;

memset set the first n(3rd parameter) bytes of the variable(1st parameter) to the specified 
value (2nd parameter)

// initialize lattice array to 0 (unoccupied)
memset (lattice, 0, gridsize*sizeof(unsigned int));
APPENDIX A. APPENDIX

rand_walkstep=atoi(first_token);
first_token=strtok(NULL, delimit);
temperature=atof(first_token);
first_token=strtok(NULL, delimit);
unit_surface_energy=atof(first_token);
first_token=strtok(NULL, delimit);
corner_potential=atof(first_token);

// higher hopping probability with higher coordination number
for (i = 0; i < 27; i++)
    probability[i]=exp(-temperature*unit_surface_energy*(6.0-(double)i/2.25));

for (i = 0; i < 360; i++)
    ang_flux[i]=1.0/cos(PI*(double)i/4.0/180.0); // question: why

// get max height given subst_type
// currHeight = max_height;

printf("String to express substrate: %s\n", subst_type);
printf("Diffusion parameters: %s\n", diffusion_parameter);

for (i = 0; i < 999; i++)
{
    sprintf(extension, "%d", i);
    filename=chgex(file_name, extension);
    if (!FileExist(filename))
        break;
}
strcat(logname, file_name);
strcat(logname, ".log");
if ((f_log=fopen(logname, "wb"))==NULL)
    exit(EXIT_FAILURE);
// generate output filename
strcat(output1, "data/");
strcat(output1, file_name);
strcat(output1, ".densVtime.log");
strcat(output2, "data/");
strcat(output2, file_name);
strcat(output2, ".roughVtime.log");
strcat(output3, "data/");
strcat(output3, file_name);
strcat(output3, ".finalDens.log");
strcat(output4, "data/");
strcat(output4, file_name);
strcat(output4, ".finalRough.log");

printf("output1 is %s\n", output1);
printf("output2 is %s\n", output2);
printf("output3 is %s\n", output3);
printf("output4 is %s\n", output4);
APPENDIX A. APPENDIX

```c
if ((fdensfinal=fopen(output3,"w")) == NULL) exit(EXIT_FAILURE);
if ((froughfinal=fopen(output4,"w")) == NULL) exit(EXIT_FAILURE);

fprintf(f_log, "# VFGS\tVersion\n");
fprintf(f_log, "# %lu\tSize in x direction\n", mesh.L);
fprintf(f_log, "# %lu\tSize in y direction\n", mesh.M);
fprintf(f_log, "# %lu\tSize in z direction\n", mesh.N);
fprintf(f_log, "# Number of layers: %d\n", repeat_layer*iteration);
for (i=0; i<iteration; i++)
{
    fprintf(f_log, "# %d\tHeight %d\n", alpha[i].layer_height, i);
    fprintf(f_log, "# %f\t%f\tDispersion of alpha\tphi\n", alpha[i].sigma/PI*180.0, phi[i].sigma/PI*180.0);
    fprintf(f_log, "# %f\t%f\t%f\t%f\tControl of alpha\t\t\t\n", alpha[i].min, alpha[i].max, alpha[i].phase, alpha[i].type, alpha[i].period);
    fprintf(f_log, "# %f\t%f\t%f\t%f\tControl of phi\t\t\t\n", phi[i].min, phi[i].max, phi[i].phase, phi[i].type, phi[i].period);
}
max_height=make_substrate(lattice, mesh, subst_type);
cur_layer_height+=alpha[0].layer_height;
for (m=0;m<repeat_layer;m++)
{
    for (n=0;n<iteration;n++)
    {
        // iterating through different layer (periodic or non-periodic) and set up alpha_table
        switch(alpha[n].type)
        {
        case 1:
            sinwave(alpha_table, alpha[n].min, alpha[n].max, alpha[n].phase, n);
            break;
        case 2:
        
```
triwave (alpha_table, alpha[n].min, alpha[n].max, alpha[n].phase, n);  
break;
case 3:  
rectwave (alpha_table, alpha[n].min, alpha[n].max, alpha[n].phase, n);  
break;
case 4:  
linewave (alpha_table, n);  
break;
case 5:  
stepwave (alpha_table, alpha[n].min, n);  
break;
case 6:  
pulsedwave (alpha_table, alpha[n].min, alpha[n].max, alpha[n].phase, n);  
break;
default:  
filewave (alpha_table, alpha[n].type, n);  
break;
}
switch (phi[n].type)
{
case 1:  
sinwave (phi_table, phi[n].min, phi[n].max, phi[n].phase, n);  
break;
case 2:  
triwave (phi_table, phi[n].min, phi[n].max, phi[n].phase, n);  
break;
case 3:  
rectwave (phi_table, phi[n].min, phi[n].max, phi[n].phase, n);  
break;
case 4:  
linewave (phi_table, n);  
break;
case 5:  
stepwave (phi_table, phi[n].min, n);  
break;
case 6:  
pulsedwave (phi_table, phi[n].min, phi[n].max, phi[n].phase, n);  
break;
default:  
filewave (phi_table, phi[n].type, n);  
break;
}

while ((flux <= mesh.number_of_particle) && (max_height < currlayer_height) && (max_height < mesh.N - 3) && mouse_stop)
{
x=mrnd()*(double)mesh.L;  
y=mrnd()*(double)mesh.M;  
z=(double)max_height+2.0;
// phase1=(int)floor(720.0*flux/alpha.period);
phase1=(int)floor(720.0*max_height/alpha[n].period);
while((phase1>=720) || (phase1==720))
alpha[n].av=alpha_table[n][phase1]; //average deposition angle

// phase2=(int)floor(720.0*flux/phi.period);
phase2=(int)floor(720.0*max_height/phi[n].period);
while((phase2>=720) || (phase2==720))
phi[n].av=phi_table[n][phase2]; //average phi deposition angle

//determine deposition angle with dispersion using gaussian distribution function
disp_angle_alpha=gaussd(0.000, alpha[n].sigma/*PI*/180.0*);
disp_angle_phi=gaussd(0.000, phi[n].sigma/*PI*/180.0*);
// disp_angle_phi=mrnd(); 2.0*PI;
v_x=sin(disp_angle_alpha)*cos(disp_angle_phi);
v_y=sin(disp_angle_alpha)*sin(disp_angle_phi);
v_z=cos(disp_angle_alpha);

deposition_angle=-alpha[n].av/*PI*/180.0*; 
v_x_temp=cos(deposition_angle)+v_x=sin(deposition_angle)*v_z;
v_x_temp=sin(deposition_angle)+v_x+cos(deposition_angle)*v_z;

v_x=v_x_temp;
v_z=v_z_temp;

in_plane_angle=-phi[n].av;
v_x_temp=cos(in_plane_angle)+v_x=sin(in_plane_angle)*v_y;
v_y_temp=sin(in_plane_angle)+v_x+cos(in_plane_angle)*v_y;

v_x=v_x_temp;
v_y=v_y_temp;

x_temp=x; y_temp=y; z_temp=z;

//random walk until stick to neighbor
//while neighbor is unoccupied and height < max height2
while(!sticking_judge((unsigned long)x, (unsigned long)y, (unsigned long)z, lattice, mesh)) 
& ((unsigned long)z < max_height+3)
{
while(abs((int)x-(int)x_temp)<1 & abs((int)y-(int)y_temp)<1 & abs((int)z-(int)
z_temp)<1)
{
    x_temp+=v_x;
    if (x_temp<0)
        x_temp=(double)mesh.L;
    else if (x_temp>=(double)mesh.L)
        x_temp=(double)mesh.L;
    y_temp+=v_y;
    if (y_temp<0)
        y_temp=(double)mesh.M;
else if (y_temp > (double) mesh.M)
    {y_temp -= (double) mesh.M;}
    s_temp += v_z;
}
x=x_temp;
y=y_temp;
z=z_temp;
// printf("%d\n", (unsigned long)z);
}
if((unsigned long)z<max_height+2)
{
    X=(unsigned long)x;
    Y=(unsigned long)y;
    Z=(unsigned long)z;
    // for (i=0; i<rand_walkstep; i++)
    i = 0;
    // higher random walk step leads to higher column diameter as particles tends to be able to move to a lower energy position
    while (i < rand_walkstep && Z < mesh.N - 2)
    {
        relax_particle2(&X, &Y, &Z, probability, lattice, mesh);
        i++;
        // if(Z==mesh.N - 2)
        // break;
    }
particle_set(X, Y, Z, lattice, mesh);
    if(Z>max_height)
    {
        max_height=(unsigned long)Z;
        printf("%f\t%f\t%lu\t%f\t\n", alpha[n].av/PI*180.0, phi[n].av/PI*180.0, max_height, flux);
        fprintf(f_log, "%f\t%f\t%lu\t%f\t\n", alpha[n].av/PI*180.0, phi[n].av/PI*180.0, max_height, flux);
    }
    no_bond++;
}
flux+=ang_flux[(int)floor(fabs(alpha[n].av/PI*180.0)) +4];
num_part_generated++; 
}
currlayer_height+= alpha[n].layer_height;
} 
}
count=0;
// output final density
for(k=0; k<(int)mesh.N; k++)
{
    for (j=0; j<(int)mesh.M; j++)
    {
        for (i=0; i<(int)mesh.L; i++)
        {
...
if (display_particle(i, j, k, lattice, mesh)==1) {count++;
}

davg+=(double)(count/area);
fprintf(fdensfinal,"%d\t%lf\n",k,(double)(count)/area);
count=0;
}

//davg=max_height;
//fprintf(fdensfinal,"Average density\t%lf\n",davg);
count=0; //sum of max height

//output final roughness
for (i=0;i<dim1;i++)
{
    for (j=0;j<dim2;j++)
    {
        for (k=(int)subst_height+3; k<dim3;k++)
        {
            if (display_particle(i, j, k, lattice, mesh)==1 && zmax[i][j] < k)
                zmax[i][j] = k; // max height given a xy
        }
        sum+=zmax[i][j];
    }
}

zavg=(double)(sum)/area;
for (i=0;i<(int)mesh.L;i++)
{
    for (j=0;j<(int)mesh.M;j++)
    {
        za[i][j]=abs(zmax[i][j]-zavg);
        zs[i][j]=pow((zmax[i][j]-zavg),2);
        zc[i][j]=pow((zmax[i][j]-zavg),3);
        za_sum+=za[i][j];
        zs_sum+=zs[i][j];
        zc_sum+=zc[i][j];
    }
}

rrms=sqrt(zs_sum/area);
ra=za_sum/area;
ssk=zc_sum/(area*pow(rrms,3));
fprintf(f_log, "%lf\tAverage Roughness\n", ra);
fprintf(f_log, "%lf\tRMS Roughness\n", rrms);
fprintf(f_log, "%lf\tSurface Skewness\n", ssk);

for (i=0;i<(int)mesh.L;i++)
{
    for (j=0;j<(int)mesh.M;j++)
    {
        zs[i][j]=pow((zmax[i][j]-zavg),2);
        zs[i][j]=sqrt(zs[i][j]);
        ravg += zs[i][j];
        fprintf(froughfinal,"%d\t%d\t%lf\n",i,j,zs[i][j]);
    }
}
APPENDIX A. APPENDIX

```c
{ fprintf(froughfinal,"\n"); } 
rvavg = rvavg/area; 
fprintf(froughfinal,"RMS Roughness:\t%f\n",rvavg); 
fclose(fdensityfinal); 
// fclose(frough); 
// fclose(fdensity); 
// fclose(fsusb); 
// printf("%d",no_bond/0); 
printf("%lu particles were generated. \n", num_part_generated); 
printf("%lu particles were deposited. \n", no_bond); 
fprintf(f_log,"%lu\tNumber of generated particles\t%f\tIntegrated flux\n", num_part_generated, flux); 
fprintf(f_log,"%lu\tNumber of deposited particles\n", no_bond); 
fclose(f_log); 
// output result to "test.[num]"
if((f_result=fopen(filename,"wb"))==NULL) 
exit(EXIT_FAILURE); 
fprintf(f_result,"VF693\tVersion\t2\n"); 
fprintf(f_result,"%lu\tSize in x direction\n", mesh.L); 
fprintf(f_result,"%lu\tSize in y direction\n", mesh.M); 
fprintf(f_result,"%lu\tSize in z direction\n", mesh.N); 
fprintf(f_result,"%d\tNumber of defined layer\n", iteration); 
fprintf(f_result,"%d\tNumber of periodic layers\n", repeat_layer); 
for (i=0;i<iteration;i++) 
{ fprintf(f_result,"%d\tHeight %d\n", alpha[i].layer_height,i); 
 fprintf(f_result,"%f\t%f\tDispersion of alpha\tphi\n", alpha[i].sigma/PI*180.0,phi[i].sigma/PI*180.0); 
 fprintf(f_result,"%f\t%f\t%f\t%f\tControl of alpha\t\n", alpha[i].min, alpha[i].max, alpha[i].phase, alpha[i].type, alpha[i].period); 
 fprintf(f_result,"%f\t%f\t%f\t%f\tControl of phi\t\n", phi[i].min, phi[i].max, phi[i].phase, phi[i].type, phi[i].period); 
 } 
fprintf(f_result,"12345\n"); 
fprintf(f_result,"%lu\tMax height\n", max_height); 
fprintf(f_result,"%f\tUpper limit of the number of generated particles (normalized at alpha=0 deg.\n", mesh.number_of_particle); 
fprintf(f_result,"%lu\tNumber of generated particles\t%f\tIntegrated flux\n", num_part_generated, flux); 
fprintf(f_result,"%lu\tNumber of deposited particles\n", no_bond); 
fprintf(f_result,"%s\tsubst_type\n", subst_type); 
fprintf(f_result,"%s\tdiffusion_parameter\n", diffusion_parameter); 
fclose(f_result); 
if((f_result=fopen(filename,"rb+"))==NULL) 
exit(EXIT_FAILURE); 
 fseek(f_result, (size_t)(512*13), SEEK_SET); 
```
for (i=0; i<(int)(mesh.L/(unsigned long)32 * mesh.M * mesh.N); i++)
{
    putc((lattice[i]>>CHAR_BIT+3)&0xFF, f_result);
    putc((lattice[i]>>CHAR_BIT+2)&0xFF, f_result);
    putc((lattice[i]>>CHAR_BIT)&0xFF, f_result);
    putc(lattice[i]&0xFF, f_result);
}
fclose(f_result);

printf("Finish!
Now saving result.
\n") ;
free(lattice);
// DisposPtr((Ptr)lattice);
// TempDisposeHandle(aHandle , &aErr);
printf("OK\n\n") ;
free(space);
free(zs);
free(zmax);
return 1;
}

/*
 * Function main
 * Integer function: return 0
 * Purpose: Read input parameter file,
 * Input: filename from argument
 * Output: - simulation.size mesh: unsigned long X, Y, Z, double number_of_particle
 * - depo_angle alpha
 *   double av: Average deposition angle (degree)
 *   double sigma: Dispersion of the deposition angle (degree)
 *   double min; Minimum deposition angle
 *   double max: Maximum deposition angle
 *   double phase: Initial phase or duty angle
 *   double period: Perios of the variation of the deposition angle
 *   int type: Controle of deposition angle (1: sin, 2: triwave, 3. rectwave alpha.min, alpha.max, alpha.phase, alpha.type) 
 * - azimuth phi: double step, double speed
 * - long random seed
 * - char subst_type, diffusion parameter
 * - Send input parameter into (int deposition)
 */
int main(int argc, char **argv )
{
    FILE *f_input;
    char parameter[512]="", *delimit="\t", *first_token, ddfilename[256]="";
    char subst_type[512]="";
    char diffusion_parameter[512]="";
    simulation.size mesh;
APPENDIX A. APPENDIX

// depo_angle alpha;
// azimuth phi;
int num_periodic, num_layer, iteration=1, repeat_layer=0;
long random_seed;
unsigned long max_height;
int i, j;

// azimuth *phi;
// phi = (azimuth *)malloc(iteration*sizeof(azimuth));
signal(SIGINT, stop_process);

for(i=1; i<argc; i++)
{
    mouse_stop=1;
    // get input file
    if(((f_input=fopen(argv[i], "r")==NULL))
    {
        exit(EXIT_FAILURE);
    }
    sprintf(ddfilename, "%s", argv[i]);
    printf(" Parameter File: %s\n", ddfilename);

    fgets(parameter, 512, f_input);
    first_token=strtok(parameter, delimit);

    // check for correct version number, returns 0 if equal
    if(strcmp(first_token, "VFiGS3")
    {
        printf(" Error. Incorrect version. \t%s\n", first_token);
        fclose(f_input);
        exit(EXIT_FAILURE);
    }

    // Split string into token using tab delimiter
    fgets(parameter, 512, f_input);
    first_token=strtok(parameter, delimit);
    mesh.L=(unsigned long) atol(first_token);
    printf(" Size in x direction=%lu\n", mesh.L);

    fgets(parameter, 512, f_input);
    first_token=strtok(parameter, delimit);
    mesh.M=(unsigned long) atol(first_token);
    printf(" Size in y direction=%lu\n", mesh.M);

    fgets(parameter, 512, f_input);
    first_token=strtok(parameter, delimit);
    mesh.N=(unsigned long) atol(first_token);
    printf(" Size in z direction=%lu\n", mesh.N);
fgets(parameter, 512, f_input);
first_token=strtok(parameter, delimit);
num_layer=(unsigned long) atol(first_token);
printf("Number of layer=%lu\n", num_layer);

fgets(parameter, 512, f_input);
first_token=strtok(parameter, delimit);
num_periodic=(unsigned long) atol(first_token);
printf("Number of periodic layer=%lu\n", num_periodic);

// read in defined layer properties
if (num_periodic!=0){
  iteration= num_periodic;
} else{
  iteration= num_layer;
}

depo_angle *=alpha;
alpha= (depo_angle *) malloc(iteration*sizeof(depo_angle));
depo_angle =phi;
phi= (depo_angle *) malloc(iteration*sizeof(depo_angle));

for (j=0; j<iteration; j++){
 fgets(parameter, 512, f_input);
  first_token=strtok(parameter, delimit);
  alpha[j].layer_height=(unsigned long) atol(first_token);
  printf("Height of layer %d=%d\n", j+1, alpha[j].layer_height);
  fgets(parameter, 512, f_input); // alpha dispersion
  first_token=strtok(parameter, delimit);
  alpha[j].sigma = atof(first_token)*PI/180.0;
  printf("Dispersion of alpha=%f (degree)\n", alpha[j].sigma/PI*180.0);
  first_token=strtok(NULL, delimit);
  phi[j].sigma = atof(first_token)*PI/180.0;
  printf("Dispersion of phi=%f (degree)\n", phi[j].sigma/PI*180.0);
  fgets(parameter, 512, f_input);
  first_token=strtok(parameter, delimit);
  alpha[j].min = atof(first_token);
  printf("Min. of alpha=%f (degree)\n", alpha[j].min);
  first_token=strtok(NULL, delimit);
  alpha[j].max = atof(first_token);
  printf("Max. of alpha=%f (degree)\n", alpha[j].max);
  first_token=strtok(NULL, delimit);
  alpha[j].phase = atof(first_token);
  printf("Initial phase=%f (degree)\n", alpha[j].phase);
  first_token=strtok(NULL, delimit);
  alpha[j].type = atoi(first_token);
  printf("Control of alpha=%d \n", alpha[j].type);
  first_token=strtok(NULL, delimit);
  alpha[j].period = atof(first_token);
printf("Period of alpha changing=%f (No. of particles) normalized at alpha=0 deg\n", alpha[j].period);

fgets(parameter, 512, f_input);
first_token=strtok(parameter, delimit);
phi[j].min = atof(first_token);
printf("Min. of phi=%f (degree)\n", phi[j].min);
first_token=strtok(NULL, delimit);
phi[j].max = atof(first_token);
printf("Max. of phi=%f (degree)\n", phi[j].max);
first_token=strtok(NULL, delimit);
phi[j].phase = atof(first_token);
printf("Initial phase=%f (degree)\n", phi[j].phase);
first_token=strtok(NULL, delimit);
phi[j].type = atoi(first_token);
printf("Control of phi=%d \n", phi[j].type);
first_token=strtok(NULL, delimit);
phi[j].period = atof(first_token);
printf("Period of phi changing=%f (No. of particles) normalized at phi=0 deg\n", phi[j].period);

fgets(parameter, 512, f_input);
first_token=strtok(parameter, delimit);
random_seed=atol(first_token);
printf("Random seed=%ld\n", random_seed);

fgets(parameter, 512, f_input);//max height
first_token=strtok(parameter, delimit);
max_height=(unsigned long)atol(first_token);
printf("Max Height=%ld\n", max_height);

fgets(parameter, 512, f_input);//upper particles count limit
first_token=strtok(parameter, delimit);
mesh.number_of_particle=(double)atof(first_token);
printf("Generated particle limit (normalized at alpha=0 deg.)=%f\n", mesh.number_of_particle);

fgets(parameter, 512, f_input);//particle generated, intergrated flux
fgets(parameter, 512, f_input);//deposited particle
fgets(subst_type, 512, f_input);//substrate type
fgets(diffusion_parameter, 512, f_input);//diffusion parameter
fclose(f_input);

if (num_periodic!=0){
    repeat_layer = (int)(num_layer/num_periodic);
} else { repeat_layer = 1; }
printf("repeat=%d\n", repeat_layer);

//initialize random seed and gaussian distribution
APPENDIX A. APPENDIX

```c
init_gaussd(init_mrnd, mrnd, random_seed);
init_rnd(random_seed);
deposition(mesh, alpha, phi, max_height, subst_type, diffusion_parameter, ddfilename,
itration, repeat_layer, num_periodic);
}
return 0;
}
```

A.2 Sample input file

VFGS3 Version
256 Size in x direction
256 Size in y direction
40 Size in z direction
2 Number of layer (define layer properties below)
0 Number of periodic layers
20 Height1
1.000000 1.000000 Dispersion of alpha phi
65.000000 65.000000 0.000000 1 10.000000 (min_alpha, max_alpha, alpha_phase, alpha_wave_type, alpha_period)
0.000000 180.000000 180.000000 3 10.000000 (min_phi, max_phi, phi_phase, phi_wave_type, phi_period)
20 Height2
1.000000 1.000000 Dispersion of alpha phi
81.000000 81.000000 0.000000 1 10.000000 (min_alpha, max_alpha, alpha_phase, alpha_wave_type, alpha_period)
0.000000 180.000000 180.000000 3 10.000000 (min_phi, max_phi, phi_phase, phi_wave_type, phi_period)
12345
40 Max height
100000000.000000 Upper limit of the number of generated particles (normalized at alpha=0 deg.)
3383600 Number of generated particles
3383600 Number of deposited particles
0 10 10 1 substrate code substrate height number of protrusion radian
20 1 1 number_of_random_walk_steps temperature face_energy corner_energy

A.3 Sample output from Yorick

A.4 FDTD Source Code: Numerical Dispersion

%Appendix 1
%Question 2.8 Taflove 2006
%Purpose: Calculations of pulse propagation using finite difference scalar
%wave equation (2.16 Taflove 2006), two pulse (Gaussian or rectangular
%pulse) are used as incident wave source, and the courant stability factor
%defined
%Jason Cheung
Figure A.1:

```
%November 8, 2006

clear;

imax=200; % total number of grid cells in 1-D mesh
nmax=input('Input nmax: '); % total number of time steps
S=input('Input ratio of c*dt/dx: '); % Courant stability factor

sourcetype=input('Input source type (1 for Gaussian, 2 for rectangular pulse): ');
xtau=input('Input width of pulse (number of grid cells): ');
ttau=round(xtau/S);

if sourcetype == 1 % Gaussian pulse
ttau=ttau/2;
source(1:nmax)=exp(-(1:nmax)-3*ttau).^2 / ttau^2);
else % rectangular pulse
source(1:ttau)=1.0;
source(ttau+1:nmax)=0.0;
end
```
APPENDIX A. APPENDIX

u(1:imax)=0.0; %wavefunction at new time step (n+1)
un0(1:imax)=0.0; %wavefunction at previous time step (n)
un1(1:imax)=0.0; %wavefunction at previous time step (n-1)

for n=1:nmax
u(1)=source(n); %incident wave condition: hard source at i=1

%regular update of fields at interior grid points
u(2:imax-1)=S^2*(un0(1:imax-2)-2.0*un0(2:imax-1)+un0(3:imax))...
+2.0*un0(2:imax-1)-un1(2:imax-1);

un1=un0; %transfer stored values at time step n to time step n-1
un0=u; %transfer stored values at time step n+1 to time step n

plot(u),xlabel('ith coordinate'),ylabel('Wavefunction u(i)');
title([ 'Time Step=',num2str(n)]);
axis([1 imax-0.5 1.5]);
pause(0.05)
end

legend(['S=',num2str(S)]);

A.5 FDTD Source Code: PEC Simulation

%Appendix 3
%Question 3.4
%Purpose: Model 1D x-directed plane wave propagation using the Yee
%algorithm (3.41a and 3.41c). Source is a unit amplitude Gaussian pulse.
%Assume free space everywhere. Time step is chosen as dt = dx/c. Grid
%terminate at the right boundary for Hy(ih) = 0 to simulate perfect magnetic
%conductor that reflect the magnetic wave
%Jason Cheung
%November 15, 2006

clear;
c=3e8; %speed of light in vacuum
murn=4*pi*1e-7; %permeability of free space
epsn=1.0/(c*c*murn); %permittivity of free space

%Assume relative permittivity and permeability is 1 in free space
eps=[1.0]; %relative permittivity
mur=[1.0]; %relative permeability
%assume conductivity and magnetic loss is 0 in free space
sig = [0.0]; %electric conductivity
sim = [0.0]; %magnetic loss

imax = input('Input total number of grid cell in the 1D mesh:'); %total number of grid cells in 1D mesh
S = input('Input stability factor (S):'); %Courant stability factor
width = input('Input width of grid:');

sourcetype = input('Input source type (1 for Gaussian, 2 for rectangular pulse):');
xtau = input('Input width of pulse (number of grid cells):');
ttau = round(xtau / S);

time = 1e-10; %assume 0.1 ns Gaussian pulse
dt = time / ttau; %time increment
dx = c * dt / S; %space increment

i = round(width / dx) + 1; %number of Ez samples in grid
ih = i - 1; %number of Hy samples in grid
nmax = 2 * round(i * S); %number of time step

%incident wave source condition (hard source)
if sourcetype == 1 % Gaussian pulse
ttau = ttau / 2;
source(1:nmax) = exp(-((1:nmax) - 3*ttau).^2 / ttau^2);
else % rectangular pulse
source(1:ttau) = 1.0;
source(ttau + 1:nmax) = 0.0;
end

%Initialize E and H field to zero
ez(1:ie) = 0.0;
hy(1:ie) = 0.0;

media = length(eps); %number of media with different electrical properties

%updating coefficient
for i = 1:media
efrac = sig(i) * dt / (2.0 * epsn * eps(i));
ca(i) = (1.0 - efrac) / (1.0 + efrac);
cb(i) = dt / (epsn * eps(i) * dx) / (1.0 + efrac);
hfrac = sim(i) * dt / (2.0 * muzn * mur(i));
da(i) = (1.0 - hfrac) / (1.0 + hfrac);
db(i) = dt / (muzn * mur(i) * dx) / (1.0 + hfrac);
end

mediaez(1:ie) = 1; %media pointers for Ez
mediahy(1:ie) = 1; %media pointers for Hy
%time stepping
for n=1:nmax

ez(2:ie)=ca(mediaez(2:ie)).*ez(2:ie)+cb(mediaez(2:ie)).*(hy(2:ie)-hy(1:ie-1));
ez(1)=source(n);

hy(1:ih)=da(mediahy(1:ih)).*hy(1:ih)+db(mediahy(1:ih)).*(ez(2:ie)-ez(1:ih));

subplot(2,1,1), plot(ez,'r'), axis([1 ie -2 2]); ylabeL('Ez');
ti tle(['n=',num2str(n)]);
subplot(2,1,2), plot(hy,'b'), axis([1 ih -6.0e-3 6.0e-3]);
ylabeL('Hy'); xlabel('grid coordinate ');
 pause(0.001)
end

A.6  FDTD Source Code: PMC

%Ques tion 3.7
%Purpo se: Model 2D TMz cylindrical wave propagation in a uniform Yee grid.
%Using 3.41a, 3.41b, 3.41c as time stepping algorithm. Assume cubic yee
%cell, free space everywhere, and a time step dt = S*dx/(csqrt(2)).

%Termin ate the grid in Ez=0 components at its outer boundaries, simulating perfect
%electri c conductor. A single Ez component is located in the center of the
%grid to a specific function, such as a unit step, gaussian pulse, or a
%sinusoidal. Generating a radially-propagating step, a gaussian pulse, or a
%sinusoidal wave in the grid through the Yee algorithm. Visualizations of
%the Ez, Hx, Hy fields of the outgoing wave at a number of time step.

%Jason Cheung
%December 19, 2006

clear;

% . . . . . . . . . . Material Parameters . . . . . . . . . . . .
cc = 2.99792458e8; %speed of li ght in free space
muz = 4.0*pi*1.0e-7; %perm aetivity of free space
epsz = 1.0/(cc*cc*muz); %permittivity of free space
eps = [1.0]; %relative permittivity
sig = [0.0]; %electric conductivity
mur = [1.0]; %relative permeability
sim = [0.0]; %magnetic loss
media = length(eps);

% . . . . . . . . . . Space and Time Parameters . . .
freq = 500e12; %carrier frequency = 500 THz
tau = 1.6e-15; %1/e temporal full width = 3.2 fs
lambda = cc/freq;
omega = 2.0*pi*freq;
dx = lambda/20; %grid sampling density
S=input ('Enter courant stability factor: ');
dt=S*dx/sqrt(2)/cc;

% . . . . . . . . . . Grid Parameters . . . . . . . . . . . . .
length=3e-6; %physical dimension of grid = 3 microns x 3 microns
ie=round(length/dx)+1;
j=ie;
je=ie;

% . . . . . . . . . . Source Parameters . . . . . . . . . . . .
is=round((ie+1)/2); %initialize source at center
js=round((je+1)/2);

ntau=round(tau/dt);
n0=3*ntau;
n_max=2*ie;
sourced(1:nmax)=exp(-(1:nmax)-n0).^2/ntau^2).*sin(omega*(1:nmax)-n0)*dt);

% . . . . . . . . . . Initialization . . . . . . . . . . . . .
ez(1:ie,1:je)=0.0;
hx(1:ie,1:jh)=0.0;
hy(1:ih,1:je)=0.0;

% . . . . . . . . . . Update Coefficients . . . . . . . . . .
for i=1:media
eprop=sig(i)*dt/(2.0*epsz*eps(i));
ca(i)=(1.0-e.prop)/(1.0+e.prop);
cb(i)=dt/(epsz*eps(i)*dx)/(1.0+e.prop);
hprop=sim(i)*dt/(2.0*muz*mur(i));
da(i)=(1.0-h.prop)/(1.0+h.prop);
db(i)=dt/(muz*mur(i)*dx)/(1.0+h.prop);
end
mediaez(1:ie,1:je)=1; %media pointers
mediahx(1:ie,1:jh)=1;
mediahy(1:ih,1:je)=1;

% . . . . . . . . . . Time-Stepping Loop . . . . . . . . . .
for n=1:nmax
% . . . EZ updates . . .
ez(is,js)=source(n);

% . . . HX updates . . .
hx(2:ih,1:jh)=da(mediahx(2:ih,1:jh)).*hx(2:ih,1:jh)+

% . . . HY updates . . .
hy(1:ih,2:jh)=da(mediahy(1:ih,2:jh)).*hy(1:ih,2:jh)+...
db(mediahy(1:ih,2:jh)).*ez(2:ie,2:jh)-ez(1:ih,2:jh));

% ... graphing field distributions in space...
timestep=int2str(n);
subplot(3,1,1),imagesc(ez');
shading flat;
caxis([-0.2 0.2]); colorbar;
axis image; axis xy; axis off,
title(['Ez at n = ',timestep]);
subplot(3,1,2),imagesc(hx');
shading flat;
caxis([-5.0e-4 +5.0e-4]); colorbar;
axis image; axis xy; axis off,
title(['Hx at n = ',timestep]);
subplot(3,1,3),imagesc(hy');
shading flat;
caxis([-5.0e-4 +5.0e-4]); colorbar;
axis image; axis xy; axis off,
title(['Hy at n = ',timestep]);
pause(0.0005)
end

A.7 FDTD Memory requirement

Let N be the total number of grid cells in a 3D TMz FDTD computational domain. The main area contains a finite number of distinct dielectric mediums, with magnetic properties constant throughout the sampling region. Assuming a memory limit of 10000 MB for the simulation process. The largest 3D model that can be simulated is calculated as follow (1 word = 4 byte)

6 vector field components per grid cells x N x 2 updating coefficients per vector field component = 12N words = 72N bytes

72N bytes = 10000 x 10^6 N = 139 x 10^6 grid cells 139 millions grid cells
A.7.1 Time steps calculation

Assume source is a Gaussian pulse with decay characteristic 1/e of maximum, carrier pulse frequency $f_0 = 500 \text{ THz}$, and 1/e temporal full width $\tau = 1.6 \text{ fs}$.

$$f = f_0 + \frac{1}{\pi \tau}$$

$$\lambda_0 = \frac{c}{f_0} = 0.6 \mu m$$

$$f = 500 \text{ THz} \left( \frac{1}{\pi (1.6 \text{ fs})} \right) = 700 \text{ THz}$$

$$\lambda = \frac{c}{f} = 0.43 \mu m$$

$$\Delta x = \frac{\lambda_0}{18} = 33 \text{ nm}$$

$$N_\lambda = \frac{\lambda}{\Delta x} = 12.9$$

A.7.2 Boundary condition for FDTD simulation

1D

$E_z(i) = 0$ at $i = 0$ and $i = i_e$

2D

$E_x(i,j) = 0$ at $i = 0$ and $i = i_b$

$E_y(i,j) = 0$ at $j = 0$ and $j = j_b$

3-D

$E_x(i,j,k) = 0$ on the $j = 1, j = j_b, k = 1, \text{ and } k = k_b$

$E_y(i,j,k) = 0$ on the $i = 1, i = i_b, k = 1, \text{ and } k = k_b$

$E_z(i,j,k) = 0$ on the $i = 1, i = i_b, j = 1, \text{ and } j = j_b$