

# **Kinetic Monte Carlo Simulation of the Growth of Gold Nanostructures on a Graphite Substrate**

Christina Hester Claassens

A thesis submitted for the fulfillment of the requirements for the degree

**PHILOSOPHIAE DOCTOR**

in the

Department of Physics  
Faculty of Natural and Agricultural Science

at the

University of the Free-State

Bloemfontein

June 2006

**Supervisor: Dr. M.J.H. Hoffman**

**Co-supervisor: Prof. J.J. Terblans**



**This thesis is dedicated to my parents. God has truly blessed me by entrusting me in your care.**

## ACKNOWLEDGEMENTS

**I would like to express my sincere gratitude and deepest appreciation to the following people:**

- My supervisors, for their support, assistance and kind words during the tenure of my studies, in specific to Dr. Matie Hoffman.
- A special thanks to Mr. Etienne Wurth, for his assistance with calculations on the computers in the Department of Physics when I left the university.
- My colleagues at Mintek for their support and assistance.
- A special thanks to Dr. Daven Compton, for his guidance and support.
- Dr. Elma van der Lingen and Dr. Daven Compton for presenting me with an opportunity to pursue a new career path.
- The sponsors of Project AuTEK and Mintek for financial support.
- The NRF for financial support.
- Above all, to my heavenly Father, who embraced me with His love, compassion and forgiveness during the most difficult time in my life. I owe all to Him.

## ABSTRACT

Nanotechnology has, without a doubt, ushered in a new era of technological convergence and holds the promise of making a profound impact on the way research in physics, chemistry, materials science, biotechnology etc. are conducted. The novel properties of materials at the nanoscale (or nanostructures) make them useful in a variety of applications, from catalysis to the medical field and electronics industry. However, to exploit these properties at the nanoscale, precise control over the morphology and size of nanostructures is required. One strategy that may be explored to tailor nanostructure morphology and size is vapour deposition. A lot of further insight can be gained from computer simulations of the processes governing the growth of nanostructures during vapour deposition. A method that shows promise in simulating thin film growth through vapour deposition is kinetic Monte Carlo (KMC). Therefore, in this study, a KMC model was developed to describe growth through vapour deposition. A gold on graphite system was simulated to test the model. In this KMC model, substantial effort was devoted to developing the model in different stages, each stage being more robust than the previous one. The assumptions made at each stage and possible artefacts (unphysical consequences) arising from them are discussed in order to distinguish real physical effects from artificial ones. In the model, data structures, search algorithms and a random number generator were developed and employed in an object-orientated code to simulate the growth. Several simulations were performed at different growth conditions for each of the stages. The results are interpreted based on the kinetic constraints imposed during the growth.

**KEY WORDS: Kinetic Monte Carlo, Gold Nanostructures, Atomic Deposition**

## SAMEVATTING

Nanotegnologie het ongetwyfeld 'n nuwe era van tegnologiese konvergensie ingelui en hou die belofte in om 'n beduidende impak te hê op die wyse waarop navorsing in fisika, chemie, materiaalwetenskap, biotegnologie ens. onderneem word. Die unieke eienskappe van materiale op die nanoskaal (of nanostrukture) maak hulle in 'n verskeidenheid van toepassings, van katalise tot die mediese veld en elektroniese industrie, bruikbaar. Om hierdie eienskappe op die nanoskaal aan te wend, word presiese beheer oor die morfologie en grootte van die nanostrukture vereis. Een strategie om die nanostruktuur, morfologie en grootte te verander, wat ondersoek kan word is atoomdeposisie. Heelwat verdere insig oor die prosesse wat die groei van nanostrukture tydens atoomdeposisie beheer, kan vanuit rekenaarsimulasies verkry word. 'n Metode wat potensiaal toon om dunfilmgroei deur atoomdeposisie te simuleer, is kinetiese Monte Carlo (KMC). In hierdie studie is 'n KMC model gevolglik ontwikkel om groei tydens atoomdeposisie te beskryf. Om die model te toets is 'n goud op grafiet sisteem gesimuleer. In hierdie KMC model is beduidende moeite gedoen om die model in verskillende stadiums te ontwikkel, elke stadium meer robuust as die vorige. Die aannames wat tydens elke stadium gemaak is en die moontlike kunsmatige (of onfisiese) gevolge wat as gevolg daarvan ontstaan, word bespreek om te onderskei tussen werklike fisiese effekte en kunsmatiges. In die model is datastrukture, soek algoritmes en 'n ewekansige getal genereerder ontwikkel en in 'n objek-georiënteerde kode ingespan om groei te simuleer. Verskeie simulasies is by verskillende groei voorwaardes vir elk van die stadiums uitgevoer. Die resultate word geïnterpreteer met in agname van die kinetiese beperkinge teenwoordig gedurende die groei.

**SLEUTELWOORDE:** Kinetiese Monte Carlo, Goud Nanostrukture, atoomdeposisie



# TABLE OF CONTENTS

<b>CHAPTER 1</b> .....	<b>1</b>
1.1 BACKGROUND .....	1
1.2 MOTIVATION OF STUDY .....	5
1.3 LAYOUT OF THESIS .....	7
<b>CHAPTER 2</b> .....	<b>8</b>
2.1 CRYSTAL GROWTH.....	9
2.1.1 <i>Thermodynamic considerations</i> .....	9
2.2 VALIDITY OF THE WULFF-KAISCHEW THEOREM FOR NANOSTRUCTURES.....	21
2.3 NANOSTRUCTURE GROWTH.....	23
<b>CHAPTER 3</b> .....	<b>26</b>
3.1 LENGTH SCALES IN GROWTH.....	26
3.2 POTENTIAL ENERGY SURFACE.....	31
3.3 ELECTRONIC AND ATOMIC SCALE .....	33
3.4 MICROSCOPIC SCALE .....	38
3.5 MESOSCOPIC SCALE.....	40
3.5.1 <i>Stochastic processes</i> .....	41
3.5.2 <i>The conditional probability</i> .....	42
3.5.3 <i>The Markov process</i> .....	43
3.5.4 <i>The Master Equation</i> .....	44
3.5.5 <i>Solution of the Master Equation</i> .....	46
3.5.6 <i>Transition state theory (TST)</i> .....	52
3.6 MACROSCOPIC SCALE.....	54
3.6.1 <i>Rate equation analysis</i> .....	55
3.6.2 <i>Continuum theories</i> .....	56
<b>CHAPTER 4</b> .....	<b>58</b>
4.1 THE LATTICE GAS ASSUMPTION .....	59
4.1.1 <i>The lattice geometry</i> .....	61
4.2 IDENTIFICATION OF THE RELEVANT PROCESSES IN THE SYSTEM.....	65
4.3 CALCULATION OF PROCESS RATES .....	69
4.4 DATA STRUCTURES FOR THE STORAGE OF THE PROCESS RATES .....	72
4.5 AN ALGORITHM EMPLOYING KMC TO SIMULATE NANOSTRUCTURE GROWTH THROUGH VAPOUR DEPOSITION .....	75
<b>CHAPTER 5</b> .....	<b>79</b>
5.1 PRELIMINARY PREDICTIONS .....	80
5.2 DEVELOPMENT STAGES .....	81
5.2.1 <i>Stage 1</i> .....	87
5.2.2 <i>Stage 2</i> .....	98
5.2.3 <i>Stage 4</i> .....	114
5.3 COMMENTS ON THE KMC METHOD .....	117



<b>CHAPTER 6</b> .....	<b>119</b>
6.1    CONCLUSIONS.....	119
6.2    FUTURE WORK .....	123
<b>APPENDIX A: DERIVATION OF THE MASTER EQUATION</b> .....	<b>125</b>
<b>APPENDIX B: DATA STRUCTURES</b> .....	<b>128</b>
B.1    THE LIST ADT .....	129
B.2    THE TREE ADT .....	130
<b>APPENDIX C: CLASS STRUCTURES</b> .....	<b>139</b>
<b>APPENDIX D: RANDOM NUMBERS</b> .....	<b>146</b>
D.1    THE MERSENNE TWISTER RNG .....	150
<b>APPENDIX E: CALCULATION OF ACTIVATION ENERGY BARRIERS WITH MOLECULAR DYNAMICS</b> .....	<b>151</b>
E.1    CALCULATION OF THE ACTIVATION ENERGY BARRIERS ON GRAPHITE .....	152
E.2    CALCULATION OF THE ACTIVATION ENERGY BARRIERS ON AU(111).....	153
<b>APPENDIX F: SUPPLEMENTARY CODE</b> .....	<b>157</b>
<b>REFERENCES</b> .....	<b>191</b>
<b>BIOGRAPHY</b> .....	<b>201</b>
<b>PUBLICATIONS AND CONFERENCES</b> .....	<b>202</b>

# LIST OF FIGURES

FIGURE 1.1: SCHEMATIC ILLUSTRATION OF THE TWO APPROACHES FOLLOWED IN THE MANUFACTURING OF NANOSTRUCTURED MATERIALS. ....	4
FIGURE 2.1: THE EQUILIBRIUM SHAPE OF A TWO-DIMENSIONAL FACETTED CRYSTAL. THE “AREA” OF THE F-TH FACET IS $A_f$ AND $h_f = \mathbf{R} \cdot \mathbf{N}_f$ IS THE DISTANCE FROM THE CENTRE OF SYMMETRY WITH $\mathbf{R}$ ANY POINT ON THE CRYSTAL SURFACE. ....	11
FIGURE 2.2: EQUILIBRIUM SHAPES AT 0 K OF AN (A) FCC TRUNCATED OCTAHEDRON AND A (B) BCC RHOMBIC DODECAHEDRON. ....	13
FIGURE 2.3: SCHEMATIC REPRESENTATION OF THE EQUILIBRIUM SHAPE OF A SUPPORTED POLYHEDRON CRYSTAL. THE SHAPE OF THE FREE CRYSTAL IS TRUNCATED AT THE INTERFACE BY AN AMOUNT $\Delta h_s$ WHICH IS PROPORTIONAL TO THE ADHESION ENERGY. THE $h$ 'S REPRESENT THE DISTANCE FROM THE CENTRE OF THE CRYSTAL TO THE DIFFERENT FACETS. THE SURFACE FREE ENERGIES OF THE SUBSTRATE, DEPOSITED FILM AND INTERFACE ARE GIVEN BY $\gamma_s$ , $\gamma_d$ AND $\gamma_{int}$ RESPECTIVELY. ....	14
FIGURE 2.4: SCHEMATIC ILLUSTRATION OF (A) LATTICE-MATCHED, (B) STRAINED AND (C) RELAXED HETEROEPITAXIAL STRUCTURES. ....	15
FIGURE 2.5: SCHEMATIC ILLUSTRATION OF A DROPLET IN EQUILIBRIUM ON A SURFACE. THE RADIUS OF THE DROPLET IS $R$ AND THE DROPLET IS TRUNCATED BY AN AMOUNT $\Delta h$ . THE SURFACE FREE ENERGIES OF THE SUBSTRATE, DEPOSITED FILM AND INTERFACE ARE GIVEN BY $\gamma_s$ , $\gamma_d$ AND $\gamma_{int}$ RESPECTIVELY. ....	16
FIGURE 2.6: SCHEMATIC ILLUSTRATION OF THE THREE EQUILIBRIUM GROWTH MODES. (A) FRANK-VAN-DER-MERWE GROWTH (B) VOLMER-WEBER GROWTH AND (C) STRANSKI-KRASTANOV GROWTH. ....	17
FIGURE 2.7: THE THREE EQUILIBRIUM GROWTH MODES AS A FUNCTION OF MISMATCH AND SURFACE ENERGY RATIO. THE SURFACE FREE ENERGIES OF THE SUBSTRATE, DEPOSITED FILM AND INTERFACE ARE GIVEN BY $\gamma_s$ , $\gamma_d$ AND $\gamma_{int}$ RESPECTIVELY. THE LATTICE CONSTANTS OF THE DEPOSITED FILM AND THE SUBSTRATE ARE GIVEN BY $a_d$ AND $a_s$ RESPECTIVELY. ....	18
FIGURE 2.8: SCHEMATIC ILLUSTRATION OF THE VARIOUS KINDS OF FACETS (S, F AND K) ON A GROWING CRYSTAL. ....	19
FIGURE 2.9: EQUILIBRIUM SHAPE OF Pd NANOSTRUCTURE SUPPORTED ON A MgO (100) SURFACE. THE SIZE OF THE STRUCTURE IS LARGER THAN 10 NM. ....	22
FIGURE 2.10: THE DIFFERENT ATOMISTIC PROCESSES FOR ADATOMS ON A SURFACE: (A) DEPOSITION, (B) DIFFUSION ON TERRACES, (C) DIFFUSION ALONG A STEP EDGE, (D) DISSOCIATION FROM A STEP EDGE, (E) INTERLAYER DIFFUSION (F) DESORPTION AND (G) CAPTURE BY A STEP EDGE. ....	24
FIGURE 2.11: REPRESENTATION OF THE TIMESCALE OVER WHICH PROCESSES OCCUR DURING EPITAXIAL GROWTH. ....	25
FIGURE 3.1: A SCHEMATIC ILLUSTRATION OF THE SPATIAL AND TEMPORAL SCALES ACHIEVABLE BY VARIOUS SIMULATION APPROACHES: DFT/QMC (DENSITY FUNCTIONAL THEORY/QUANTUM MONTE CARLO); CLASSICAL MD (MOLECULAR DYNAMICS); KMC (KINETIC MONTE CARLO); RE (RATE EQUATION ANALYSIS), AND MULTISCALE ENCOMPASSING ALL THE PRECEDING TIME SCALES. ....	29
FIGURE 3.2: SCHEMATIC OF A TWO DIMENSIONAL SLICE $U(X, Z)$ THROUGH THE MULTI-DIMENSIONAL POTENTIAL ENERGY HYPERSURFACE FOR THE ADATOM SURFACE INTERACTION. $Z$ IS THE DISTANCE TO THE SURFACE AND $X$ IS THE ADATOM COORDINATE ALONG A GIVEN SURFACE DIRECTION. ....	32
FIGURE 3.3: A SCHEMATIC INTERPRETATION OF A STOCHASTIC PROCESS, $Y$ , AS A FUNCTION OF STOCHASTIC VARIABLES $\{X(t_i)\}$ . AT SUCCESSIVE TIMES, THE MOST PROBABLE VALUES OF $Y$ HAVE BEEN DRAWN AS HEAVY DOTS. THE MOST PROBABLE TRAJECTORY CAN BE SELECTED FROM SUCH A PICTURE. TWO OR MORE TRAJECTORIES CAN OCCUR WITH EQUAL PROBABILITY. ....	42
FIGURE 3.4 SCHEMATIC ILLUSTRATION OF THE DIFFERENCE BETWEEN THE STEADY STATE CONDITION PROPERTY AND DETAILED BALANCE. THE LENGTHS OF THE ARROWS ARE PROPORTIONAL TO THE TRANSITION RATE). IN (A) STEADY STATE IS SATISFIED BUT DETAILED BALANCE NOT, WHEREAS IN (B) BOTH STEADY STATE AND DETAILED BALANCE ARE SATISFIED. ....	46

FIGURE 3.5: SCHEMATIC ILLUSTRATION OF A ONE-STEP PROCESS. ONLY JUMPS BETWEEN ADJACENT STATES (LABELLED N) ARE ALLOWED. THE PROBABILITY PER UNIT TIME FOR JUMPS IN THE FORWARD AND REVERSE DIRECTION IS DENOTED BY $G_N$ AND $H_N$ RESPECTIVELY. ....	48
FIGURE 3.6: SCHEMATIC ILLUSTRATION OF THE POISSON PROCESS. ....	49
FIGURE 3.7: SCHEMATIC ILLUSTRATION OF THE LOWEST FREE ENERGY PATH FOR A THERMALLY ACTIVATED JUMP OF AN ADATOM FROM I TO J OVER THE SADDLE POINT $X_0$ . ....	54
FIGURE 4.1: SCHEMATIC ILLUSTRATION OF THE LATTICE GAS ASSUMPTION. A TWO-DIMENSIONAL LATTICE IS CONSTRUCTED WHICH CONSTITUTES POSSIBLE ADSORPTION SITES ON THE SUBSTRATE. DIFFUSION PROCESSES ARE REPRESENTED BY HOPS BETWEEN THE VARIOUS LATTICE SITES. THE SYSTEM SIZE IS $L \times L$ . ....	60
FIGURE 4.2: SCHEMATIC ILLUSTRATION OF THE LATTICE GAS ASSUMPTION. A TWO-DIMENSIONAL LATTICE IS CONSTRUCTED WHICH CONSTITUTES POSSIBLE ADSORPTION SITES ON THE SUBSTRATE. DIFFUSION PROCESSES ARE REPRESENTED BY HOPS BETWEEN THE VARIOUS LATTICE SITES. THE SYSTEM SIZE IS $L \times L$ . AS OVERHANGS AND BULK VACANCIES ARE NEGLECTED, THE SURFACE IS FULLY CHARACTERIZED BY AN INTEGER ARRAY OF HEIGHT VARIABLES ABOVE A SQUARE LATTICE SUBSTRATE. ....	60
FIGURE 5.1: POTENTIAL ENERGY CURVE FOR THE SUTTON-CHEN POTENTIAL. ....	82
FIGURE 5.2 : RATES FOR DIFFERENT PROCESSES ON STEP OF A Au ISLAND ON GRAPHITE. THE DIFFERENT CURVES APPLY TO JUMPS CHARACTERISED BY THE CHANGE IN THE NUMBER OF NEAREST NEIGHBOURS IN THE SURFACE PLANE. ....	85
FIGURE 5.3 : RATES FOR DIFFERENT PROCESSES ON AN A-STEP OF A Au ISLAND ON Au (111). THE DIFFERENT CURVES APPLY TO JUMPS CHARACTERISED BY THE CHANGE IN THE NUMBER OF NEAREST NEIGHBOURS IN THE SURFACE PLANE. ....	86
FIGURE 5.4 : RATES FOR DIFFERENT PROCESSES ON THE B-STEP OF A Au ISLAND ON Au (111). THE DIFFERENT CURVES APPLY TO JUMPS CHARACTERISED BY THE CHANGE IN THE NUMBER OF NEAREST NEIGHBOURS IN THE SURFACE PLANE. ....	86
FIGURE 5.5: AVERAGE ISLAND SIZE (NUMBER OF SITES) FOR DIFFERENT TEMPERATURES AT A GIVEN DEPOSITION RATE. ....	91
FIGURE 5.6: AVERAGE ISLAND DENSITY FOR DIFFERENT TEMPERATURES AND DEPOSITION RATES. ....	91
FIGURE 5.7: AVERAGE NANOSTRUCTURE SIZE (NUMBER OF SITES) FOR DIFFERENT TEMPERATURES AT A GIVEN DEPOSITION RATE. ....	95
FIGURE 5.8: AVERAGE NANOSTRUCTURE DENSITY FOR DIFFERENT TEMPERATURES AND DEPOSITION RATES. ....	95
FIGURE 5.9: AVERAGE ISLAND SIZE (NUMBER OF SITES) FOR DIFFERENT TEMPERATURES AT A GIVEN DEPOSITION RATE WITH A COVERAGE OF 0.25 ML AND 50 NUCLEATION SITES. ....	102
FIGURE 5.10: AVERAGE ISLAND DENSITY FOR DIFFERENT TEMPERATURES AND DEPOSITION RATES WITH A COVERAGE OF 0.25 ML AND 50 NUCLEATION SITES. ....	102
FIGURE 5.11: ISLAND DENSITY FOR DIFFERENT TEMPERATURES AND DEPOSITION RATES WITH A COVERAGE OF 0.05 ML WITH NO NUCLEATION SITES. ....	104
FIGURE 5.12 : TRIANGULAR NANOSTRUCTURE MORPHOLOGIES FOR A DEPOSITION RATE OF $5 \times 10^{-4}$ ML/S AT 600 K AND A COVERAGE OF 0.4 ML. THE TRIANGULAR MORPHOLOGY IS AN ARTIFICIAL EFFECT DUE TO THE ASSUMPTION MADE IN THE MODEL THAT ONLY JUMPS TO FCC SITES ARE ALLOWED. ....	107
FIGURE 5.13: THE NUMBER OF STEPS NEEDED FOR COMPLETION OF A SIMULATION (DEPOSITION OF 1000 ATOMS IN A FULL DIFFUSION MODEL) DEPENDS ON THE TEMPERATURE AND DEPOSITION RATE. ....	117
FIGURE 5.14: ILLUSTRATION OF THE EXPONENTIAL INCREASE OF COMPUTATIONAL TIME WITH TEMPERATURE FOR MONOLAYER DEPOSITION USING A DEPOSITION RATE OF 0.1 ML/S (DEPOSITION OF 1000 ATOMS WITH A FULL DIFFUSION MODEL). ....	118



# LIST OF TABLES

TABLE 1.1: EXAMPLES OF REDUCED-DIMENSIONAL MATERIAL GEOMETRIES, DEFINITIONS OF THEIR DIMENSIONALITY AND OF THE ASSOCIATED TYPE OF CONFINEMENT. ....	3
TABLE 5.1: SUMMARY OF ACTIVATION ENERGY, $\Delta E$ , FOR THE DIFFUSION OF A GOLD ADATOM AROUND AN ISLAND EDGE ON GRAPHITE AND Au(111). ONLY THOSE PROCESSES DISCUSSED IN CHAPTER 4, TABLE 4.1 ARE SHOWN. THE PROCESSES ARE IDENTIFIED BY $N_i$ AND $N_f$ , MEANING THAT, FOR THE STEP DIFFUSION PROCESS, WITH $N_i = 2$ AND $N_f = 2$ , THE PROCESS CAN BE REPRESENTED BY $2 \rightarrow 2$ .....	84
TABLE 5.2: ISLAND MORPHOLOGIES FOR MONOLAYER GROWTH AT DIFFERENT TEMPERATURES AND DEPOSITION RATES USING THE ASSUMPTION THAT ONLY THE MOST RECENTLY DEPOSITED ATOM IS MOBILE AT A GIVEN TIME AND CONSIDERING ONLY THE LARGEST PROBABILITY IN THE SYSTEM. THE COVERAGE IS 0.25 ML. A TOTAL NUMBER OF 50 NUCLEATIONS SITES WERE RANDOMLY PLACED ON THE SUBSTRATE.....	90
TABLE 5.3: NANOSTRUCTURE MORPHOLOGIES FOR MULTILAYER GROWTH AT DIFFERENT TEMPERATURES AND DEPOSITION RATES USING THE ASSUMPTION THAT ONLY THE MOST RECENTLY DEPOSITED ATOM IS MOBILE AT A GIVEN TIME AND CONSIDERING ONLY THE LARGEST PROBABILITY IN THE SYSTEM. THE COVERAGE IS 0.4 ML. A TOTAL NUMBER OF 25 NUCLEATIONS SITES WERE RANDOMLY PLACED ON THE LATTICE.....	94
TABLE 5.4: ISLAND MORPHOLOGY FOR VARIOUS TEMPERATURES AND DEPOSITION RATES WITH A COVERAGE OF 0.25 ML WITH 50 NUCLEATION SITES. PROCESS SELECTION WAS DONE BY COMPARING THE WEIGHTED PROBABILITY OF EACH PROCESS IN THE SYSTEM WITH A RANDOM NUMBER IN THE INTERVAL [0, 1]. ONLY THE MOST RECENTLY DEPOSITED ATOM WAS ALLOWED TO MOVE BETWEEN DEPOSITIONS. ....	101
TABLE 5.5: ISLAND MORPHOLOGY FOR VARIOUS TEMPERATURES AT A DEPOSITION RATE OF $5 \times 10^{-4}$ ML/S AND COVERAGE OF 0.05 ML WITH NO NUCLEATION SITES.....	103
TABLE 5.6: NANOSTRUCTURE MORPHOLOGIES FOR DIFFERENT TEMPERATURES AND DEPOSITION RATES AND COVERAGES OF 0.4 ML. PROCESS SELECTION WAS DONE BY COMPARING THE WEIGHTED PROBABILITY OF EACH PROCESS IN THE SYSTEM WITH A RANDOM NUMBER IN THE INTERVAL [0, 1]. ONLY THE MOST RECENTLY DEPOSITED ATOM WAS ALLOWED TO MOVE BETWEEN DEPOSITIONS. ....	106
TABLE 5.7: ISLAND MORPHOLOGIES FOR DIFFERENT DEPOSITION RATES AND TEMPERATURES (MONOLAYER GROWTH) VARIED AS INDICATED. THE TOTAL COVERAGE AFTER DEPOSITION WAS 0.2 ML. ONLY PART OF THE SIMULATED SURFACES IS SHOWN.....	112
TABLE 5.8: NANOSTRUCTURE MORPHOLOGIES (MULTILAYER GROWTH) FOR DIFFERENT DEPOSITION RATES AND TEMPERATURES VARIED AS INDICATED. THE TOTAL COVERAGE AFTER DEPOSITION WAS 0.2 ML. ONLY PART OF THE SIMULATED SURFACES IS SHOWN.....	113



# CHAPTER 1

## Introduction

### 1.1 Background

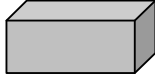
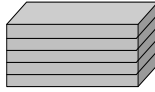
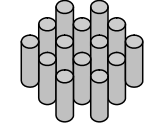
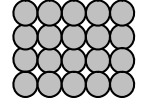
The earliest impetus towards nanotechnology was given by the Nobel-prize winning physicist, Richard Feynmann, in his visionary lecture *“There is plenty of room at the bottom”* delivered at the American Physical Society (APS) meeting at Cal Tech in 1959 in which he said, *“The problems of chemistry and biology can be greatly helped if our ability to see what we are doing, and to do things on an atomic level, is ultimately developed – a development which I think cannot be avoided”* [1]. Indeed, this vision was realized in the late 1980’s with the invention of the scanning tunnelling microscope (STM) [2, 3] which is an instrument that exploits the quantum mechanical tunnelling current to generate atomically resolved images

of electronic states. In the simplest terms, nanotechnology deals with the size selected collection of a few atoms to a few tens of thousands of atoms. Structures on this scale exhibit unique and novel properties, vastly different from microstructures which already, to a reasonable extent, approximate that of the infinite bulk. These properties originate either from spatial confinement of a physical entity inside a specified volume (for example, the confinement of electronic wave functions inside a region with a size smaller than the electron mean free path) or from the significant volume fraction of material located near surfaces, interfaces or domain walls [4, 5]. Fabrication and manipulation of these nanostructures is therefore a fundamentally exciting and technological relevant area of research.

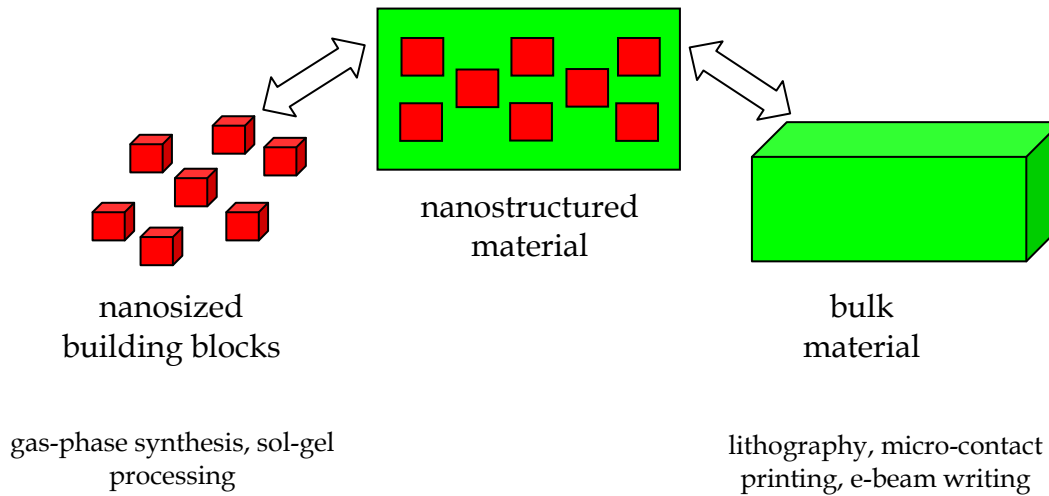
Nanostructures are usually classified according to their dimensionality which can be defined as the number of orthogonal directions  $L_x, L_y, L_z$  smaller than the nanoscale dimension  $L_o$  (see Table 1.1) below which size effects become important. They may be manufactured through one of two disparate approaches; namely the top-down and bottom-up techniques [6] as illustrated in Figure 1.1. The former makes direct use of lithographic techniques such as STM related nanolithography [7], electron beam writing [8] and micro-contact printing [9]. Although these methods can achieve high spatial resolution, they are quite slow when used to design nanoscale features. Bottom-up approaches, on the other hand, aim to guide the assembly of atomic and molecular constituents into organized nanostructures through processes inherent to the manipulated system.



*Table 1.1: Examples of reduced-dimensional material geometries, definitions of their dimensionality and of the associated type of confinement.*

Lengths	Confinement	Dimensionality	Type	Illustration
$L_{X,Y,Z} > L_0$	None	No Nanostructures	Bulk Material	
$L_{X,Y} > L_0 > L_Z$	1D	2D Nanostructures	Wells	
$L_X > L_0 > L_{Y,Z}$	2D	1D Nanostructures	Wires	
$L_0 > L_{X,Y,Z}$	3D	0D Nanostructures	Dots	

A number of self-assembly techniques have been reported for fabricating nanoscale structures and these are broadly divided into two categories, namely gas phase synthesis and sol-gel processing. Gas phase synthesis is based on evaporation and condensation in a sub-atmospheric inert-gas environment or vacuum (for example vapour deposition [10], laser ablation [11], spray pyrolysis [12], plasma etching [13] etc.) whereas sol-gel processing [14] is a wet chemical synthesis approach that can be used to generate nanostructures by gelation, precipitation, and hydrothermal treatment.



*Figure 1.1: Schematic illustration of the two approaches followed in the manufacturing of nanostructured materials.*

Although great advances have already been made with both these methods in the development of protocols that can be used to synthesise nanostructures within a narrow size and shape distribution, a detailed fundamental understanding of the processes governing synthesis is still lacking in most cases. Consequently, substantial effort is put into establishing models that can foster the development of the above mentioned synthesis protocols [15-17]. Ultimately, these models will be able to define the design of the nanostructures and as such reduce the number of design iterations, experiments and tools required for design.

Sol-gel processing of nanostructures, although the most promising in terms of scale-up strategies and thus commercial exploitation of nanostructures, are in many ways far more complex to model than gas phase synthesis. Numerous factors contribute to this, of which the large number of reactions occurring during processing, the difficulty in obtaining accurate activation energies and

rate constants and the time scale of the synthesis process are but a few. On the other hand, several simulation methodologies with different degrees of crudeness have already been employed to study gas-phase synthesis and some have been particularly successful in describing observations made on nanostructure growth in molecular-beam epitaxy with the kinetic Monte Carlo method shown to be able to reach experimental time-scales [18, 19].

Metallic nanostructures, especially the coinage metals, have been extensively studied for the past decade [20-22], because of their unique physical and chemical properties such as a strong optical absorption in the visible region [23]. Gold nanostructures, in particular, have received substantial attention over the ages, and date back to the pioneering work of Faraday on the synthesis of gold hydrosols [24]. Gold nanostructures can potentially be utilized in several new technologies [25], ranging from electronics, to catalysis, to the chemical industry, to biotechnology and more. Applications include biosensors and DNA labelling [26, 27], nanoelectronics [29, 30], calorimetric [31] etc. However, all of these applications demand nanostructures with a well-defined size and shape.

## **1.2 Motivation of study**

Clearly, there is a strong need to synthesise nanostructures with a well-defined size and shape in order to exploit the unique properties presented on this scale which can ultimately result in industrial and commercial applications. Moreover, precise control over the growth of gold nanostructures can be useful for various technologies and especially holds great promise for the bio-medical community. Even though valuable information on the mechanisms responsible for nanostructure growth can be extracted from experiment, this can be a very time-

consuming and costly process. However, one can resort to modelling and perform "computer experiments" to aid in elucidating the mechanisms responsible for nanostructure growth and to provide guidelines for tailoring nanostructure size and shape. This is also the approach followed in this work. More specific, the growth of gold nanostructures via gas-phase synthesis (physical vapour deposition) was studied on a model substrate, graphite. The reasons for choosing graphite as substrate will become evident in Chapter 4.

The primary objectives for this work can therefore be summarised as follow:

- Establish a model for the simulation of gold nanostructure growth on a graphite substrate through vapour/atomic deposition.
- This model should make provision for all relevant time scales (excluding atomic vibrations) so that a real experiment can be simulated.
- From this model, the mechanisms involved in the growth process should be determined and deductions made on how they influence the nanostructure size and shape.
- The model has to be designed in such a way that it can be extended to vapour deposition experiments with different materials (i.e other than gold and graphite).

It must be emphasised that the focus of this work was not conduct a detailed study to elucidate the exact energy barriers for the processes involved in nanostructure growth (discussed in Chapter 2); rather it was to develop a model to perform simulations on a mesoscopic or microscopic scale. It was also

endeavoured that this model should be sophisticated enough so that further development and enhancement can easily follow.

### **1.3 Layout of thesis**

*Chapter 1* gives a general introduction to nanostructure synthesis and provides the motivation of this study.

*Chapter 2* describes the self-assembled growth of nanostructures during vapour deposition, thermodynamic equilibrium and the kinetic constraints that are imposed during growth.

*Chapter 3* gives an overview of some of the simulation methods that can be used to model the growth of nanostructures with specific emphasis placed on the method used in this study, namely the kinetic Monte Carlo (KMC) method.

*Chapter 4* describes the implementation of the KMC method in terms of data structures, search algorithms, process identification and how these are combined into an algorithm that can be computer coded. The application of method for this study is also discussed.

*Chapter 5* gives a summary and interpretation of the results obtained using the KMC method, as implemented in this study, for the simulation of gold nanostructure growth on graphite.

*Chapter 6* concludes the results and some suggestions are made on future work.

## CHAPTER 2

### Self-assembled Growth

Self-assembly has become an increasing hallmark in the field of nanostructure synthesis and is essentially based on growth phenomena [31-33]. In self-assembled growth, atoms and/or molecules are deposited on a substrate, which can reside in vacuum, atmosphere or solution. Nanostructures (or larger crystals) consequently evolve on the substrate as a result of numerous competing processes. The equilibrium morphology of crystals has been well established more than 100 years ago [34] and can be insightful in understanding the equilibrium morphology of nanostructures. This chapter therefore gives a general discussion of crystal growth morphology, under equilibrium as well as non-equilibrium conditions, in order to lay the foundation for subsequent chapters.

## 2.1 Crystal growth

### 2.1.1 Thermodynamic considerations

The equilibrium morphology of crystals, as determined by thermodynamics, can be obtained by minimizing the total surface free energy of the crystal at a constant volume and temperature [35]. For isotropic surface free energies (as in the case of liquids) the crystal morphology will be spherical and the chemical potential constant everywhere on the surface. The chemical potential is given by [35]

$$\mu = \mu_0 - \frac{V}{N} \left[ \frac{\partial}{\partial x} \left( \frac{\partial \phi}{\partial z_x} \right) + \frac{\partial}{\partial y} \left( \frac{\partial \phi}{\partial z_y} \right) \right], \quad (2.1)$$

with  $\mu_0$  the chemical potential inside the crystal,  $V$  the volume of the crystal,  $N$  the number of atoms in the crystal,  $z_x = \frac{\partial z}{\partial x}$  and  $z_y = \frac{\partial z}{\partial y}$  the partial derivatives of the height,  $z$ , over the coordinates of the  $(x, y)$  plane respectively and  $\phi(z_x, z_y) = \sigma(z_x, z_y) \sqrt{1 + z_x^2 + z_y^2}$  the projected surface free energy. The surface free energy,  $F$ , is defined as [35]

$$F_{surf} = \iint \sigma(z_x, z_y) dS, \quad (2.2)$$

with  $\sigma(z_x, z_y)$  the local surface tension integrated over the entire surface  $S$ . The constant chemical potential implies that the surface free energy (given by equation (2.2)) is a minimum at equilibrium. Equivalently, the change in the

surface free energy,  $\Delta F$ , at a constant temperature and volume is given by

$$\Delta F = \mu_2 - \mu_1 = 0. \quad (2.3)$$

The crystal surface can consequently be described by equation (2.1), provided that the functions  $F(z_x, z_y)$  or  $\phi(z_x, z_y)$  are known. Assuming that the units are selected in such a way that the molar volume  $v = V/N = 1$ , and that  $\mu_0 = 0$ , the equation for the crystal surface can be written as

$$\frac{\partial}{\partial x} \left( \frac{\partial \phi}{\partial z_x} \right) + \frac{\partial}{\partial y} \left( \frac{\partial \phi}{\partial z_y} \right) = -\mu = \text{constant}. \quad (2.4)$$

Equation (2.4) can only be applied to crystal morphologies for which derivatives of  $\phi$  exist. This will be the case for rounded crystals since  $\phi$  for faceted crystals are only defined in a discrete set of orientations in the equilibrium morphology (see Figure 2.1). The surface free energy must then be written as a sum of the contributions from the various facets i.e

$$F_{surf} = \sum_f \sigma_f A_f, \quad (2.5)$$

with  $\sigma_f$  the value of  $\sigma(\mathbf{n})$  at  $\mathbf{n} = \mathbf{n}_f$  and  $A_f$  the area of the corresponding facet. The equilibrium morphology can be found by minimizing  $F_{surf}$  at a fixed volume  $V = \frac{1}{2} \sum_f A_f h_f$ , where  $h_f = \max_{\mathbf{R}} \{\mathbf{R} \cdot \mathbf{n}_f\}$  and  $\mathbf{R}$  any point on the crystal surface. A constrained minimization with a Lagrange multiplier  $\lambda$  [36] can be performed to enforce the fixed volume condition

$$\delta(\lambda V + F_{surf}) = \sum_f \left( \frac{\lambda}{2} h_f + \sigma_f \right) \delta A_f = 0. \quad (2.6)$$



This minimization yields

$$\frac{\sigma_f}{h_f} = -\frac{\lambda}{2} . \quad (2.7)$$

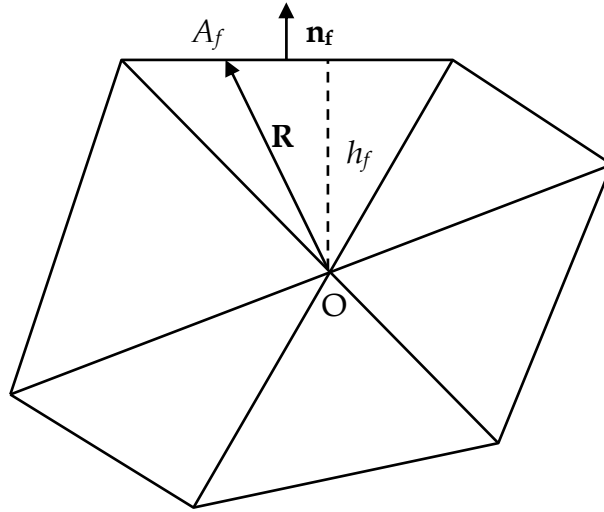


Figure 2.1: The equilibrium shape of a two-dimensional faceted crystal. The “area” of the  $f$ -th facet is  $A_f$  and  $h_f = \mathbf{R} \cdot \mathbf{n}_f$  is the distance from the centre of symmetry with  $\mathbf{R}$  any point on the crystal surface.

It follows that ratio of the surface energy to the distance  $h_f$  from the origin is constant. At this stage,  $\lambda$  is still unknown, but it can be deduced by noting that the variation in free energy at fixed volume can be written as [35]

$$\delta F_{surf} = \mu \delta N , \quad (2.8)$$

with  $\delta N$  the variation in particle quantity  $N$ . According to equation (2.6),

$\delta(\lambda V + F_{surf}) = 0$ , therefore

$$\delta F_{surf} = \mu \delta N = -\delta V \lambda = -v \delta N \lambda, \quad (2.9)$$

with  $v = V / N$  the volume per particle. It subsequently follows that

$$\lambda = -\mu / v. \quad (2.10)$$

An expression for the set of  $\{h_f\}$ , which describes the equilibrium morphology, can then be obtained by combining equation (2.7) and (2.10) resulting in

$$h_f = \frac{2v\sigma_f}{\mu}. \quad (2.11)$$

Examples of equilibrium morphologies are the truncated octahedron and rhombic dodecahedron for fcc and bcc structures respectively, as shown in Figure 2.2. These polyhedron shapes are only valid when the surface anisotropy is maximal, which is the case at 0 K. At higher temperatures, the crystal equilibrium morphology is more rounded and eventually becomes completely spherical at the melting point [35]. In addition to the above methodology, a geometric construction, namely the Wulff construction [35], can be used to find the equilibrium crystal morphology.

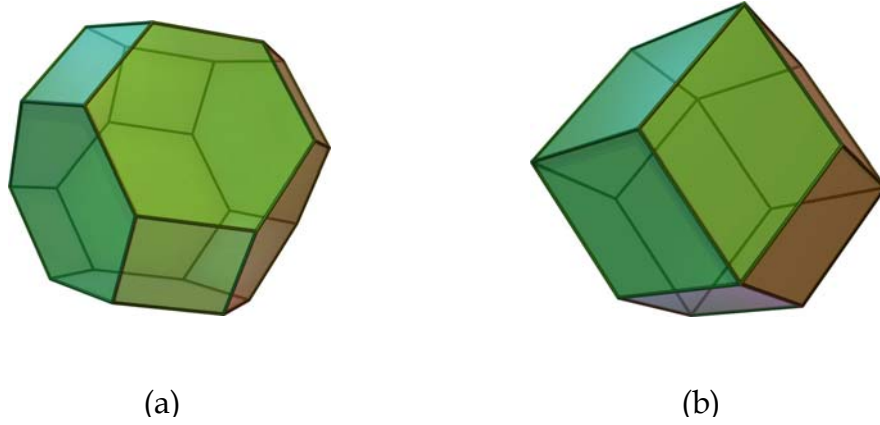


Figure 2.2: Equilibrium shapes at 0 K of an (a) fcc truncated octahedron and a (b) bcc rhombic dodecahedron.

The above discussion is only valid for crystals in free space. Nanostructures are however usually grown on supports. Equation (2.4) must therefore be modified to describe the growth of supported crystals, which was done in [37]. Accordingly, the equilibrium shape of a supported crystal is given by

$$\frac{\Delta h}{h_i} = \frac{E_{adh}}{\gamma_i} , \quad (2.12)$$

with  $\Delta h$  the amount by which the crystal's shape is truncated,  $h_i$  and  $\gamma_i$  the central distance to the facet parallel to the interface and the corresponding surface free energy and  $E_{adh}$  the work of adhesion (see Figure 2.3).

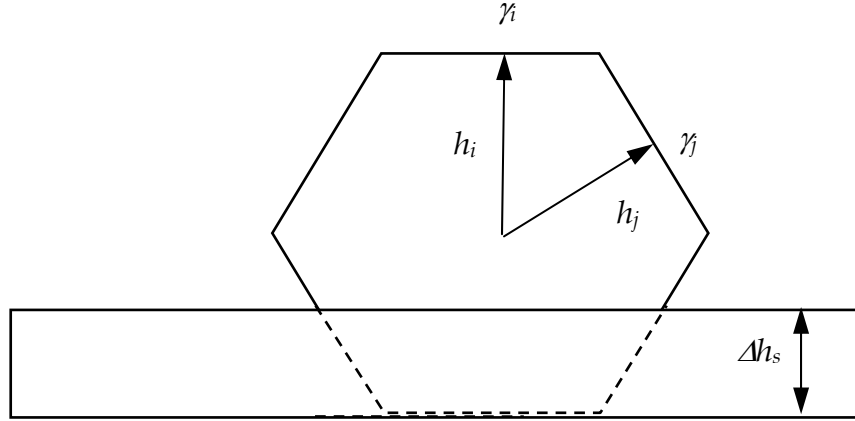


Figure 2.3: Schematic representation of the equilibrium shape of a supported polyhedron crystal. The shape of the free crystal is truncated at the interface by an amount  $\Delta h_s$  which is proportional to the adhesion energy. The  $h$ 's represent the distance from the centre of the crystal to the different facets. The surface free energies of the substrate, deposited film and interface are given by  $\gamma_s$ ,  $\gamma_d$  and  $\gamma_{\text{int}}$  respectively.

Equation (2.12) assumes that the structure of the crystal and the support are identical [38] or that homoepitaxy occurs. Epitaxy refers to the growth of a crystalline layer on (epi) a crystalline substrate, with the substrate orientation imposing an order (axis) on the deposited layer's orientation [39]. Sometimes there is a misfit between the lattice of the crystal and the support (heteroepitaxy) which can be quantified by

$$m = \frac{a_s - a_d}{a_s}, \quad (2.13)$$

with  $a_s$  and  $a_d$  the lattice parameters of the substrate and the deposited layer respectively (see Figure 2.4 for the three different types of epitaxial growth).

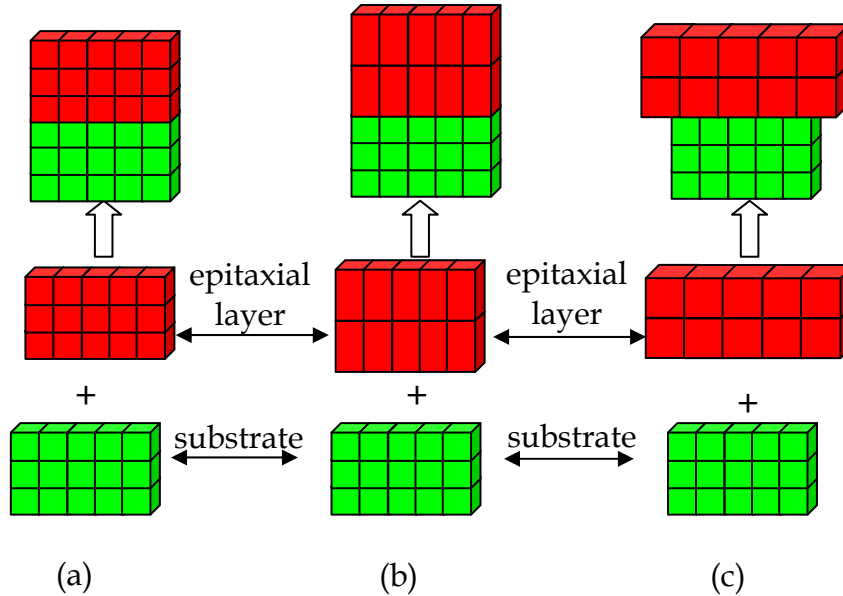


Figure 2.4: Schematic illustration of (a) lattice-matched, (b) strained and (c) relaxed heteroepitaxial structures.

It is useful to combine the Wulff-Kaischew theorem (equation (2.12)) with Young's equation for mechanical equilibrium, since this relationship provides a means of using the adhesion energy to determine whether or not a supported crystal will wet a surface. Young's equation for mechanical equilibrium is given by [35]

$$\gamma_s = \gamma_d \cos \theta + \gamma_{\text{int}}, \quad (2.14)$$

with  $\gamma_s$  the surface free energy of the substrate,  $\gamma_d$  the surface free energy of the deposited film (liquid),  $\gamma_{\text{int}}$  the interface energy between the film and the substrate and  $\theta$  the contact angle (see Figure 2.5).

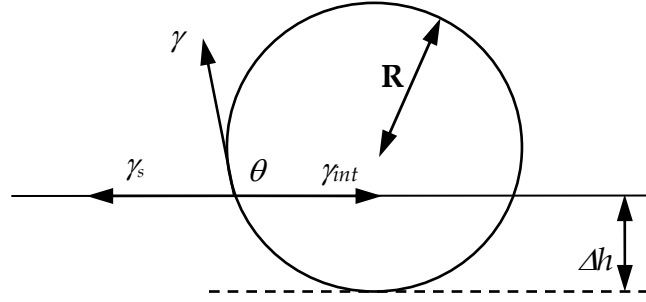


Figure 2.5: Schematic illustration of a droplet in equilibrium on a surface. The radius of the droplet is  $R$  and the droplet is truncated by an amount  $\Delta h$ . The surface free energies of the substrate, deposited film and interface are given by  $\gamma_s$ ,  $\gamma_d$  and  $\gamma_{int}$  respectively.

The adhesion energy is related to the surface interfacial energy through [40]

$$E_{adh} = \gamma_s + \gamma_d - \gamma_{int} . \quad (2.15)$$

By combining equations (2.14) and (2.15) it follows that

$$E_{adh} = \gamma_d (1 + \cos \theta) . \quad (2.16)$$

Complete wetting of the surface will thus occur at a contact angle of  $0^\circ$  and an adhesion energy of  $2\gamma_d$ . This corresponds to the well-known Frank-van-der-Merwe growth mode [41] in which interactions between the substrate and deposit atoms greatly exceed those between the deposit atoms. Each layer is therefore completely filled before growth of the next layer commences (Figure 2.6 (a)). Conversely, at a contact angle of  $180^\circ$ , the crystal morphology will be spherical with adhesion energy of zero. Non-wetting or Volmer-Weber

growth thus ensues [41]. Characteristic of Volmer-Weber growth is that the interactions between the deposit atoms are stronger than those between the deposit and substrate atoms. Three-dimensional islands therefore nucleate and grow directly on the substrate surface (Figure 2.6 (b)). Lastly, at intermediate contact angles, a combination of wetting and non-wetting behaviour can be observed and this is termed the Stranski-Krastanov mode [41]. In the Stranski-Krastanov mode, growth is initially two-dimensional but eventually proceeds with the growth of three-dimensional crystals - with their natural lattice constant - on top of the two-dimensional layers (Figure 2.6 (c)).

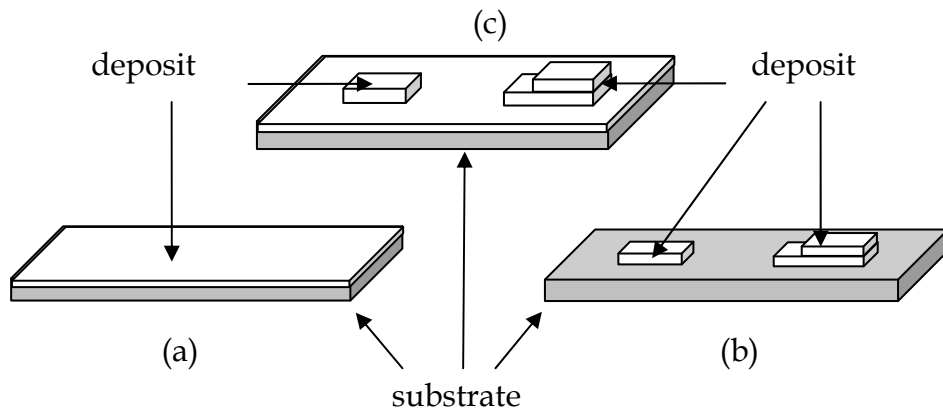


Figure 2.6: Schematic illustration of the three equilibrium growth modes. (a) Frank-van-der-Merwe growth (b) Volmer-Weber growth and (c) Stranski-Krastanov growth.

The definition of the contact angle as in equation (2.14) can only be used in descriptions of isotropic media like a liquid droplet. In supported crystals, the contact angle will be the angle between the substrate and the bottom side facets (see Figure 2.3). It is therefore defined by crystallography and not thermodynamic equilibrium. Equation (2.16) is thus insufficient to determine the equilibrium growth modes for non-isotropic media. Instead, the growth modes can be

determined by plotting the relationship between the lattice mismatch and the surface energy ratio  $W$  given by

$$W = \frac{\gamma_s - \gamma_d}{\gamma_s}, \quad (2.17)$$

in a so-called phase diagram (Figure 2.7). If  $W > 0$ , layer-by-layer growth will occur at negligible lattice mismatch. Volmer-Weber growth will dominate for  $W < 0$  whereas Stranski-Krastanov growth lies between, and competes with, layer-by-layer and Volmer-Weber growth.

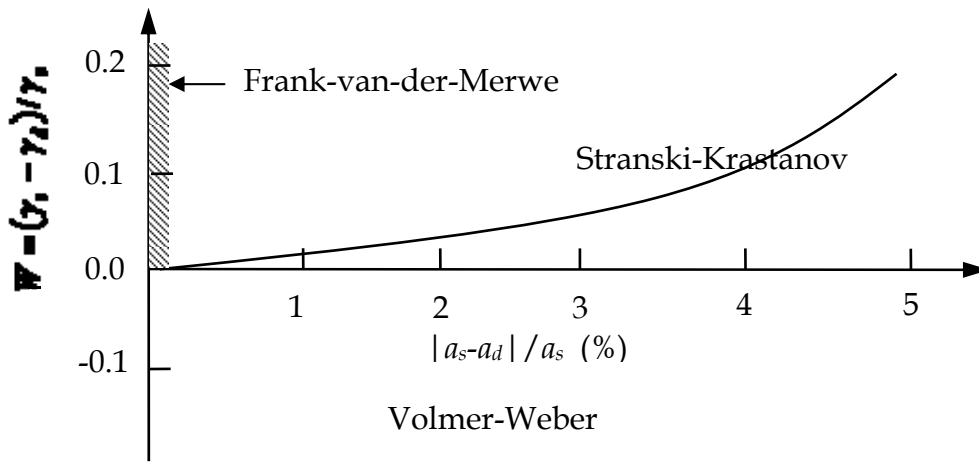


Figure 2.7: The three equilibrium growth modes as a function of mismatch and surface energy ratio. The surface free energies of the substrate, deposited film and interface are given by  $\gamma_s$ ,  $\gamma_d$  and  $\gamma_{\text{int}}$  respectively. The lattice constants of the deposited film and the substrate are given by  $a_d$  and  $a_s$  respectively.



### 2.1.1 Kinetic considerations

In the preceding sections, crystal morphology and growth were discussed at thermodynamic equilibrium. However, in practice, crystal growth rarely occurs at equilibrium. This is because the super saturation,  $S$ , which is the ratio of the pressure around the growing crystal and the equilibrium pressure at the same temperature, is typically larger than one. Generally speaking, the morphology of a crystal during growth will be determined by the growth rate of different facets. Three different types of facets can be distinguished namely (see Figure 2.8)

- Flat or  $F$ -facet which are parallel to at least two dense atomic rows;
- Stepped or  $S$ -facet which are parallel to at least one dense atomic row;
- Kinked or  $K$ -facet which are not parallel to any dense atomic rows.

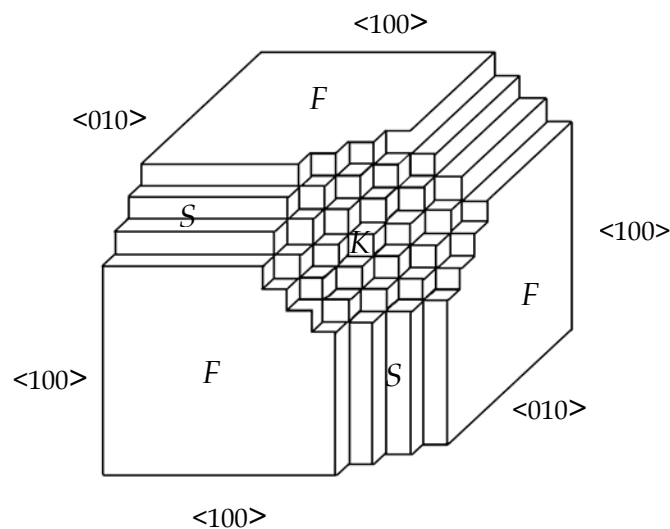


Figure 2.8: Schematic illustration of the various kinds of facets ( $S$ ,  $F$  and  $K$ ) on a growing crystal.

The *F*-facets are mostly atomically flat. Growth on these facets are thus only possible if (i) the super saturation is large enough or if (ii) the growth temperature exceeds the roughening temperature. The roughening temperature indicates the threshold above which surface roughening of an *F*-facet occurs. The *S* and *K* facets, on the other hand, are atomically rough, as shown in Figure 2.8, and grow spontaneously. The *F*-facets clearly grow much slower than the *K* - and *S*-facets and consequently growth morphologies will usually be limited by *F*-facets. The existence of faceted (or anisotropic) growth morphologies can primarily be attributed to the anisotropy in the flow of material to the different facets. Several factors can contribute to this source of anisotropy and include the following:

- *Deposition flux and surface diffusion:* In the case of growth from a vapour, if the main flux to the crystal facets comes from surface diffusion, the growth morphologies will be anisotropic if the surface diffusion is anisotropic.
- *Presence of defects:* Defects also lead to the growth of anisotropic crystals. As an example, acircular forms occur due to the presence of screw dislocations that increases the growth rate in one direction [42, 43];
- *Presence of impurities:* Impurities can drastically influence the growth shape of a crystal. Impurity ions absorb preferentially on  $\langle 111 \rangle$  faces and reduce the growth rate in this direction [44];
- *Twinning:* Twinning generates reentrant corners that are repeatable growth sites. Twinned crystals are elongated in one direction or flat [45]. Successive twinning in a  $\langle 111 \rangle$  direction gives rise to platelet triangular fcc nanocrystals [46];

- *Coalescence*: If two growing crystals touch one another, they will produce an anisotropic form that will persist unless the temperature is elevated so as to increase surface diffusion to the extent that matter is redistributed between the different facets [45].

## 2.2 Validity of the Wulff-Kaischew theorem for nanostructures

The Wulff-Kaischew theorem assumes crystals of macroscopic dimensions. In this study, however, the emphasis is on nanostructures making the use of this theorem questionable. In order to assess their validity for studying nanostructures, the various changes induced when scaling down from a macro- to nanoscale should be considered. These include:

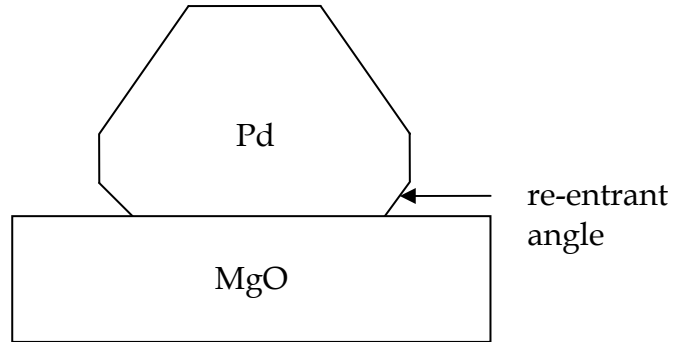
- An increase in the surface energy and stress;
- An increased stability of different structures, such as the isocahedron;
- A more prominent relationship between the edge atoms of the different facets. For example, consider a Wulff shape limited by (111) and (100) facets with  $n$  and  $m$  respectively the number of atoms along the edges of these facets. The anisotropy of the surface energy is given by [47]

$$\left( \frac{\gamma_{(100)}}{\gamma_{(111)}} \right) = \sqrt{3} \left( \frac{n+m}{n+2m} \right). \quad (2.18)$$

In a macroscopic crystal,  $n = m$ , and the anisotropy factor is  $\left( \frac{2}{\sqrt{3}} \right) \approx 1.15$ .

In a nanometre sized structure, the (100) facets disappears which means

that  $m \rightarrow 0$ ; consequently  $\left(\frac{n}{m}\right) \rightarrow \infty$ . In these conditions the anisotropy factor approaches  $\sqrt{3}$ . Using an anisotropic factor of  $\sqrt{3}$  and applying the Wulff construction [35], an octahedral shape is produced. This is indeed the morphology for a nanostructure in which (100) facets are absent. An experimental validation of the Wulff-Kaischew theorem for nanostructures is supplied in [48] in which case the adhesion energy of Pd nanostructures on MgO were determined using the Wulff-Kaischew theorem. The Pd nanostructures were grown at a high temperature to obtain the equilibrium shape of a truncated octahedron with re-entrant angles at the interface (see Figure 2.9). The measured value of 0.91 N/m is in good agreement with the 0.85 N/m obtained with molecular dynamics [49].



*Figure 2.9: Equilibrium shape of Pd nanostructure supported on a MgO (100) surface. The size of the structure is larger than 10 nm.*

## 2.3 Nanostructure Growth

Studying the equilibrium morphology and size of nanostructures and crystals is an essential step to understand how these are influenced by various factors. Examples mentioned in this chapter include surface diffusion, defects, impurities, interaction between the substrate and supported structure, rate of material flow to different facets and so forth. Naturally, precise control of these factors can ultimately satisfy the need to tailor the morphology and size of nanostructures. However, of critical importance is to first determine the processes that contribute to the growth of these equilibrium or albeit non-equilibrium morphologies. Since this study is focused on vapour deposition, the relevant processes that lead to nanostructure formation in this situation can therefore be mapped out as follows (see Figure 2.10):

- a. deposition of atoms (now termed adatoms) either on the surface or existing islands;
- b. diffusion of an adatom across a terrace;
- c. diffusion of an adatom along an island edge;
- d. dissociation of an adatom from an island (step edge);
- e. diffusion of an adatom down from an upper to a lower terrace and vice versa (interlayer diffusion);
- f. desorption from a terrace and
- g. capturing of an adatom by an existing island or step edge.

Each of these processes has to overcome a certain energy barrier before they can occur and as such they have different time scales (see Figure 2.11). The difference

in the time scales can be problematic when simulation studies for nanostructure and surface growth is conducted. This is because atomic vibration periods are of the order of  $10^{-13}$  s, whereas the formation of more complicated structures such as quantum dots, nanostructures or the deposition of an entire layer can take as much as seconds. Between these two limits there is a huge interval of 13 orders of magnitude to describe which can be very cumbersome from a computational point of view. Recently, however, there has been an upsurge in the so-called multiscale simulation methods that are specifically aimed at addressing this “time-scale” problem. An overview of these methods and, specifically, how it is relevant to this study, is given in the next chapter.

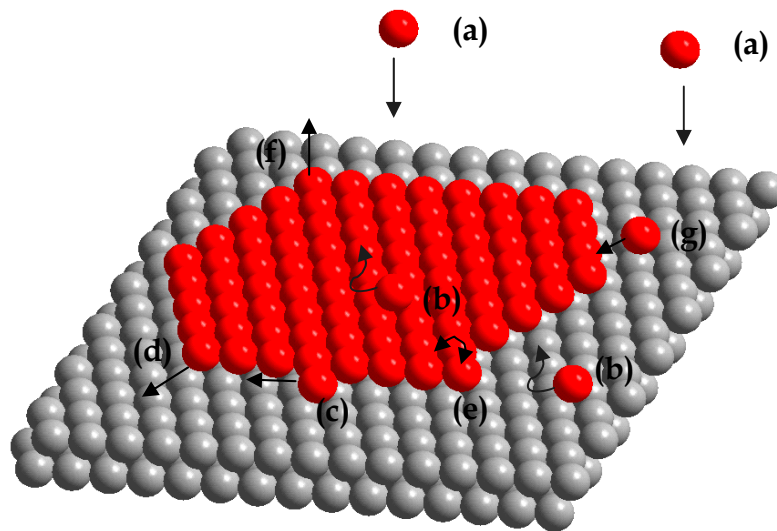


Figure 2.10: The different atomistic processes for adatoms on a surface: (a) deposition, (b) diffusion on terraces, (c) diffusion along a step edge, (d) dissociation from a step edge, (e) interlayer diffusion (f) desorption and (g) capture by a step edge.

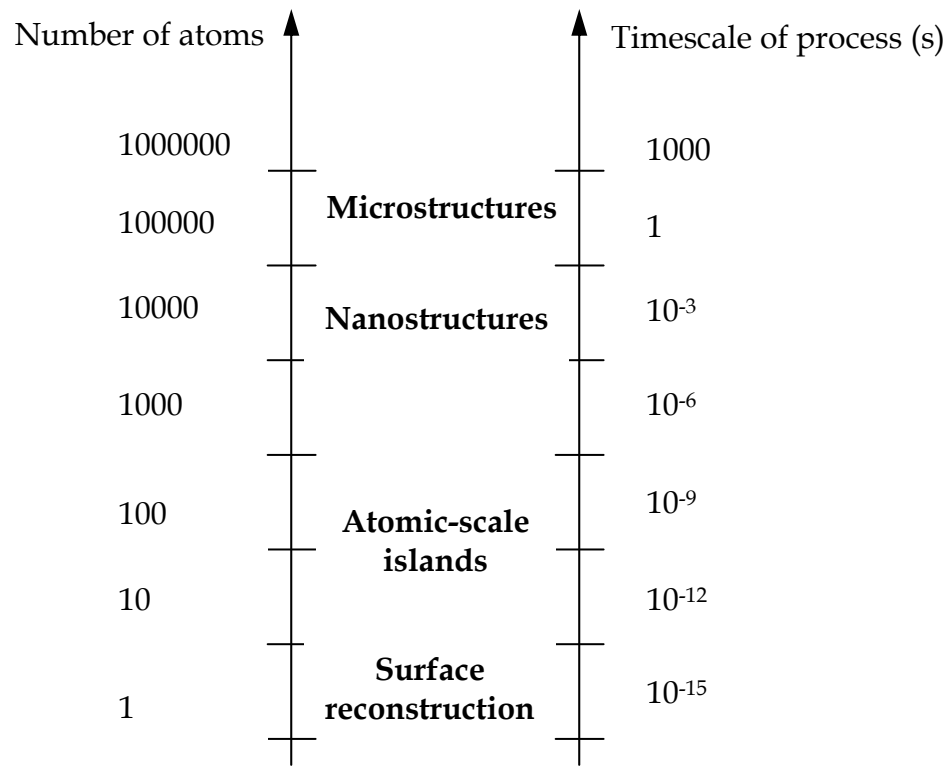


Figure 2.11: Representation of the timescale over which processes occur during epitaxial growth.

## CHAPTER 3

### Overview of Simulation Methods

#### 3.1 Length scales in growth

The growth of materials, nanostructured and otherwise, is to a large extent governed by the interactions between their atomic constituents, and, perhaps, to a lesser extent by the collective environment in which they reside in. These interactions occur on a timescale of nanoseconds to femtoseconds and ultimately influence the behaviour of the material at four characteristic length scales encountered during growth. These length scales are robustly defined as:

- The electronic and atomic scale ( $\sim 10^{-9}$  m) in which the electrons' quantum mechanical state dictate the interactions amongst the atoms;



- The microscopic scale ( $\sim 10^{-6}$  m) where the interaction between atoms are described by interatomic potentials. The potentials encapsulate the effects of bonding between the atoms, as mediated by the electrons;
- The mesoscopic scale ( $\sim 10^{-4}$  m) where lattice defects such as dislocations, grain boundaries and other microstructural elements occur. The interactions between these various entities are usually derived from phenomenological theories that encompass the effects of the interactions between their atoms;
- The macroscopic scale ( $\sim 10^{-2}$  m) where a constitutive law determines the behaviour of the physical system, viewed as a continuous medium. On this scale, continuum fields such as density, velocity, temperature, displacement and stress dominate. The constitutive laws are mostly formulated in such a way so as to capture the effects on materials properties from lattice defects, grain boundaries and microstructural elements.

Phenomena at each length scale typically have a corresponding timescale that, in correspondence to the aforementioned four length scales, ranges from femtoseconds to picoseconds, to nanoseconds, to milliseconds and beyond. At each of these length and time scales, well-established and efficient computational approaches have been developed to address the relevant phenomena (see Figure 3.1). For example, to treat electrons explicitly and accurately at the atomic scale, methods like Hartree-Fock (HF) [53], quantum Monte Carlo [52], Green's function methods [52], density functional theory (DFT) [53] in the local density approximation (LDA) [53] and generalized gradient approximation (GGA) [53] can be employed. Methods performed at the atomic scale are known to be

computationally very expensive and are therefore restricted to the study of systems containing only tens to hundreds of atoms, depending on the approximations made. Recent progress has however been made in the development of linear scaling electronic structure methods, which has enabled DFT-based calculations to deal with systems consisting of thousands of atoms [54-56]. For the determination of microscopic properties, classical Molecular Dynamics (MD) [57, 58] simulations utilizing inter-atomic potentials, often derived from DFT calculations, can be performed. Quantum MD using the Car-Parrinello scheme can also be used for the determination of microscopic properties.

Classical MD is not as accurate as DFT calculations, but is able to provide insight into atomic processes involving considerably larger systems, up to  $10^9$  atoms. The time scales probed with classical MD are however quite limited, at most of the order of picoseconds and thus not suitable for simulating growth. At the mesoscopic scale, the atomic degrees of freedom are not explicitly treated, and only larger-scale entities are modelled, usually probabilistically, enabling time scales in the range of seconds to be reached. Lastly, on the macroscopic scale, finite element methods such as continuum theories examine the large-scale properties of materials that are considered to be an elastic continuum over time scales of minutes.

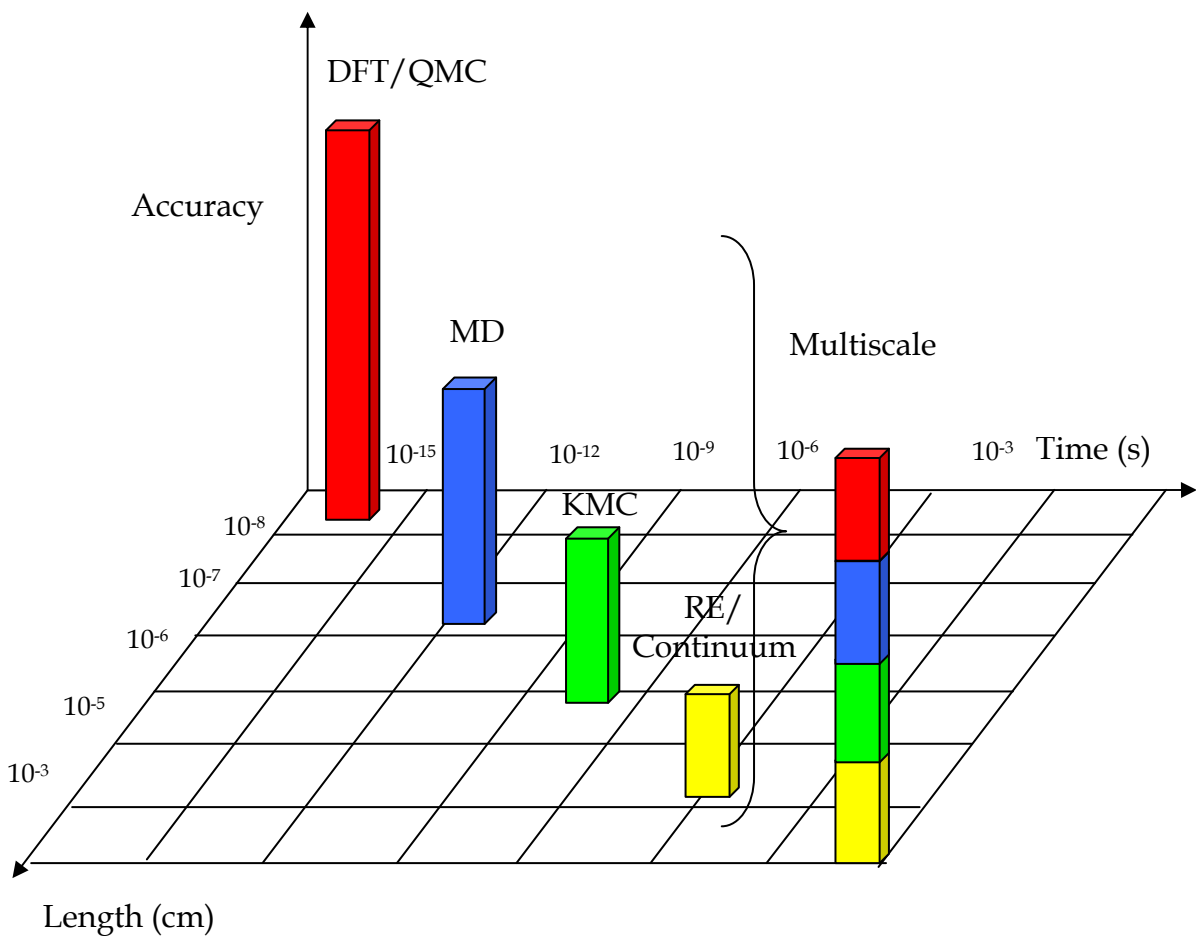


Figure 3.1: A schematic illustration of the spatial and temporal scales achievable by various simulation approaches: DFT/QMC (Density Functional Theory/Quantum Monte Carlo); classical MD (Molecular Dynamics); KMC (Kinetic Monte Carlo); RE (Rate Equation Analysis), and Multiscale encompassing all the preceding time scales.

The challenge in modelling materials, and in particular their growth, is that phenomena occur on a scale that require a very accurate and computationally expensive description, but also on another scale for which a coarser description is satisfactory and, in fact, necessary to describe the entire system. The goal consequently becomes to develop models that combine different methods specialised at different scales, effectively distributing the computational power where it is needed most.

Two categories of multiscale simulations can currently be envisioned, namely sequential and concurrent. The sequential methodology attempts to piece together a hierarchy of computational approaches in which information is obtained from more detailed, smaller-scale models. This sequential modelling approach has proved to be effective in systems where the different scales are weakly coupled. The characteristic of the systems that are suited for a sequential approach is that the large-scale variations decouple from the small-scale physics, or that the large-scale variations appear homogeneous and quasi-static from the small-scale point of view. Sequential approaches have also been referred to as serial, implicit, or message-passing methods. The second category of multiscale simulations consists of the so-called concurrent parallel or explicit approaches. These approaches attempt to link methods appropriate at each scale together in a combined model, where the different scales of the system are considered concurrently and communicate with some type of hand-shaking procedure. This approach is necessary for systems that are inherently multiscale. That is, systems whose behaviour at each scale depends strongly on what happens at the other scales.

In this work, a sequential multiscale approach is followed to study growth phenomena. Specifically, this entails coupling a mesoscopic method, namely

kinetic Monte Carlo (KMC), with atomistically determined kinetic energy barriers of relevant elemental processes that occur during growth of materials. This was done within the framework of transition state theory (TST). In the following sections, the simulation methods used in this study are discussed, with DFT receiving a bit more attention than the other microscopic methods, since this method is being used more often in multiscale methodologies. A brief overview of methods used on the macroscopic scale is also given, primarily for comparative purposes. Special emphasis is however placed on the kinetic Monte Carlo method due to it being the method employed in this study.

### 3.2 Potential energy surface

Growth of materials includes surface diffusion, which is the ultimate process through which mass transport on surfaces occurs. The probability of surface diffusion of adatoms is intimately related to the specifics of the interactions of these adatoms with the substrate. In order to elucidate a description of these interactions, suppose that the surface degrees of freedom are denoted by  $\mathbf{R}$  and those of the adatoms by  $\mathbf{R}^{ad} = (X, Y, Z)$ . The adatom/surface mechanics is then determined by the corresponding interaction potential  $U(\mathbf{R}, \mathbf{R}^{ad})$ . In most situations, the Born-Oppenheimer approximation (discussed in more detail in the next section) can be invoked for the surface; therefore one can restrict the interaction potential to the spatial degrees of freedom of the adatoms or  $U(\mathbf{R}^{ad}) = U(X, Y, Z)$ . The interaction potential perpendicular to the surface,  $U(Z)$ , has a local minimum at  $Z = Z_0$  (see Figure 3.2). An atom deposited on the surface (during growth) might get adsorbed in this minimum. On the other hand,  $U(X, Y)$ , the interaction potential in the lateral direction, displays an oscillatory

behaviour and is bound in the surface plane. This feature of  $U(X, Y)$  makes diffusive motion probable. The minima of  $U(X, Y)$  will provide stable, or metastable adsorption sites for the adatoms and is typically defined as

$$U(X, Y) = \min_Z \min_{\mathbf{R} \in \mathbf{R}} E(\mathbf{R}, \mathbf{R}^{ad}) - E^{surf} - E^{ad}, \quad (3.1)$$

where  $E(\mathbf{R}, \mathbf{R}^{ad})$  represents the total energy of the substrate/adatom system,  $E^{surf}$  and  $E^{ad}$  are, respectively, the energies of the surface and the adatom. Minimization is carried out with respect to the height ( $Z$ ) of the adatom and a given subset  $\mathbf{R}'$  or eventually all coordinates  $\mathbf{R}$  of the substrate atoms. This is done via a constrained atomic relaxation to yield equation (3.1), referred to as the potential energy surface or PES in short. This atomic relaxation can be accounted for by quantum mechanical methods such as DFT, of which a summary is given in the next section. For highest accuracy full surface relaxation should be allowed.

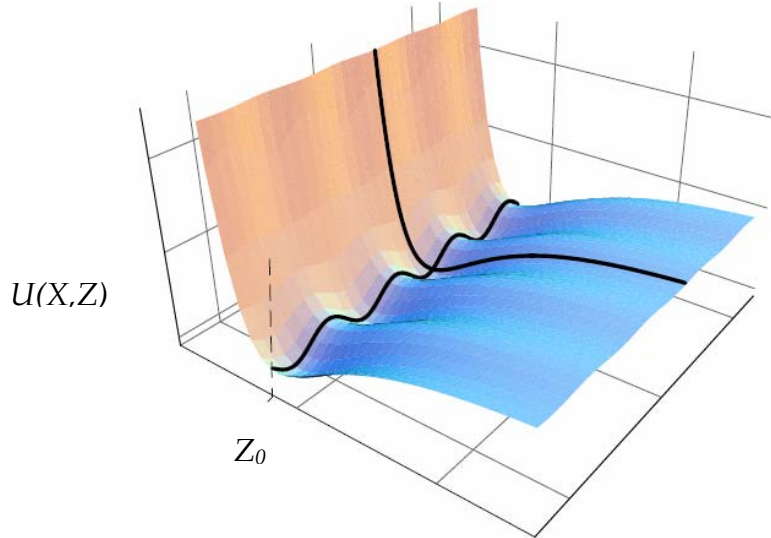


Figure 3.2: Schematic of a two dimensional slice  $U(X, Z)$  through the multi-dimensional potential energy hypersurface for the adatom surface interaction.  $Z$  is the distance to the surface and  $X$  is the adatom coordinate along a given surface direction.

### 3.3 Electronic and atomic scale

At the electronic and atomic scale, explicit electronic interactions are relevant; thus, quantum mechanical methods have to be employed. Quantum mechanical methods are aimed at solving the Schrödinger equation, which describes the interaction between the nuclei and electrons in a system [50] and is given by

$$H\Psi(\mathbf{r}, \mathbf{R}, t) = i\hbar \frac{\partial \Psi}{\partial t}, \quad (3.2)$$

with  $\mathbf{R} \equiv \{\mathbf{R}_I\}$  the positions of  $N_i$  ions,  $\mathbf{r} \equiv \{r_i\}$  the positions of  $N_e$  electrons,  $H$  the non-relativistic Hamiltonian of the system and  $\Psi(\mathbf{r}, \mathbf{R}, t)$  the many body wave function. The Hamiltonian of the system can be written as

$$H = \sum_i^{N_e} \frac{p_i^2}{2m_e} + \sum_I^{N_i} \frac{P_I^2}{2M_I} + \frac{1}{2}V_{e-e}(\mathbf{r}) + \frac{1}{2}V_{e-i}(\mathbf{r}, \mathbf{R}) + V_{i-i}(\mathbf{R}), \quad (3.3)$$

with  $p_i$  and  $P_I$  the momenta of the electrons and ion cores respectively,  $m_e$  the electron mass and  $M_I$  the mass of the ion core at position  $R_I$ . The sums over  $i$  and  $I$  run over all the electrons and ion cores respectively. The first two terms in equation (3.3) are the kinetic energies,  $T_e$  and  $T_i$ , for the electrons and nuclei and the last three terms are respectively the Coulomb contributions given by the electron-electron, electron-nucleus and nucleus-nucleus interaction. This complex many body problem can be simplified by treating the Hamiltonian perturbatively within the Born-Oppenheimer approximation [59]. This approximation makes use of the fact that  $T_i$  is usually very small due to the large masses of the nuclei; therefore, equation (3.3) can be written as

$$H \equiv T_e + \frac{1}{2}V_{e-e} + \frac{1}{2}V_{e-i} + V_{i-i} + T_i = H_0 + \lambda H_1, \quad (3.4)$$

with  $\lambda = (m/M_0)^{1/4}$  the perturbation introduced by Born and Oppenheimer and  $M_0$  some of the nuclear masses or their mean. Within the Born-Oppenheimer approximation,  $T_i$  is neglected by setting  $\lambda = 0$ . The many-electron Hamiltonian is then given by

$$H_0 \equiv T_e + \frac{1}{2}V_{e-e} + \frac{1}{2}V_{e-i}. \quad (3.5)$$

There are however situations where the Born-Oppenheimer approximation is no longer valid, for example in high-energy atom-surface collisions, or when electron-phonon coupling and electronic transitions to excited states are important. Fortunately, the properties of the system studied here can be understood on the basis of the Born-Oppenheimer approximation, thus equation 3.5 can be used with confidence. Furthermore, the latter provides a quite simple concept toward *ab initio* molecular dynamics in which the Schrödinger equation is solved for the electronic ground state and subsequently the ions are moved classically according to forces calculated from the ground state energy.

Although the many body problem has been reduced to a many electron problem using the Born-Oppenheimer approximation, the electronic Hamiltonian is approximately of the same order as the total Hamiltonian, an practically intractable for most systems. However, the degrees of freedom in the system can be dramatically reduced by reformulating the many electron problem in terms of an effective one-electron scheme. This is done within the Hartree-Fock (HF) approximation, in which case a Slater determinant of one electron orbitals,  $\varphi_i(\xi_i)$ ,



$$\psi(\xi) = \left( \frac{1}{\sqrt{N_e!}} \right) \sum_{P\{p_1, p_2, \dots\}} (-1)^P \phi_{p_1}(\xi_1) \phi_{p_2}(\xi_2) \dots \phi_{p_{N_e}}(\xi_{N_e}), \quad (3.6)$$

is employed as a trial wave function. In the above equation,  $P$  is the parity of the permutation of state indices  $\{p_1, p_2, \dots\}$  and the summation is over all permutations. The ground state of the electronic Hamiltonian is subsequently determined from a variational principle [60] using this trial wave function. Hartree-Fock theory, although it forms the foundation for the majority of computational methods in quantum chemistry, is however incapable of treating electron screening efficiently [50]. Another approach to the many body problem that is most often used in obtaining potential energy surfaces, is density functional theory (DFT). DFT reformulates the many body problem in terms of the single particle density and is founded on the following theorem [61]:

*THEOREM (Hohenberg-Kohn): Given an arbitrary number of electrons  $N_e$  moving under the influence of static, local and spin-independent external potential  $v(\mathbf{r})$  leading to the Hamiltonian*

$$H_e = T_e + V_{e-e} + v$$

*with non-degenerate ground state  $\psi_0$  and corresponding ground-state density  $n_0(\mathbf{r})$ , being a functional of  $v(\mathbf{r})$ ,*

$$n_0(\mathbf{r}) \equiv \left\langle \psi_0 \left| \sum_i^{N_e} \delta(\mathbf{r} - \mathbf{r}_i) \right| \psi_0 \right\rangle, \quad (3.7)$$

*it follows then that*

1.  $v(\mathbf{r})$  and therefore  $\psi_0$  are, within a constant, unique functionals of  $n_0(\mathbf{r})$ ;

## 2. the energy functional

$$E_v[n] = \int v(\mathbf{r})n(\mathbf{r})d\mathbf{r} + F[n], \quad (3.8)$$

where  $F[n] \equiv \langle \psi_0 | T_e + V_{ee} | \psi_0 \rangle$  is a universal functional, assumes its minimum value for the correct  $n_0(\mathbf{r})$

$$E_0 = \min_n E_v[n] \quad (3.9)$$

if the admissible functions are restricted by the condition

$$N[n] \equiv \int n(\mathbf{r})d\mathbf{r} = N_e \quad n(\mathbf{r}) \geq 0. \quad (3.10)$$

The Hohenberg-Kohn theorem does however not explain how to construct the functional  $F[n]$ . Kohn and Sham [62] proposed an equivalent scheme to treat the variational problem. Their scheme is founded on the existence of an auxiliary problem for non-interacting particles, with kinetic energy functional  $T_s[n]$  and local single particle potential  $v_s(\mathbf{r})$ , such that the ground state density of the interacting system  $n_0(\mathbf{r})$  is reproduced by that of the auxiliary problem  $n_{s,0}(\mathbf{r})$ ,  $n_0(\mathbf{r}) = n_{s,0}(\mathbf{r})$ . Then, from the ‘‘auxillary’’ one-particle Schrodinger equation, one gets a representation of  $n_0(\mathbf{r}) = \sum_i^{N_e} |\varphi_{s,i}(\mathbf{r})|^2$ . By virtue of the Hohenberg-Kohn theorem,  $\varphi_{s,i}(\mathbf{r}) = \varphi_{s,i}([n]; \mathbf{r})$  and thus  $T_s[n]$  is also a unique functional of  $n(\mathbf{r})$ . The kinetic energy term is therefore exactly represented by

$$T_s[n] = \sum_i^{N_e} \int \varphi_{s,i}(\mathbf{r}) * \left[ -(\hbar^2 / 2m) \nabla_{\mathbf{r}}^2 \right] \varphi_{s,i}(\mathbf{r}) d\mathbf{r} \quad (3.11)$$

From this central assertion, the ground-state density,  $n_0(\mathbf{r})$ , for a particular external potential  $v(\mathbf{r})$  can be written as

$$E_v[n] = \int v(\mathbf{r})n(\mathbf{r})d\mathbf{r} + \frac{1}{2} \frac{e^2}{4\pi\epsilon_0} \iint \frac{n(\mathbf{r})n(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} d\mathbf{r}d\mathbf{r}' + G[n], \quad G[n] = T_s[n] + E_{xc}[n], \quad (3.12)$$

where, by definition,  $E_{xc}[n]$  is the exchange-correlation energy functional of the interacting system with density  $n(\mathbf{r})$ . From equations (3.11) and (3.12), as well as making use of the stationary dependence of  $E_v[n]$  upon density variations such that  $\int \delta n[\mathbf{r}]d\mathbf{r} = 0$ , Kohn and Sham have derived a set of equations to determine the auxiliary potential  $v_s(\mathbf{r})$  that generates the quantity  $n_0(\mathbf{r})$ . These Kohn-Sham equations are given by

$$n_0(\mathbf{r}) = \sum_i^{N_e} |\varphi_{s,i}(\mathbf{r})|^2; \quad (3.13)$$

$$H_{KS} \varphi_{0,i}(\mathbf{r}) \equiv \left[ -(\hbar^2 / 2m) \nabla_{\mathbf{r}}^2 + v_s([n_0]; \mathbf{r}) \right] \varphi_{0,i}(\mathbf{r}) = \epsilon_i \varphi_{0,i}(\mathbf{r}); \quad (3.14)$$

$$v_s([n_0]; \mathbf{r}) = v(\mathbf{r}) + \frac{e^2}{4\pi\epsilon_0} \int \frac{n_0(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} d\mathbf{r}' + v_{xc}([n_0]; \mathbf{r}); \quad (3.15)$$

where the exchange-correlation potential is defined as the functional derivative of  $E_{xc}[n]$ ,

$$v_{xc}([n_0]; \mathbf{r}) = \left. \frac{\delta E_{xc}[n(\mathbf{r})]}{\delta n(\mathbf{r})} \right|_{n_0(\mathbf{r})}. \quad (3.16)$$

The Kohn-Sham equations (equations (3.13) to (3.15)) have to be solved self-consistently due to the density dependence of the effective Kohn-Sham potential  $v_s$ . In contrast to the HF scheme,  $v_s$  is common for all one-particle Kohn-Sham

orbitals. Thus, starting from some initial guess for the density,  $n^{[0]}(\mathbf{r})$ , the effective Kohn-Sham potential is set up according to equations (3.14) and (3.15) and the new density  $n^{[1]}(\mathbf{r})$  is generated by equation (3.13). This procedure is repeated until a certain convergence criterion is fulfilled. Once the self-consistent density is obtained, the ground-state total energy is computed from equation where  $E_0$  is given by the exact expression

$$E_0 = \sum_i^{N_e} \varepsilon_i - \frac{1}{2} \frac{e^2}{4\pi\epsilon_0} \iint \frac{n_0(\mathbf{r})n_0(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} . \quad (3.17)$$

The formulation of DFT as discussed thus far is, in principle, exact, except for the exchange-correlation potential that has to be approximated. Numerous approximations to this potential have been made over the years, with the most widely used probably the local density (LDA) and generalised gradient (GGA) approximations. Clearly, the Kohn-Sham-Hohenberg formulations provide an enormous conceptual and computational simplification of the many-electron problem.

### 3.4 Microscopic scale

Although quantum mechanical methods, such as DFT, are ideally the methods of choice due to their accuracy, their applicability to growth simulations is hindered by their high computational demand. However, depending on the system studied, one may assume that the atoms in the system are classical particles moving on a potential energy surface,  $U(\mathbf{r})$ . Following this assumption, the Schrödinger equation (equation (3.2)) may then be replaced by Newton's

equations of classical mechanics [58]

$$F_i(t) = m_i \ddot{r}_i(t), \quad (3.18)$$

with  $m_i$  the mass of the  $i$ -th atom,  $r_i$  the coordinates of the  $i$ -th atom  $F_i$  the force experienced by this atom which can be written in terms of  $U(\mathbf{r})$

$$F_i = -\nabla_i U = -\frac{\partial U(r_N)}{\partial r_i}. \quad (3.19)$$

Equations (3.18) and (3.19) underly the molecular dynamics method and are typically integrated by finite difference methods to obtain the atoms' trajectories on the potential energy surface. Most integrators are based on the Verlet algorithm and predictor-corrector methods [58], one of which is the velocity Verlet algorithm which approximates the atoms' positions,  $\mathbf{r}$ , and velocities,  $\mathbf{v}$ , using a truncated Taylor series

$$\mathbf{r}(t + \Delta t) = \mathbf{r}(t) + \mathbf{v}(t)\Delta t + (\frac{1}{2})\mathbf{a}(t)\Delta t^2. \quad (3.20)$$

The velocities are only computed at every half a time step

$$\mathbf{v}(t + \Delta t / 2) = \mathbf{v}(t) + (\frac{1}{2})\mathbf{a}(t)\Delta t, \quad (3.21)$$

which are then used to update the acceleration,  $\mathbf{a}$ , given by

$$\mathbf{a}(t + \Delta t) = -(1/m)\nabla U(\mathbf{r}(t + \Delta t)). \quad (3.22)$$

Lastly, the velocities at a full time step is calculated as follows

$$\mathbf{v}(t + \Delta t) = \mathbf{v}(t + \Delta t / 2) + (\frac{1}{2})\mathbf{a}(t + \Delta t)\Delta t. \quad (3.23)$$

A major impediment to molecular dynamics is that the integration time step must be small enough to capture the vibration modes of the system, with frequencies in the order of  $10^{13} \text{ s}^{-1}$ . This requires time steps in the femtosecond range [58]. On the other hand, transitions during growth occur infrequently, ranging from between pico - and microseconds for diffusive processes to milliseconds and minutes for aggregation phenomena. This presents the so-called “time gap” problem encountered when trying to use molecular dynamics in growth simulations as a significant number of time steps is needed to obtain macroscopic time scales. As a result of this, numerous methods have been developed to accelerate molecular dynamics. However, these are still only applicable to infrequent event systems. Examples of such methods are parallel-replica [63], hyperdynamics [64, 65] and temperature accelerated dynamics [66]. These three methods have some similarities to the mesoscopic kinetic Monte Carlo method implemented in this study.

### 3.5 Mesoscopic scale

Despite the existence of the accelerated molecular dynamics methods, other, more intuitive, approaches have been developed to address the “time-gap problem” presented above. These approaches have been developed on a mesoscopic scale and replace the deterministic equations of Molecular Dynamics with stochastic transitions for the infrequent events in the system. The transitions are employed via transition state theory (TST) [67]. The convenience of these methods is that they can be directly used for this study, since, on this scale, growth can be considered as a stochastic (random) process.

### 3.5.1 Stochastic processes

A stochastic process is defined in terms of a stochastic variable, say  $X$ , which, is specified by [68]:

- The set of possible values (called “range”, “set of states”, “sample state” or “phase space”);
- The probability distribution over this set. This set can be discrete (e.g. the number of molecules of a component in a reacting mixture), continuous (e.g. the Brownian motion of a particle) or multidimensional. In the latter case, the stochastic variable is just a vector (e.g. the three velocity components of a Brownian particle).

An infinite number of stochastic variables,  $Y$ , can be derived from the stochastic variable,  $X$ . These stochastic variables are defined as functions of  $X$  by some mapping  $f$ . They can be any kind of mathematical object, including functions of an additional variable  $t$ , written as

$$Y(t) = f(X, t) = X(t_1), X(t_2), \dots, X(t_N). \quad (3.24)$$

Such a quantity,  $Y(t)$ , is called a random function or, since in most cases  $t$  stands for time, a stochastic process. Examples of stochastic processes include Brownian motion, random walks, Poisson and Markov processes. Figure 3.3 gives a more intuitive interpretation of a stochastic process.

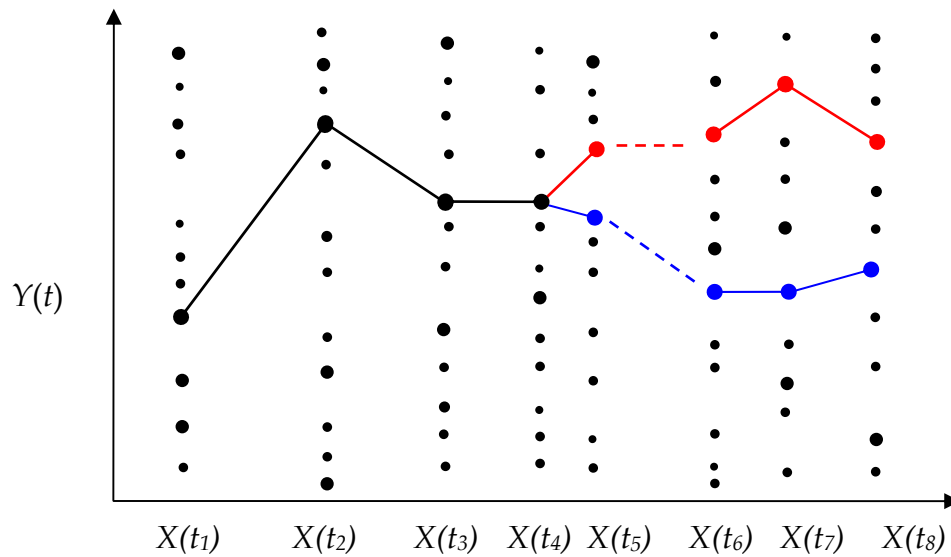


Figure 3.3: A schematic interpretation of a stochastic process,  $Y$ , as a function of stochastic variables  $\{X(t_i)\}$ . At successive times, the most probable values of  $Y$  have been drawn as heavy dots. The most probable trajectory can be selected from such a picture. Two or more trajectories can occur with equal probability.

### 3.5.2 The conditional probability

In a system that is intrinsically stochastic, one can derive a Master Equation that describes the probability of the system to occupy each one of a set of discrete states,  $W$ . In order to obtain such a Master Equation, one has to first define the conditional probability. In general, the conditional probability  $P_{lk}$  is given as [69]



$$P_{l|k}(y_{k+1}, t_{k+1}; \mathbf{K}; y_{k+l}, t_{k+l} | y_1, t_1; \mathbf{K}; y_k, t_k) = \frac{P_{k+l}(y_1, t_1; \mathbf{K}; y_k, t_k; y_{k+1}, t_{k+1}; \mathbf{K}; y_{k+l}, t_{k+l})}{P_k(y_1, t_1; \mathbf{K}; y_k, t_k)} \quad (3.25)$$

In the above equation,  $P_k(y_1, t_1; \mathbf{K}; y_k, t_k)$  is the probability that the stochastic variable,  $Y$ , assumes the value  $y_1$  at  $t_1$ ,  $y_2$  at  $t_2$  and so on.

### 3.5.3 The Markov process

The stochastic processes can be divided into various subclasses, of which the Markov processes are part. A Markov process can be defined as a stochastic process with the property that, for any set of  $N$  successive times (i.e.,  $t_1 < t_2 < \dots < t_N$ ), it follows that [69]

$$P_{l|N-1}(y_N, t_N | y_1, t_1; \mathbf{K}; y_{N-1}, t_{N-1}) = P_{l|1}(y_N, t_N | y_{N-1}, t_{N-1}) . \quad (3.26)$$

Thus, the conditional probability at  $t_N$ , given the value  $y_{N-1}$  at  $t_{N-1}$ , is uniquely determined and not affected by knowledge of the values at prior times. A Markov process is therefore only dependent on  $P_1(y_1, t_1)$  and  $P_{1|1}(y_2, t_2 | y_1, t_1)$ ; and subsequently the whole hierarchy can be reconstructed from them. For instance, if  $t_1 < t_2 < t_3$ , it can be written [69]

$$\begin{aligned} P_3(y_1, t_1; y_2, t_2; y_3, t_3) &= P_2(y_1, t_1; y_2, t_2) P_{1|2}(y_3, t_3 | y_1, t_1; y_2, t_2) \\ &= P_1(y_1, t_1) P_{1|1}(y_2, t_2 | y_1, t_1) P_{1|1}(y_3, t_3 | y_2, t_2) . \end{aligned} \quad (3.27)$$

The above algorithm can be continued successively to find all  $P_N$ . This property makes Markov processes manageable, which is the reason why they are so useful

in many applications [70, 71]. The Markov property states that, to make predictions of the behaviour of a system in the future, it suffices to consider only the present state of the system and not the past history. The details of the Markov processes are much more complex than described here and can be found in [69-71].

### 3.5.4 The Master Equation

From the above definitions, the Master Equation can be derived. The details of this derivation is however not trivial and are therefore briefly outlined in Appendix A. Instead, only the most important result is presented which is that a Markov process, as outlined above, corresponds to a particular Master Equation, given by [72]

$$\frac{\partial p_n(t)}{\partial t} = \sum_n \{ W_{m'n'} p_{n'}(t) - W_{n'n} p_n(t) \} \quad (3.28)$$

Thus, the Master Equation can be interpreted as a gain/loss equation for the probability of each state  $n$ . The first term,  $W_{m'n'}$ , is the gain due to transitions from other states  $n'$ , and the second term,  $W_{n'n}$ , is the loss due to transitions into other states  $n'$ . Two very important criteria have to be satisfied when using the Master Equation, namely steady state and detailed balance [73]. Steady state occurs when the time derivative of the Master Equation is zero. This implies that the sum of all the transitions into a particular state  $n$  equals the sum of all the transitions out of a particular state  $n'$ . Thus the steady state condition can be

written as

$$\sum_{n'} W_{nn'} p_{n'}(t) = \sum_{n'} W_{n'n} p_n(t) . \quad (3.29)$$

Detailed-balance, on the other hand, asserts that for each pair,  $n$  and  $n'$ , the transitions must balance, or,

$$W_{nn'} p_{n'}(t) = W_{n'n} p_n(t) . \quad (3.30)$$

It is important to impose detailed balanced to ensure that the Monte Carlo transition probabilities are consistent with the Boltzmann distribution [74]

$$p(t) = \frac{e^{-\Delta E / k_B T}}{Z} , \quad (3.31)$$

with  $Z$  the partition function,  $k_B$  Boltzmann's constant,  $T$  the temperature of the system and  $\Delta E$  the energy barrier between states  $n$  and  $n'$ . A clear distinction has to be made between detailed balance and steady state, as explained in Figure 3.4. In Figure 3.4 (a), the anticlockwise transition proceeds at twice the rate of the clockwise transition, therefore,  $S$  (steady state) holds, but  $D$  (detailed balance) does not. In Figure 3.4 (b), both transitions occur at the same rate, thus  $D$  is satisfied and consequently  $S$  as well. Detailed balance is therefore a sufficient, but not necessary condition for thermodynamic equilibrium.

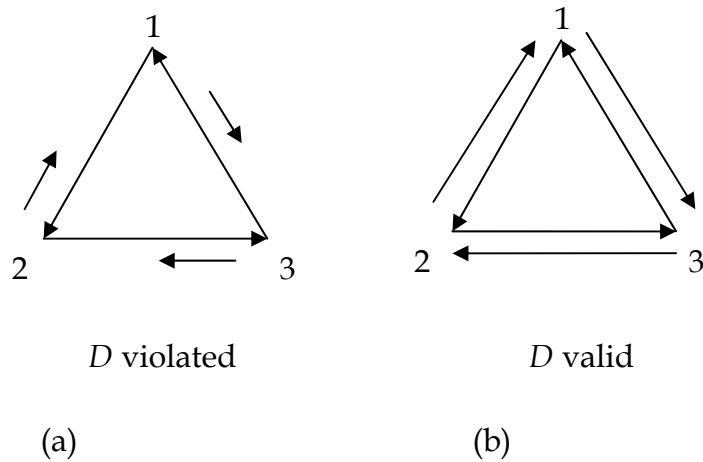


Figure 3.4 Schematic illustration of the difference between the steady state condition property and detailed balance. The lengths of the arrows are proportional to the transition rate). In (a) steady state is satisfied but detailed balance not, whereas in (b) both steady state and detailed balance are satisfied.

### 3.5.5 Solution of the Master Equation

The Master Equation can only be solved analytically for the simplest systems [75]. This stems from the fact that all the states in the system must be represented in the Master Equation description. Usually this yields a large number of states that are computationally difficult to cope with. Therefore, the Master Equation is frequently solved stochastically with algorithms such as Metropolis Monte Carlo. Metropolis Monte Carlo is widely used since, within the context of detailed balance, it follows a Markov process to evolve the system towards equilibrium, regardless of the pathway. It also makes all transitions of the system to states of lower or equal energy with probability of unity, regardless

of the energy barrier,  $\Delta E$ , to access this state or the process required. In other words, the transition probability per unit time (or transition rate),  $W$ , between states  $n$  and  $n'$  is chosen as

$$W = \begin{cases} e^{-\Delta E/k_B T} & \text{for } \Delta E > 0 \\ 1 & \text{for } \Delta E \leq 0 \end{cases} \quad (3.32)$$

Hence, Monte Carlo moves based on the Metropolis algorithm cannot be interpreted dynamically as a process that simulates random motion in time. The solution given by equation (3.32) is one of many solutions to the Master Equation that can generate the equilibrium configuration. However, the correct transition rates might not obey detailed balance and might correspond to a non-equilibrium process. In order for a method to describe the correct evolution in time as well as the proper equilibrium state, it has to consist of a sequence of microscopic processes underlying the various transitions in the system [76]. These processes can be grouped together by certain distinct events,  $e_i$ :

$$\mathbf{E} \equiv \{e_1, e_2, \dots, e_n\}, \quad (3.33)$$

which can be characterized by average transition rates

$$\mathbf{R} \equiv \{r_1, r_2, \dots, r_n\}. \quad (3.34)$$

The transition rate, as mentioned before, is just the probability per unit time of the system undergoing a transition from one state to the next. From equations (3.33) and (3.34), it can now be assumed that any particular transition possible at time  $t$ , can be possible at any time  $t + \Delta t$  with a uniform probability which is based on its rate and independent of any previous events. This is, by definition, a Poisson process. The Poisson process is part of a family of Markov

processes that are called one-step processes. One-step processes are continuous in time, have a range of integers  $n$ , and can only jump between adjacent states. Figure 3.5 helps to visualize these processes.

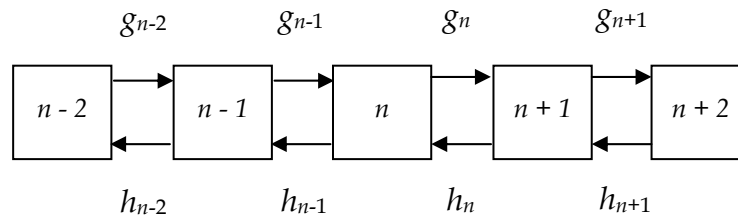


Figure 3.5: Schematic illustration of a one-step process. Only jumps between adjacent states (labelled  $n$ ) are allowed. The probability per unit time for jumps in the forward and reverse direction is denoted by  $g_n$  and  $h_n$  respectively.

The Master Equation for such processes is given by

$$\dot{P}_n = h_{n+1}P_{n+1} + g_{n-1}P_{n-1} - (h_n + g_n)P_n, \quad (3.35)$$

where  $h_n$  is the probability per unit time for a jump from state  $n$  to state  $n - 1$  and  $g_n$  is the probability per unit time for a jump from  $n$  to  $n + 1$ . One step processes occur at:

- Generation and recombination of charge carriers;
- single-electron tunnelling;
- surface growth of atoms.

Based on the coefficients  $h_n$  and  $g_n$ , one-step processes can be categorized as one of the following:

- linear for coefficients that are linear functions of  $n$ ;
- nonlinear for coefficients that are nonlinear functions of  $n$  and
- random walks for coefficients that are constant.

An example of a random walk is the Poisson process which determines the probability of  $n$  events occurring at time  $t > 0$ . This event could for example be the tunnelling of electrons through a single barrier. The Poisson process is defined by setting

$$h_n = 0, \quad g_n = q, \quad p_n(0) = \delta_{n,0}, \quad (3.36)$$

with  $q$  a constant and the probability of processes in the reverse direction is zero ( $r_n = 0$ ). The Kronecker delta indicates that the probability for no events to occur after time zero equals one, and the probability of more than one event to occur after time zero equals zero. Figure 3.6 is a schematic illustration of a Poisson process.

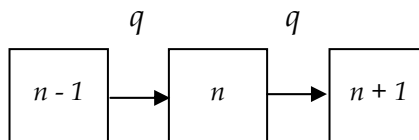


Figure 3.6: Schematic illustration of the Poisson process.

The Master Equation for the Poisson process has the form:

$$\dot{p}_n = q(p_{n-1} - p_n) , \quad (3.37)$$

which has the following solution:

$$p(n) = \frac{(qt)^n}{n!} e^{-qt} . \quad (3.38)$$

In the context of transition rates (see next section),  $q$  can be set to  $\mathbf{R}$ , thus

$$p(n) = \frac{(Rt)^n}{n!} e^{-Rt} . \quad (3.39)$$

The probability density between successive events is then given by

$$p(t) = Re^{-Rt} . \quad (3.40)$$

From this probability density, the mean time between successive events is given by

$$\langle t \rangle = \frac{1}{R} . \quad (3.41)$$

Equation (3.41) only gives the mean time between successive events; however, it may be of interest to know the real time for an event to occur. This can be inferred from equation (3.40) by noting that the probability for an event to occur at time  $\tau$  is given by

$$T(\tau) = \int_0^{\tau} dt' Re^{-Rt'} = 1 - e^{-R\tau} . \quad (3.42)$$



This probability lies in the interval  $[0, 1]$ . The probability for an event not to occur is just  $K(\tau) = 1 - T(\tau)$ , which implies  $K(\tau) = e^{-R\tau}$ . The functional inverse of  $K(\tau)$  can then be written as  $K^{-1}(\tau) = \ln(K(\tau)) = -R\tau$ . Therefore, it follows that  $K^{-1}(\tau)$  can relate time to sampling distribution. Since  $K(\tau)$  can be any number (random) in the interval  $[0, 1]$ , the real time,  $\tau$ , between successive events is given by

$$\tau = \frac{-\ln U}{R}, \quad (3.43)$$

with  $U = e^{-R\tau}$  a random number uniformly distributed between  $[0, 1]$ . Suppose now that there are  $X$  species in the system and  $k$  events characterised by the rates  $\{r\} = \{r_1, \dots, r_k\}$ . The  $X$  species can then partition among various possible transition events as  $\{X\} = \{x_1, \dots, x_k\}$ , where  $x_i$  is the number of species capable of undergoing a transition with a rate  $r_i$  at  $X = \sum_{i=1}^k x_i$ . If a sufficiently large system is used to achieve independence of the various events, then the Monte Carlo algorithm effectively simulates the Poisson process and the passage of real time can be maintained in terms of  $\{r\}$  and  $\{X\}$  [76]. For each trial  $i$  in which the event is realised, the time should be updated with an increment  $\tau_i$  selected from an exponential distribution with  $R_i = \sum_{j=1}^k x_j r_j$ .

This above formulation is the essence of kinetic Monte Carlo (KMC). KMC has a direct relation to real time, instead of the steps of Monte Carlo. It can consequently be used to study dynamic processes, in particular those where energy barriers govern the transition between subsequent states. Inherent to the KMC method is the grouping of these processes that underly the transition rates and can include the diffusion of atoms on a potential energy surface. Thus, even

though the processes can, in most cases, be easily identified, their corresponding transition rates might not always be that trivial to determine.

To date a number of schemes have been developed to address the problem of calculating the transition rates and these differ according to the complexity of the treatment of the adatom/surface system. A method that is computationally expedient and efficient for the calculation of transition rates is transition state theory (TST). TST makes use of the fact that atoms, which reside in minima on the potential energy surface, will only occasionally acquire enough energy to overcome the energy barrier between two adjacent minima. TST therefore yields, from a canonical distribution, the rate for a system to alternate between two states separated by a potential energy barrier. TST is discussed in more depth in the next section.

### 3.5.6 Transition state theory (TST)

The underlying idea of transition state theory is outlined in Figure 3.7 in which a two-state problem is represented, consisting of the initial state  $i$ , final state  $j$  and the transition state  $x_0$  separating the two states. By assuming a canonical ensemble, it is possible to derive an expression for the rate at which the infinite heat bath pushes the atom at state  $i$  through the transition state to state  $j$ . Suppose now that an atom is located in a small region,  $\Delta x$ , around the transition state and moving towards state  $j$ . The probability of finding the atom in this region around the transition state  $x_0$  is given by [77]

$$P(\Delta x) = \frac{\exp(-\beta V(x_0))}{\int_{-\infty}^{x_0} \exp(-\beta V(x)) dx}, \quad (3.44)$$

with  $\beta = \frac{1}{k_B T}$ ,  $k_B$  Boltzmann's constant and  $T$  the temperature of the system.

The upper limit of the normalization integral in the denominator expresses the assumption that the atom resides in site  $i$  initially. Likewise, the probability density for the atom having velocity  $v$  is

$$\begin{aligned} P(v) &= \frac{\exp(-\beta m v^2 / 2)}{\int_{-\infty}^{+\infty} \exp(-\beta m v^2 / 2) dv} \\ &= \sqrt{\frac{m\beta}{2\pi}} \exp(-\beta m v^2 / 2) \end{aligned} \quad (3.45)$$

Within a short time interval  $\Delta t$ , the atom with (positive) velocity  $v$  will enter site  $j$  provided that  $\Delta x$  is smaller than  $v\Delta t$ . The total probability that the atom can jump from site  $i$  to site  $j$  can be written as

$$P_{i \rightarrow j} = \Delta t v_{i \rightarrow j}^{TST} = \int_0^{\infty} P(v) P(\Delta x = v\Delta t) dv, \quad (3.46)$$

where  $v_{A \rightarrow B}$  is the transition rate. Using equations (3.45) and (3.46) an expression for the transition rate is obtained

$$v_{i \rightarrow j}^{TST} = \frac{\exp(-\beta V(x_0))}{\sqrt{2\pi m\beta} \int_{-\infty}^{x_0} \exp(-\beta V(x)) dx}. \quad (3.47)$$

It is thus clear that the problem reduces to solving the integral in the denominator of equation (3.47). If  $V(x_0) \gg k_B T$ ,  $V(x)$  can be replaced with its expansion to the second order. The expression then becomes

$$v_{i \rightarrow j}^{TST} = v_i \exp(-\beta \delta E), \quad (3.48)$$

where  $v_i$  is the harmonic vibration frequency of  $V(x)$  at site  $i$ , and  $\delta E$  is the diffusion barrier. Equation (3.48) thus represents the transition rate associated with a process that has to overcome an energy barrier  $\Delta E$  to occur.

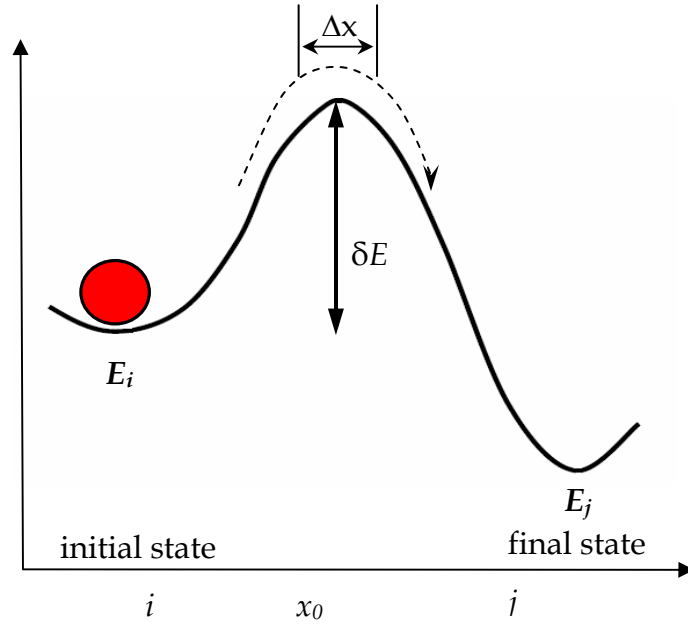


Figure 3.7: Schematic illustration of the lowest free energy path for a thermally activated jump of an adatom from  $i$  to  $j$  over the saddle point  $x_0$ .

### 3.6 Macroscopic scale

Macroscopic methods are founded on differential equations constructed from differentiable densities that replace the underlying atomic structure. As the macroscopic scale is beyond the scope of this work, two of the most popular macroscopic methods, namely rate equation analysis and continuum theories, is

discussed only briefly below. This is just to serve as reference to methods that can be used on this scale and to compare how they differ from the KMC method implemented in this study.

### 3.6.1 Rate equation analysis

Rate equation approaches are based on phenomenological identification of processes that cause adatom and island densities to change. If the density of adatoms is denoted by  $n_1(t)$  and the density of islands with size  $s > 1$  by  $n_s(t)$ , a rate equation for the density of the adatoms can be written as [78, 79]

$$\frac{dn_1}{dt} = J - 2D\sigma_1 n_1^2 - Dn_1 \sum_{s=2}^{\infty} \sigma_s n_s, \quad (3.49)$$

with  $J$  the deposition flux on the surface,  $D$  the adatom diffusion coefficient and  $\sigma_s$  the capture coefficient for an island of size  $s$ . The second and third term in equation (3.49) represent the nucleation and the attachment term. The rate equations for the density of an  $s$ -atom island  $n_s(t)$  is given by

$$\frac{dn_s}{dt} = Dn_1\sigma_{s-1}n_{s-1} - Dn_1\sigma_s n_s. \quad (3.50)$$

The first term on the right-hand side of equation (3.50) represents the rate of increase in the  $n_s(t)$  by the attachment of adatoms to  $(s-1)$ -atom islands. Similarly, the second term is the rate of decrease of  $n_s(t)$  by the attachment of adatoms to  $s$ -atom islands to form  $(s+1)$ -atom islands. Setting the capture numbers to unity and introducing the total island density  $N$ , a closed set of two equations for  $n_1$

and  $N$  can be obtained

$$\frac{dn_1}{d\theta} = 1 - 2Rn_1^2 - Rn_1N, \quad (3.51)$$

$$\frac{dN}{d\theta} = Rn_1^2, \quad (3.52)$$

where  $R = D/J$  and  $\theta = Jt$  the coverage of the surface. Numerical integration of equations (3.51) and (3.52) and setting the capture numbers equal to unity for simplicity, yields the important scaling laws of rate equation analysis at steady state

$$n_1 \sim \theta^{-1/3} (D/J)^{-2/3}; \quad (3.53)$$

$$N \sim \theta^{1/3} (D/J)^{-1/3}. \quad (3.54)$$

Rate equations are very useful in elucidating island density, however, the distribution of island sizes are not correct [78, 79]. A complete description of rate equations can be found in [78, 79].

### 3.6.2 Continuum theories

The simplest time dependent description of a growing surface is afforded by the continuum theories. In this case, the stochastic nature of the growth process, inherent to the mesoscopic methods, is neglected. Instead, on a more robust scale, every property is averaged over a small volume containing many atoms in an attempt to capture the essential mechanisms responsible for the growth morphology. A detailed description of continuum methods can be found

elsewhere, however, the first well-known continuum equation, called the

Edwards-Wilkinson equation, can be written as [80]

$$\frac{\partial h(x,t)}{\partial t} = \nu \nabla^2 h(x,t) + \eta(x,t) . \quad (3.55)$$

The term  $\nu$  represents the surface tension since  $\nu \Delta^2 h$  tends to smooth the surface, whereas  $\eta$  is defined as a noise term. Equation (3.55) corresponds to a growth model in which atoms are randomly deposited on a substrate and are allowed to diffuse over a finite distance. The above equation is linear in  $h$  and is only capable of describing the height evolution of the surface. A non-linear perturbation of the Edward-Wilkinson equation can however be introduced to account for lateral growth, which is done in the Kardar-Parisi-Zhang equation given by [80]

$$\frac{\partial h(x,t)}{\partial t} = \nu \nabla^2 h(x,t) + \lambda (\nabla h(x,t))^2 + \eta(x,t) . \quad (3.56)$$

The non-linear term  $(\Delta h)^2$  is responsible for lateral growth. Continuum models contain more information than rate equations because the surface morphology is also described locally. However, the description is on a coarsed-grained scale rather than on the atomic scale since they provide information only on the collective nature of growth processes. Therefore, their applicability is limited to length scales larger than the typical interatomic distances.

## CHAPTER 4

### Implementation of Kinetic Monte Carlo

The basic ideas of kinetic Monte Carlo can be systematically implemented as follows:

- Make justified assumptions to enhance the tractability of the KMC method;
- Identify the relevant processes in the system and employ a means of distinguishing them;
- Adopt a methodology to calculate the rates of the above processes;



- Incorporate an appropriate data structure/s which can be used to store the calculated process rates;
- Combine the aforementioned into an algorithm that can be used to simulate the system under study, in the case of this study, the vapour deposition of gold on graphite.

Each of these points is discussed in detail in the next four sections.

#### **4.1 The lattice gas assumption**

Probably the most important assumption in implementing KMC is that the system under study can be represented by a lattice gas [81]. This entails constructing a lattice which constitutes positions on the potential energy surface most likely occupied by adatoms. These “lattice gas” models neglect defects or dislocations that may develop in a growing film as well as changes in the lattice due to relaxation. Furthermore, deposition only takes place at the pre-set lattice sites and diffusion is described by hops between the various sites. A two-dimensional illustration of the lattice gas assumption is illustrated in Figure 4.1.

The lattice gas model, as illustrated in Figure 4.1, can be extended to a quasi-three-dimensional solid-on-solid (SOS) model (Figure 4.2). In this case, overhangs or voids are disallowed and atoms are deposited directly on top of one another. Essentially, this implies that the growth of the crystal/surface can be described by an integer array of variables. SOS models are therefore able to simulate large systems with relatively little complexity.

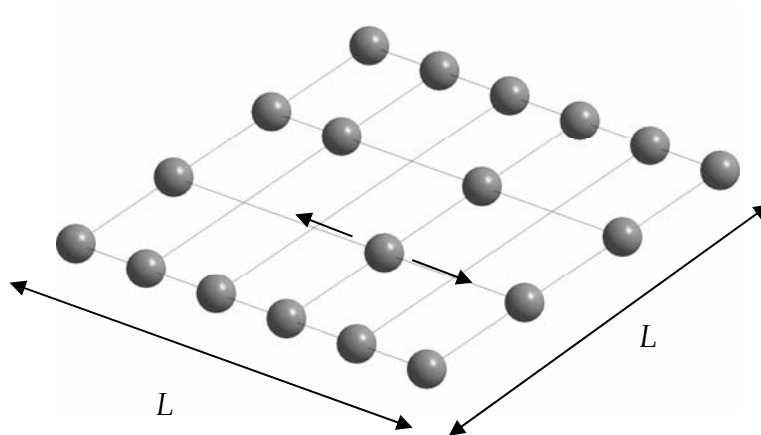


Figure 4.1: Schematic illustration of the lattice gas assumption. A two-dimensional lattice is constructed which constitutes possible adsorption sites on the substrate. Diffusion processes are represented by hops between the lattice sites. The system size is  $L \times L$ .

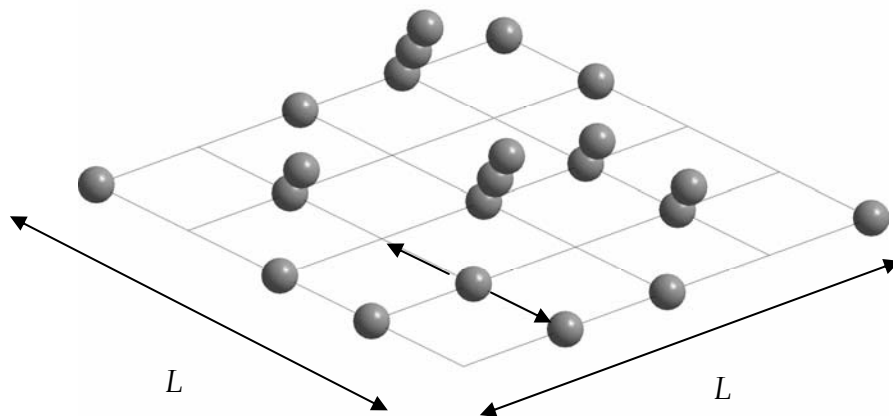


Figure 4.2: Schematic illustration of the lattice gas assumption. A two-dimensional lattice is constructed which constitutes possible adsorption sites on the substrate. Diffusion processes are represented by hops between the lattice sites. The system size is  $L \times L$ . As overhangs and bulk vacancies are neglected, the surface is fully characterized by an integer array of height variables above a square lattice substrate.

KMC does not necessarily imply the use of a lattice and can be performed using continuous particle positions. Such models are usually limited in scope since the relevant activation energy barriers must be constantly calculated during a simulation and are therefore computationally very time-consuming, especially in three dimensions. An excellent description of an off-lattice or “*on-the-fly*” KMC in one-dimension using a Lennard-Jones system can be found in [82].

#### 4.1.1 The lattice geometry

In principle, all relevant lattice structures can be implemented in a Solid-on-Solid fashion. However, imposing an inadequate geometry, e.g. using the square lattice shown in Figure 4.1 to represent an fcc (111) surface, may result in subtle difficulties. An fcc (111) surface is better represented by a hexagonal lattice; however, due to the nature this structure, atoms cannot be stacked on top of one another and SOS models can consequently not be employed. Nevertheless, the lattice gas assumption is still applicable. In addition, a heteroepitaxial system has to be treated with caution, because the structure of the substrate and deposited layer may differ significantly from one another. This would imply a large lattice mismatch which will give rise to strain that may destabilise a normally flat surface and either result in the formation of dislocations or mounds [83, 84] as a way to release the excess elastic energy. The construction of a lattice for heteroepitaxial systems is therefore not trivial. In this study (which considers a hetero-epitaxial system), the lattice was established by taking into account the geometries of the graphite and gold surfaces which are as follow:

- The graphite surface consists of hexagonally closed packed (hcp) carbon atoms, illustrated in Figure 4.3. The surface atoms are characterised as being

either  $\alpha$  or  $\beta$  bonding sites, with the  $\alpha$  site having an atom directly below it in the next layer and the  $\beta$  site not. These two chemically distinct sites have different binding energies for adsorbate materials [85]. The parameters for the unit cell is  $a = b = 2.46 \text{ \AA}$ .

- The gold surface, like graphite, has a close packed structure (fcc) with a lattice constant of  $4.08 \text{ \AA}$  [86] and interatomic distance of  $2.88 \text{ \AA}$  [86]. Generally, a gold surface will grow in the fcc (111) direction (lowest surface energy plane [87]) if no other surface structure is imposed such as the (100) and (110) directions. These three fcc structures are depicted in Figure 4.4. Additionally, the fcc (111) packing provides the closest match to the graphite surface (see Figure 4.5).

During vapour deposition, gold atoms can adsorb on (i) the  $\alpha$  or  $\beta$  sites on the graphite surface or (ii) the three-fold hollow sites created by the atoms in an existing gold island. On the graphite surface, preferential adsorption of gold on the graphite surface occurs in the direction of the  $\pi_z$  orbital (according to DFT studies [88]). Similar observations were made in ref. [89] in which the adsorption of silver atoms on graphite was studied using DFT. In both these cases it was found that the binding energy difference between the two adsorption sites on graphite ( $\alpha$  and  $\beta$ ) is relatively small ( $\sim 0.05 \text{ eV}$ ). The assumption was therefore made that adsorption of gold on graphite can be at either of these two sites. Furthermore, it was assumed that the gold atoms of the growing deposit are laid down in contiguous, linear, parallel arrays separated by  $2.88 \text{ \AA}$  such that the arrays coincide with the  $\langle 1010 \rangle$  directions of the graphite. The lattice misfit (discussed in Chapter 2) in this case will be approximately 1.4 %. Large lattice misfits ( $> 4 \%$ ) can lead to strain induced formation of three dimensional islands, however, a lattice misfit of 1.4 % is considered small enough to be ignored [90].

Even if not, the graphite lattice can easily relax to accommodate the lattice misfit. Indeed, such observations have been made before [91]. An illustration of a gold island on a graphite substrate is shown in Figure 4.5. For comparison, a gold island on Au (111) is also shown.

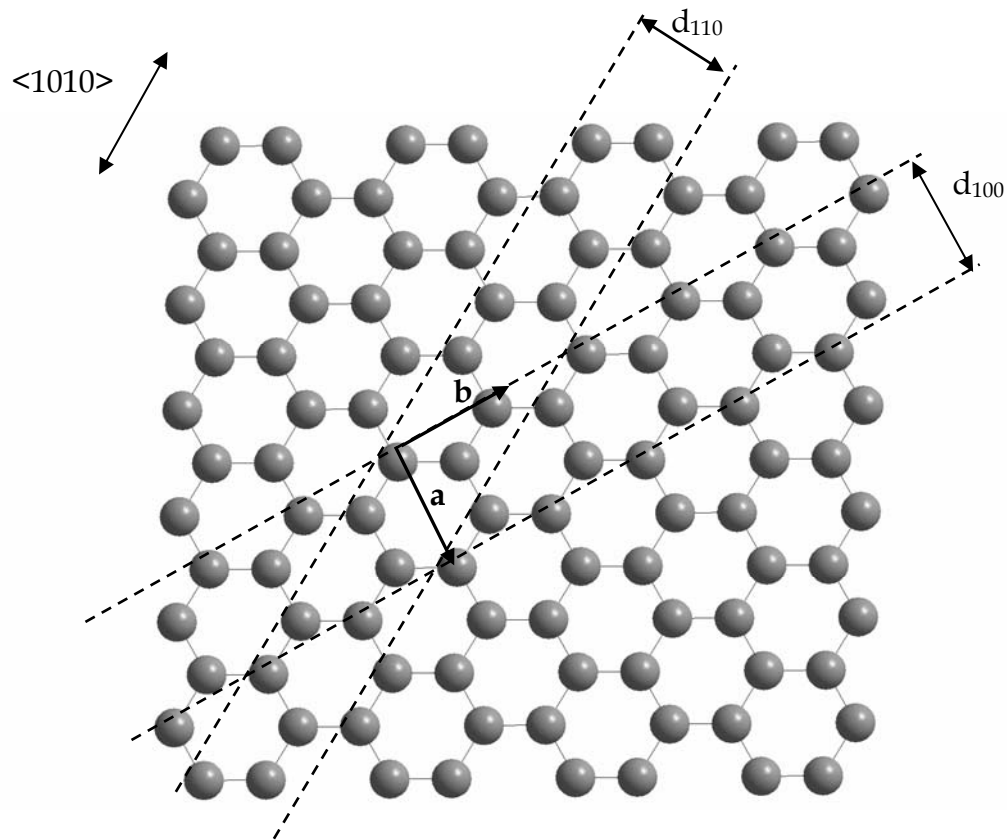


Figure 4.3: Schematic illustration of the graphite surface (basal plane). The distances  $d_{100}$  is 2.84 Å and  $d_{110}$  is 2.46 Å. The unit cell parameters is  $a = b = 2.46$  Å.

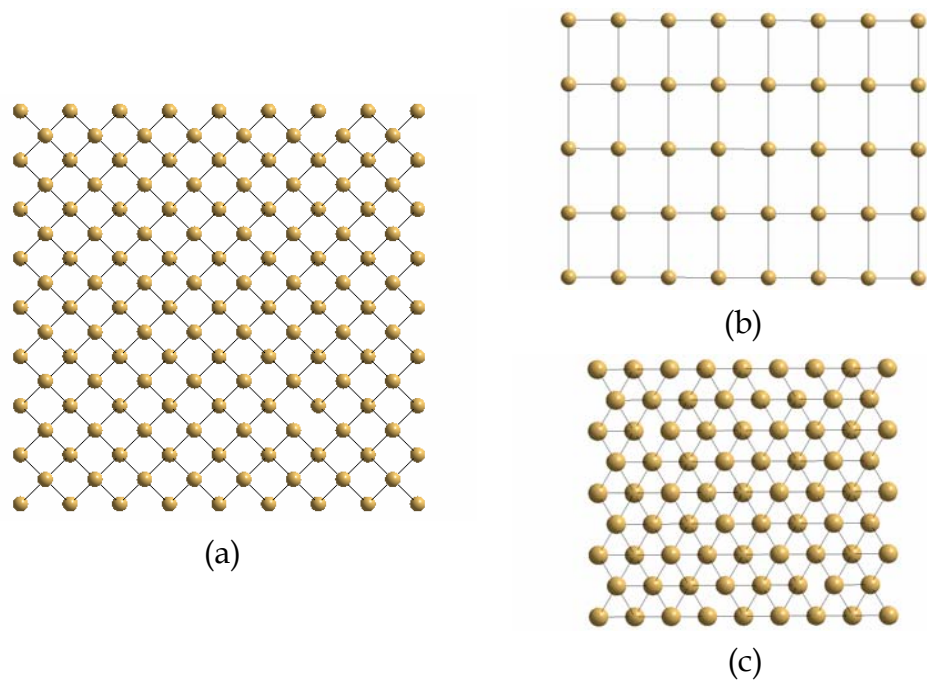


Figure 4.4: Schematic illustration of the (a) fcc (100), (b) fcc (110) and (c) fcc (111) surface structures of gold. The fcc (111) surface the lowest energy plane.

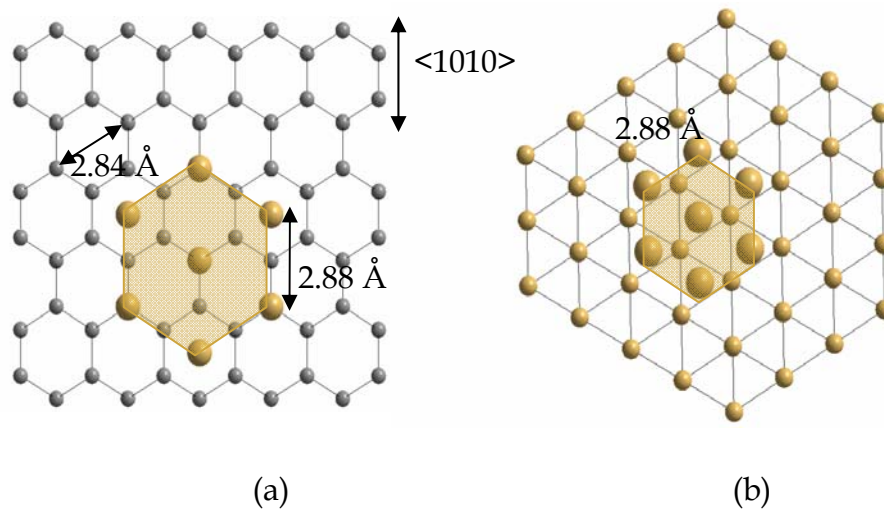


Figure 4.5: Schematic illustration of the positions of gold atoms on (a) graphite and (b) gold surface respectively. For clarity, the island gold atoms in (b) are drawn larger than the gold atoms in the substrate. The shaded region illustrates the shape of the gold island.

In light of the above discussion, an assumption was made which formed the foundation of the model developed to study the system. This was that the lattice is hexagonal, with inter-site separations of  $2.88 \text{ \AA}$ , which is the interatomic spacing of gold. Deposition and diffusion on graphite and gold is therefore performed on the same lattice. Only when the rates for the processes are calculated is it necessary to take into account whether an adatom is on graphite or on Au (111).

## 4.2 Identification of the relevant processes in the system

The KMC is a very powerful method to describe the growth of islands or nanostructures (or thin films) provided that all the relevant processes contributing to growth are considered. Neglect of some processes will result in unreliable information about size distribution and morphological behaviour of the grown island. The relevant processes that can occur during vapour deposition were indicated in Chapter 2. These processes might be sufficient to predict size distribution; however, to establish the morphology of the islands, which is dictated by the diffusion of adatoms around the edges, further information is needed. At any given time, an adatom diffusing along an island edge can occupy a certain site. Specific nomenclature can be assigned to these sites which can then be used to distinguish between the different diffusion processes that can occur along an island edge, also called edge diffusion (see Figure 4.6). Following this nomenclature, several step diffusion processes can be identified (see Figure 4.7). These are : (a) kink association, (b) dimer dissociation, (c) kink dissociation, (d) step diffusion, (e) corner diffusion and (f) corner crossing. Identification of the processes is based on the number of nearest

neighbours in the initial and final positions of the jumping atom, shown in Table 4.1.

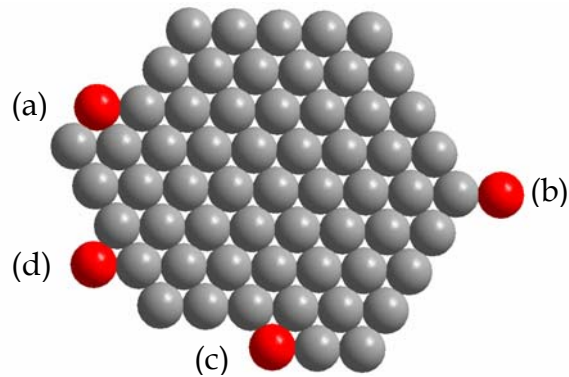


Figure 4.6: Definition of various types of atom structures that is attached to island edges and special sites the adatoms can occupy (a) step dimer, (b) corner site, (c) kink site and (d) step adatom.

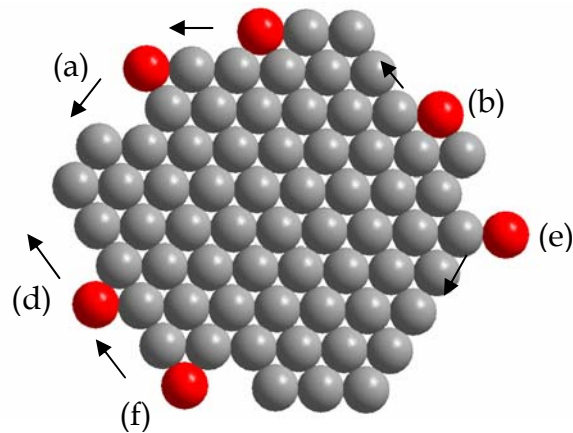
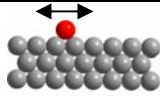
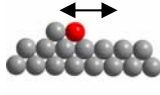
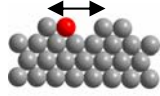
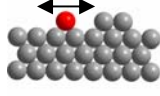
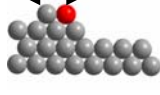
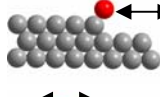
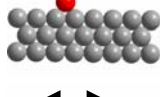
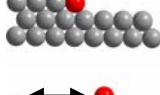
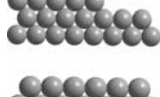
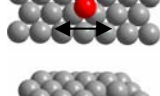
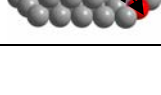


Figure 4.7: Various edge diffusion processes important for the modeling of island shape: (a) kink association, (b) dimer dissociation, (c) kink dissociation, (d) step diffusion, (e) corner diffusion and (f) corner crossing. Not shown is the diffusion of an adatom on an open terrace, dissociation from a step or desorption.

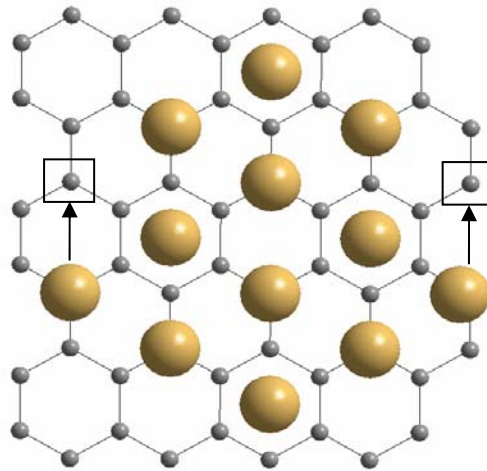


Table 4.1: Illustration of the different processes that can occur during growth. The processes can be identified according to their number of nearest neighbours in an initial ( $N_i$ ) and final state ( $N_f$ ) of a jumping atom.

Process	Configuration	$N_i$	$N_f$
Step Diffusion		2	2
Dimer association/ Dimer dissociation		3	2
Dimer-to-kink diffusion		3	3
Step-to-kink diffusion/ Kink-to-step diffusion		2	3
Corner-to-kink diffusion/ Kink-to-corner diffusion		3	1
Corner dissociation/ Corner association		1	0
Step dissociation/ Step association		2	0
Kink dissociation/ Kink association		3	0
Step-to-corner diffusion/ Corner-to-step diffusion		2	1
Terrace diffusion		0	0
Schwoebel Jumps		-	-

Edge diffusion of an adatom on graphite differs significantly from that of an adatom on Au (111). This difference arises from the morphology of the

underlying surface the diffusing adatom finds itself on. For an adatom on graphite, the morphology (and thus edge diffusion) of the underlying surface is the same, regardless at which step the adatom is (see Figure 4.8). Edge diffusion of an adatom on Au (111) (or a fcc (111) surface) is however not as simple. The morphology of the underlying surface is completely different depending on the plane that passes through the atoms of the step and the atoms of the substrate. From this, two types of close-packed steps can be identified, termed as either A or B-steps. The A-steps are made up of (111) facets and the B-steps are made up of (100) facets (see Figure 4.9). Due to the microscopic difference between these two steps, the transition states for the edge diffusion processes as well as the activation energies will be different (see Figure 4.10). The difference in step diffusion on the fcc (111) surface is a main driving force in the morphology of islands.



*Figure 4.8: Schematic illustration of a gold island with an adatom on either side of the steps. It is clear that the adatoms will have the same activation energy barrier for either step since the morphology of the graphite surface is the symmetric for the steps. The boxes indicate the positions to which the atoms should diffuse.*

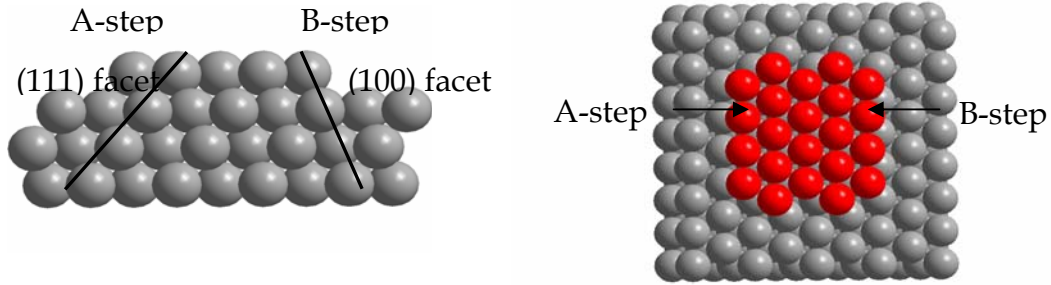


Figure 4.9: Schematic illustration of the two types of steps on an fcc (111) surface. The A step is bounded by a (111) facet and the B-step is bounded by a (100) facet.

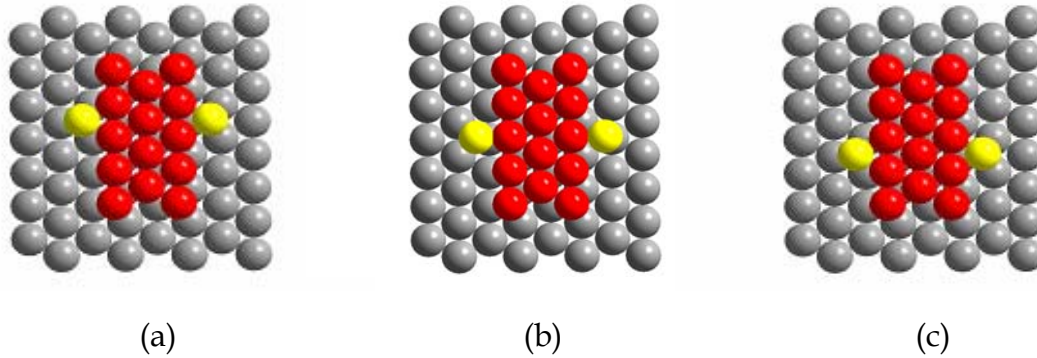
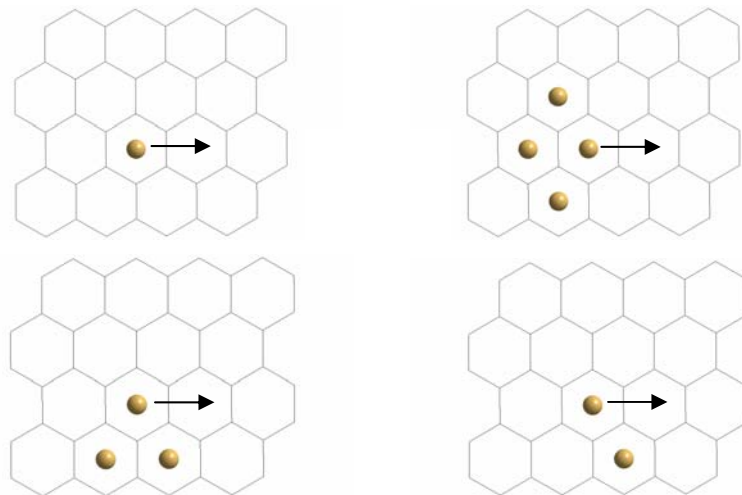


Figure 4.10: Schematic illustration of the various transition states on the fcc (111) surface on the A-step (left) and B-step (right). (a) initial state, (b) transition state and (c) final state.

### 4.3 Calculation of process rates

The relevant processes identified in Sections 2.3 and 4.2 have to be catalogued and their respective rates have to be determined if it is to be used in a KMC lattice gas model. As an illustration of how such a catalogue is constructed, consider the diffusion of a single adatom within one layer of a monoatomic

crystal surface with hexagonal symmetry (Figure 4.11). This can represent a gold adatom on either an existing gold island or on the graphite substrate. If the adatom is only permitted to hop to its nearest neighbours and its rate at each of these positions depends only on the nearest neighbourhood of 6 sites, there will be  $2^6 = 64$  possible configurations (or processes). In Figure 4.11, only four of these possible configurations are shown. For each of these configurations, a rate has to be assigned. Numerous studies have been conducted in which detailed catalogues were constructed to account for the rates of all the processes in the system [92]. Although such a methodology might seem conceptually simple at first glance, it becomes particularly unfeasible if more processes are considered.



*Figure 4.11: Diffusion hops of an adatom on a hexagonal lattice. In each picture, the adatom in the centre is assumed to hop to the right, and four different configurations of the neighbourhood are shown. In principle, there are  $2^6 = 64$  configurations. Considering symmetry, only 32 are distinct.*

For practical purposes or because the required detailed information about the rates might not be available, simplifying schemes are employed. The aim is to find efficient parameterizations of the relevant energy barriers in terms of a small number of independent quantities. Bond counting schemes have been particularly successful in this context [93]. The idea is to consider only very few distinct barriers, but to take into account the energies of the involved binding states explicitly. In some schemes only the energy of the initial state determines the rate [94], whereas, in general, initial and final configurations are considered [95]. In this study, however, a model is adopted in which the total energy barrier between two adjacent sites is given by [96]

$$E_{ij} = \delta E + \frac{1}{2}(E_j - E_i), \quad (4.1)$$

with  $\delta E$  the diffusion-energy barrier between the sites.  $E_j$  is the energy of the adatom at site  $j$  and  $E_i$  the energy at site  $i$  (see Figure 4.12).

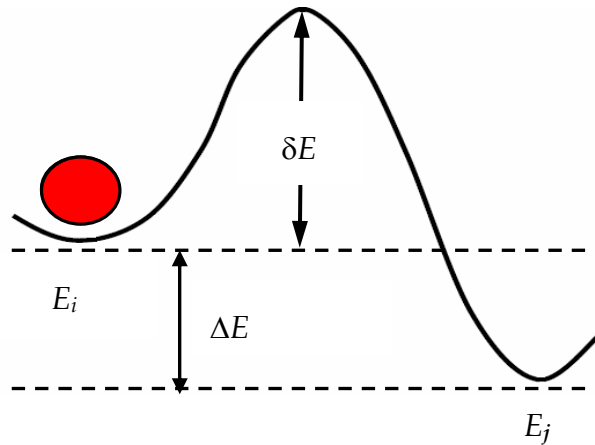


Figure 4.12: Illustration of the total energy barrier between two sites  $i$  and  $j$  with energy  $\Delta E$  between the two minima and additional barrier  $\delta E$ .

This total energy barrier can be used to calculate the rates for the processes from equation (3.48). The energy barriers shown in Figure 4.12 and equation 4.1 have to be determined from phenomenological arguments, experiments or theoretical calculations. In this study, molecular dynamics and energy minimisation algorithms were used to determine the necessary barriers. Details on these algorithms can be found in Chapter 5 and Appendix D. The total energy barriers for the processes calculated with the above mentioned methods are summarised and discussed in Chapter 5.

Usually, all the possible processes are stored in a table; however, this can become very complex (i.e searching through a very large table and accurately determining all the possible situations). Therefore, the model in this study takes into account the basic processes (as set out in Table 4.1) and calculates the proper energy barrier (see Appendix E) by counting the number of neighbours around the diffusing atom (within a certain interaction radius). The rate of the process is subsequently calculated using equation 3.48. This way the possibility of neglecting a relevant process is greatly reduced. The rates of the calculated processes should however be stored in a data structure which should be updatable and searchable during the course of a simulation.

#### **4.4 Data Structures for the storage of the process rates**

Even though the KMC method is an efficient way of evolving system dynamics, it can be very slow if the method of storing and selecting process rates (i.e. data) is not optimized. Several data structures can be employed for data storage, such as one or multi-dimensional lists, bins and trees [97]. An appropriate data structure for KMC specifically will however be determined by (i) accessibility of

data, (ii) memory management (iii) cost-effectiveness of search algorithm and (iv) scaling behaviour with system size. The primary data structure in this study was an AVL tree [97], since it conforms to the above mentioned criteria [97]. An AVL is merely a balanced binary tree [97] and contains a root node with several children or leaf nodes. Each node in the AVL tree is linked to an atom in the system and contains the sum of all the rates of the processes associated with that specific atom. Additionally, each of these nodes points to a linear list (secondary data structure in this study) in which the individual rates of the processes associated with each atom is kept. The list was constructed for this purpose since (a) it limits the size of the AVL tree, thereby reducing memory requirements and (b) it is faster to search through a “small” linear list than a very large AVL tree. An illustration of the AVL tree and linear list, and their connection, is shown in Figure 4.13. A further elaboration of the AVL tree and the linear list can be found in Appendix B.

In Figure 4.13 a square lattice is shown; however this is just a mapping of the hexagonal lattice studied in this work. This mapping was necessary to simplify the computer coding and is illustrated in Figure 4.14. For example, the accuracy of the simulation can be adjusted by the number of “rows” and “columns” included during the calculation of the rates. A specific row and column combination corresponds to a specific lattice site, which can be occupied or unoccupied. It is thus not necessary to change the code every time adjustment in accuracy is required. Details on the lattice object can be found in Appendix C.

Selection of an atom in the system which is most likely to move, as well as its associated process, is done by traversing the AVL tree and linear list respectively. The selection is based on comparison of the total rates of the atoms (as stored in the AVL tree) in the system with a random number in the interval  $[0, 1]$ . Similarly, the selection of a process associated with a selected atom is also

governed by a random number. This procedure is discussed in more detail in the next section.

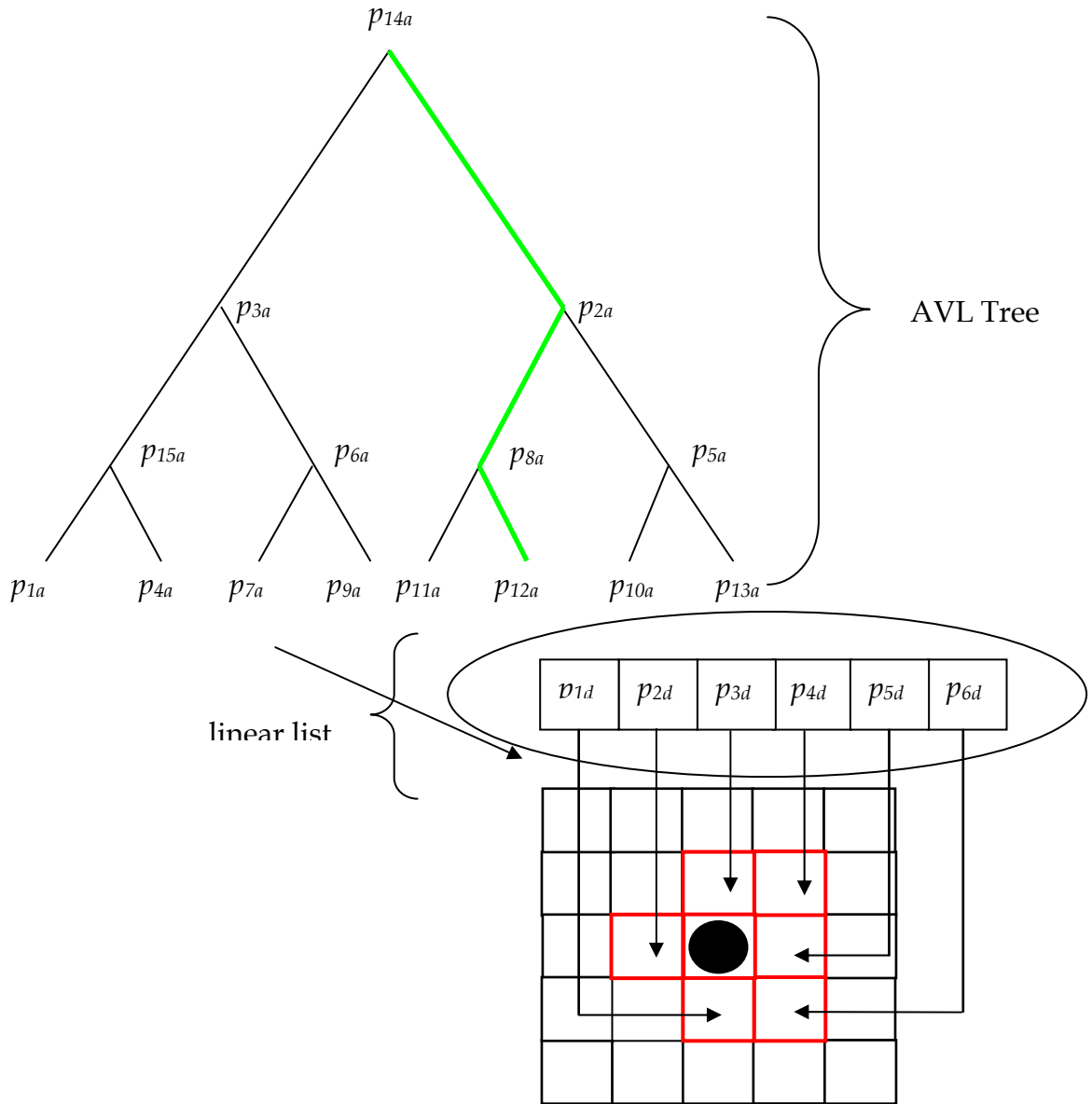


Figure 4.13: Schematic illustration of the AVL tree and linear list used for storing the process rates. In this example, atom 7 is selected which can jump to six nearest neighbours indicated by the red boxes (excluding inter-layer jumps). The lattice on which the selected atom jumps is shown as rectangular; however, this just is a hexagonal lattice “transformed” into a square one. Figure 4.14 explains this transformation.



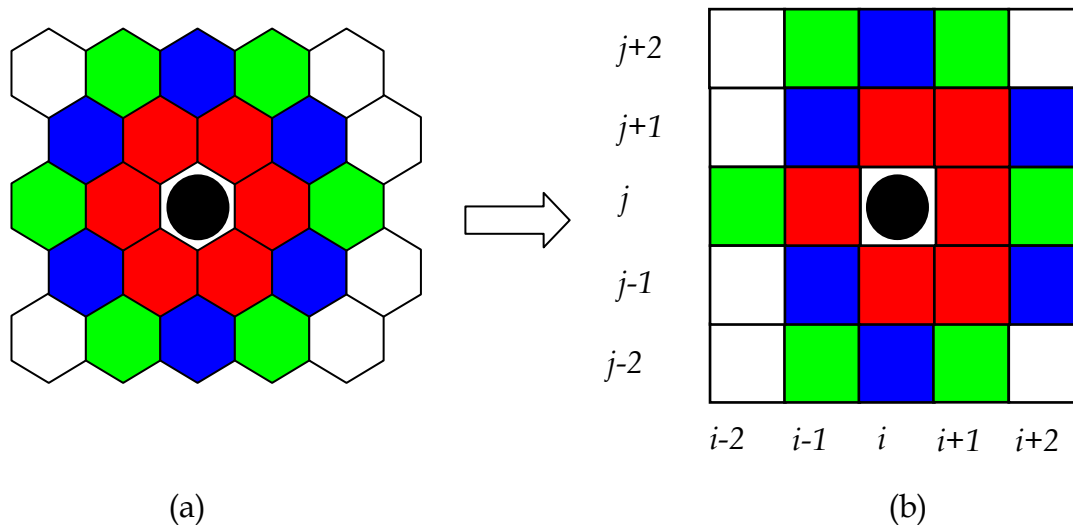


Figure 4.14: Illustration of the (b) square lattice constructed to represent the (a) hexagonal lattice. The red blocks represent first nearest neighbours, blue blocks the second nearest neighbours and green blocks the third nearest neighbours.

#### 4.5 An algorithm employing KMC to simulate nanostructure growth through vapour deposition

The final step in the implementation procedure is to combine the principles of the KMC method with the assumptions, identified processes and data structures (the model) in an algorithm which is capable of simulating growth through vapour deposition. However, some final assumptions concerning the growth process have to be made to complete the model. Different approaches can be followed to simulate growth, ranging from conceptually very simple to more complex. Many models consider only one adatom as mobile at a given time, and this atom is chosen to be the most recently deposited one. It may, for instance, move to an available empty site of lower height in the neighbourhood of the deposition site and become immobile. Thereafter, the simulation proceeds with the deposition

of another atom. Other models consider only the immediate incorporation of the particle upon the arrival at the surface, whereas activated diffusion over longer distances is completely neglected. These models are computationally very cheap and allow for the simulation of large systems and very thick. Consequently, such limited mobility models are employed, for example, in the investigation of basic phenomena like kinetic roughening. Their applicability in material specific modelling is, however, rather limited. In contrast, full diffusion models consider all atoms in the system mobile at the same physical time, which makes the simulation computationally very complex. In such full diffusion models, deposition can be considered as one of several possible processes. In this study however, deposition was treated separately. Given the deposition rate,  $F$ , the time between two successive deposition processes was calculated according to

$$t = 1/AF, \quad (4.2)$$

with  $A$  the area of the simulation area and  $F$  the deposition rate in monolayers per second. During this time, the lapse time, adatoms on the surface are considered mobile. Any of the processes summarized in Table 4.1 thus have a probability of occurring. The criteria for determining whether or not a process will occur is based on the weighted probability  $p_i$  of each process given by

$$p_i = r_i / R, \quad (4.3)$$

with  $r_i$  the rate of  $i$ -th process and  $R$  the total rate in the system of  $k$  processes

$$R = \sum_{i=1}^k r_i. \quad (4.4)$$

A random number,  $u$ , in the interval  $(0, 1]$  can then be used to select which process will occur according to

$$\sum_{i=1}^k r_i < u \leq \sum_{i=2}^{k-1} r_i . \quad (4.5)$$

The residence time (inverse of the rate) of the selected process can then be calculated and used to update the system time (i.e. the total time in the system). If the system time exceeds the lapse time, the next atom may be deposited, else the next process is selected. For convenience sake, the rates of the individual atoms were separated from the rates of the individual processes. In other words, each atom has numerous processes which it can undergo. The rates for each atom were summed up and assigned to that particular atom. This was done for each atom in the system. Selection of a particular atom, was then done in correspondence with equation 4.5. After an atom has been selected, the same procedure was followed to select a process. An algorithm for the simulation of vapour deposition can thus be written as follows (shown in Figure 4.15):

1. Read input parameters – deposition rate, temperature and total simulation time or number of atoms to be deposited.
2. Calculate the lapse time.
3. Deposit an atom.
4. Update the rates in the system.
5. Select an appropriate atom and process.
6. Add the residence time of the selected process to the system time.
7. If system time is less than lapse time, go to 4, else go to 3.
8. Repeat the procedure until the desired number of atoms has been deposited or until the total simulation time has been reached.

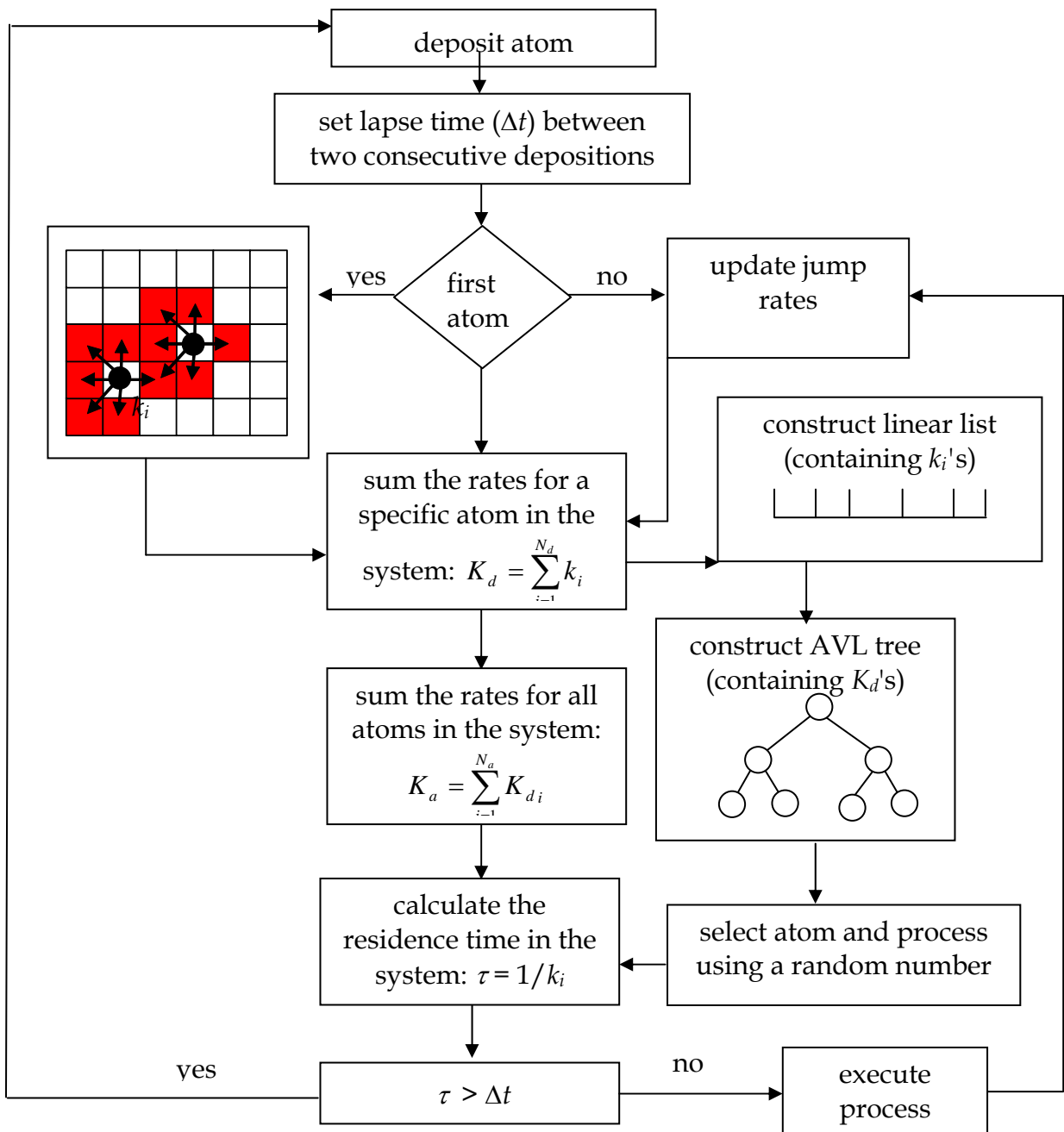


Figure 4.15: Summary of the algorithm to simulate vapour deposition.

## **CHAPTER 5**

### **Calculational Details, Results and Discussion**

The implementation of the KMC method was discussed in the previous chapter and a full diffusion model was presented to simulate the growth of gold nanostructures on the graphite surface. Although this was the envisioned model, development thereof occurred at different stages, each characterised by the assumptions made to describe the system and its dynamics. Therefore, an initially very robust model evolved to one with powerful predictive capabilities, provided of course that the inputs are reliable and trustworthy. This model is capable of simulating growth at thermodynamic equilibrium as well as at conditions subjected to kinetic constraints. This chapter thus commences with a discussion of what is to be expected for the growth of gold on graphite from a purely thermodynamic point of view. Furthermore, for each stage, the

assumptions made are set out and simulation results, considering kinetic constraints, are presented. Artifacts arising from the assumptions are also discussed for each stage.

## 5.1 Preliminary Predictions

The effect of thermodynamics on the growth of gold on graphite can be deduced from equations (2.14) and (2.15) given in Chapter 2. These equations are

$$\gamma_s = \gamma_d \cos \theta + \gamma_{\text{int}}; \quad (2.14)$$

$$E_{\text{adh}} = \gamma_s + \gamma_d - \gamma_{\text{int}}; \quad (2.15)$$

The values for  $\gamma_d$ ,  $\gamma_i$  and  $E_{\text{adh}}$  are 1.354 N/m, 0.89 N/m and 0.514 N/m respectively [98]. In order to establish whether Frank-van-der-Merwe, Volmer-Weber or Stranski Krastanov growth occur, the (i) contact angle and (ii) the relationship between the various surface energies have to be determined. The surface energy of graphite,  $\gamma_s$ , has to be calculated since values from literature are too contradictory to use [99]. This is done by substituting  $\gamma_d$ ,  $\gamma_i$  and  $E_{\text{adh}}$  into equation (2.15) which yields  $\gamma_s = 0.05$  N/m. The contact angle can subsequently be determined by substituting  $\gamma_d$ ,  $\gamma_i$  and  $\gamma_s$  into equation (2.14) indicating a contact angle of approximately  $128^\circ$ . It can therefore be concluded that Volmer-Weber growth can be expected for the growth of gold on graphite (as discussed in Chapter 2).

The equilibrium morphology of the (three-dimensional) gold nanostructures can be determined from the Wulff-construction. However, as mentioned before,

growth during vapour deposition usually occurs at non-equilibrium circumstances and kinetics aspects play a substantial role in controlling the morphology of islands which are indeed addressed by the KMC model developed in this study.

## 5.2 Development Stages

In all of the development stages, the assumptions discussed in Chapter 4 apply, except those related to how the processes are selected during a simulation and how the lattice was constructed. These two assumptions were modified from one stage to another as more insight was gained into the dynamics of the system and artifacts arising due to the assumptions were noticed. The simulations were performed for monolayer and multilayer deposition. Monolayer deposition implies the constraint of growth in the  $z$ -direction (i.e island growth), whereas multilayer deposition allows growth in the  $z$ -direction (i.e nanostructure growth). In the latter case, atoms may be deposited on top of existing islands or nanostructures and the Schwoebel barrier have to be taken into account.

In addition to the above assumptions, the interaction range of the Sutton-Chen potential, used to describe the gold-gold interaction, was taken as approximately  $10 \text{ \AA}$  (see Figure 5.1). At this distance, the interaction between two atoms drops to 0.78 % of the interaction of the distance at the bulk value ( $2.88 \text{ \AA}$ ). Ideally, a much larger interaction area should be considered, however this increases computational complexity significantly (more neighbours have to be calculated for each simulation step) and speed must be sacrificed for improved accuracy.

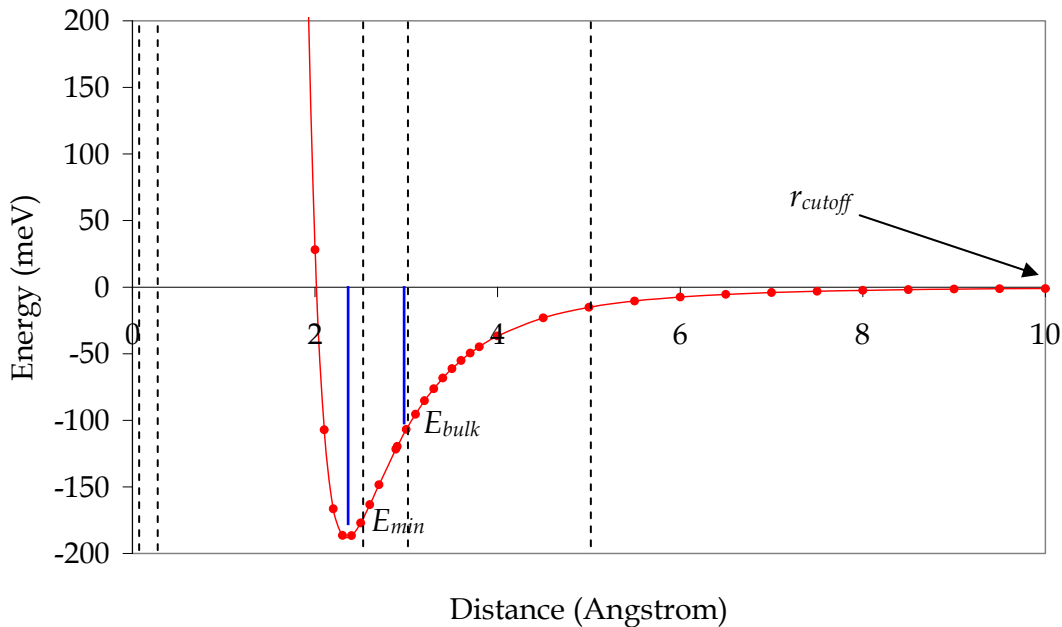


Figure 5.1: Potential energy curve for the Sutton-Chen potential.

The values of the energy barriers in the simulations were 0.005 eV and 0.12 eV for terrace diffusion on graphite and Au (111) respectively, 0.34 eV for diffusion along an A-step, 0.24 eV for diffusion along a B-step and 0.15 eV for diffusion along a step for an island on graphite. Desorption from Au (111) occurred at 0.7 eV and at 0.3 eV from graphite if the desorbing atom has no neighbours.

The energy barriers for diffusion along the A and B-step were taken from literature [104], and that of diffusion along an island edge on graphite and desorption was calculated with a self-written energy minimization code [105] utilizing the Sutton-Chen, Lennard-Jones and Brenner potential for the gold-gold, gold-carbon and carbon-carbon interactions respectively. In order to assess the accuracy of the barriers taken from literature, the same energy minimization code was used to calculate the barriers for A and B-step diffusion. Using this



methodology, energy barriers were calculated as 0.304 eV and 0.27 eV for the A and B-step respectively, which are in reasonable agreement with the values from literature. Details on these calculations are given in Appendix E.

It should be noted that for the first stage, the energy barriers used were 0.00197 eV and 0.028 eV for terrace diffusion on graphite and gold respectively and for step diffusion along an island on graphite and 0.32 eV for step diffusion along a gold island. A-steps and B-steps were not yet taken into account at this stage. These were also calculated using a self-written energy minimization code [105]; however, at this stage the code was not fully developed. The reason for using these values instead of those from literature was because at Stage 1, values from literature were not available yet.

From the above values, the activation energies,  $\Delta E$ , required for the different step diffusion processes to occur can be determined. This is done "*on-the-fly*" during a simulation. Depending on the level of sophistication required, as many or few neighbours can be included in a calculation, as mentioned in Chapter 3. The mobility of the atoms can also be influenced by its nearest neighbours. In this study, atoms were considered mobile unless they had six nearest neighbours (i.e bulk). The activation energies for the different processes discussed in Chapter 4 are summarised in Table 5.1. The activation energies shown in Table 5.1 were calculated using the energy barriers for Stage 2 - Stage 4, not Stage 1.

It can be noted that the activation energies in Table 5.1 are representative of situations where the diffusing atom has the maximum number of next nearest, third nearest and fourth nearest neighbours. Variation of the number of neighbours implies that the activation energies will vary too. For example, dimer dissociation on graphite has an activation energy of -0.08 eV (from Table 5.1).

However, this is only true if the number of next nearest neighbours is three and the number of third nearest neighbours is also three. If the number of third nearest neighbours were instead two whilst the number of next nearest neighbours remained the same, the activation energy would change to 0.02 eV.

Table 5.1: Summary of activation energy,  $\Delta E$ , for the diffusion of a gold adatom around an island edge on graphite and Au(111). Only those processes discussed in Chapter 4, Table 4.1 are shown. The processes are identified by  $N_i$  and  $N_f$ , meaning that, for the step diffusion process, with  $N_i = 2$  and  $N_f = 2$ , the process can be represented by  $2 \rightarrow 2$ .

Process	$\Delta E$ on Au(111) A-Step (eV)	$\Delta E$ on Au(111) B-Step (eV)	$\Delta E$ on Graphite (eV)
1 $\rightarrow$ 0	0.47	0.47	0.29
1 $\rightarrow$ 1	0.34	0.24	0.16
1 $\rightarrow$ 2	-0.05	-0.06	-0.1
1 $\rightarrow$ 3	0.01	-0.08	-0.28
2 $\rightarrow$ 0	0.57	0.57	0.55
2 $\rightarrow$ 1	0.72	0.62	0.41
2 $\rightarrow$ 2	0.34	0.24	0.16
2 $\rightarrow$ 3	0.25	0.16	-0.02
3 $\rightarrow$ 0	0.75	0.75	0.75
3 $\rightarrow$ 1	0.73	0.64	0.59
3 $\rightarrow$ 2	0.43	0.32	0.34
3 $\rightarrow$ 3	0.34	0.24	0.16

The rates for some the processes summarised in Table 5.1 were calculated using equation 3.48 and the influence of temperature on their activation is shown graphically in Figures 5.2 to 5.4. From the figures it can be elucidated when a process is activated. This is the case if there is a rapid increase in the magnitude of the rate for small temperature changes. Activation of a process does however not necessarily imply that it has a probability to occur. Provided that the

residence time of the process, added to the system time, is not greater than the lapse time, the process has a probability to occur, otherwise it cannot. It should be mentioned that the Schwoebel barrier, as indicated in Table 5.1, can also vary depending on the number of neighbours in the initial and final state, as discussed before.

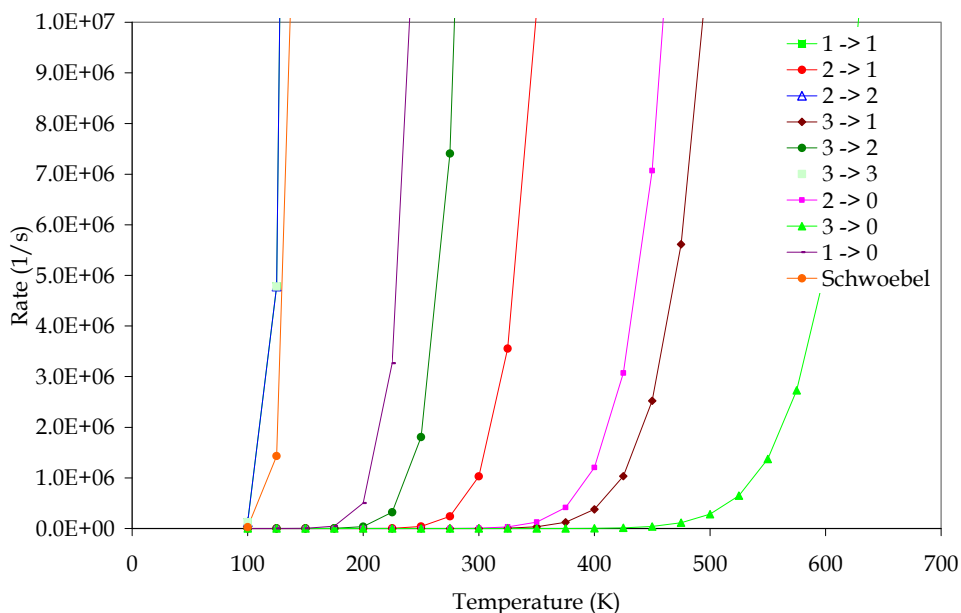


Figure 5.2 : Rates for different processes on a step of a Au island on graphite. The different curves apply to jumps characterised by the change in the number of nearest neighbours in the surface plane.

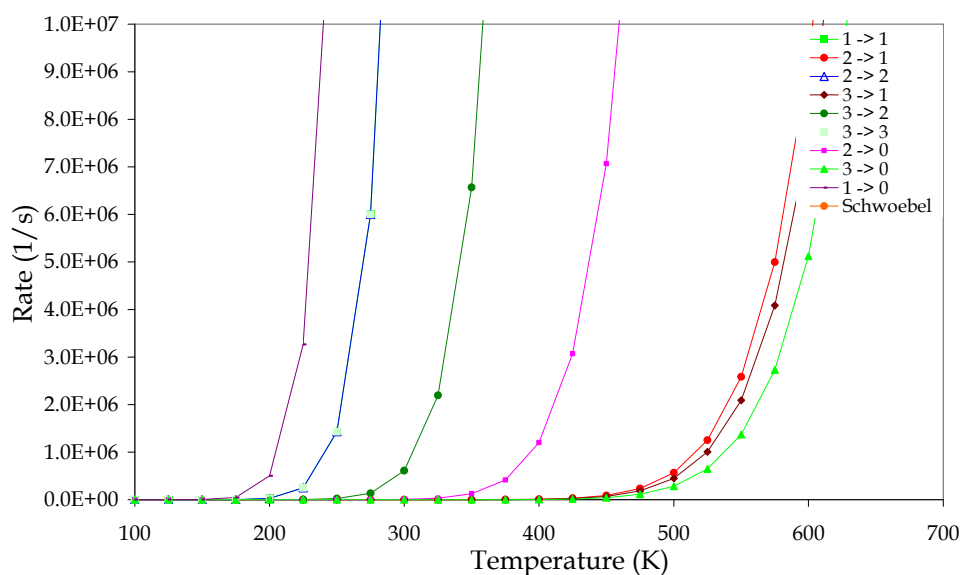


Figure 5.3 : Rates for different processes on an A-step of a Au island on Au (111). The different curves apply to jumps characterised by the change in the number of nearest neighbours in the surface plane.

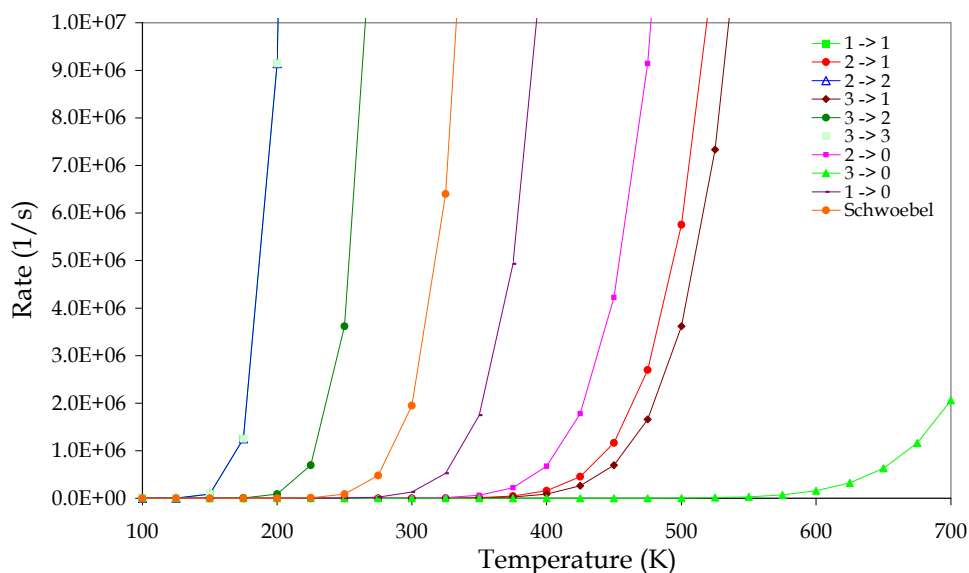


Figure 5.4 : Rates for different processes on the B-step of a Au island on Au (111). The different curves apply to jumps characterised by the change in the number of nearest neighbours in the surface plane.

The different stages and the simulations performed at each stage are discussed in detail in the next three sections.

### 5.2.1 Stage 1

The assumptions made in this stage, in addition to those in Chapter 4, include the following:

- Only  $\alpha$  and fcc sites on graphite and Au (111) respectively were considered as possible adsorption sites and jumps were only allowed to neighbouring  $\alpha$  or fcc sites;
- Following deposition, only the most recently deposited atom was allowed to move, with the other atoms in the systems considered fixed;
- The process with the largest probability was selected to occur ( $p \propto e^{-\Delta E/k_B T}$ ).
- The desorption process was neglected.

Simulations were performed for monolayer and multilayer growth at different deposition rates, temperatures and nucleation sites. Deposition occurred at randomly selected sites. The details of the simulations are given below together with the results. Before the results are interpreted though, it should be mentioned that considerable care has to be taken to differentiate between real physical effects (due to the growth conditions and processes that occur) and simulation artifacts.

### 5.2.1.1 Monolayer growth

The deposition rates probed in the simulation for monolayer deposition were between  $5 \times 10^{-1}$  10ML/s and  $5 \times 10^{-4}$  ML/s and temperatures ranged between 100 K and 600 K. A total number of 50 nucleation sites were randomly placed on a  $200 \times 200$  lattice. The surface coverage after deposition was 0.25 ML. The following deductions can be made from the simulation (see Table 5.2):

- Fractal-like islands are observed for all the deposition rates at 100 K although the number of the fractal-like islands increases ( $> 50$ , which is the number of nucleation sites) with increasing deposition rate and decreasing temperature. This increase is due to the creation of new nucleation sites, since the probability of atoms to reach existing islands decreases with an increasing deposition rate. The reason for fractal-like behaviour at 100 K is two-fold: firstly, there may be certain processes that are not activated at this temperature and secondly, some processes may be activated, but the deposition rate may be too fast to allow them to occur. The two processes dictating the fractal-like behaviour are step diffusion and step-to-corner diffusion (the latter not being relevant in this simulation as discussed below). Neither of these two processes is activated at 100 K (from Figure 5.2 it can be seen that a process with an energy barrier of 0.32 eV to occur will only be activated at around 250 K, i.e. where the magnitude of the rate increases rapidly for a small temperature change). The step-to-corner diffusion process can not occur in any case because the model only allows the process with the largest probability to occur. Based on the above discussion, it can therefore be concluded that the growth of islands occurs via a “hit-and-stick” mechanism.

- A transition from fractal-like to compact island morphology is made between 100 K and 300 K at deposition rates of  $5 \times 10^{-4}$  ML/s and  $5 \times 10^{-3}$  ML/s. The apparent transition can be ascribed to the activation of the step diffusion process. Atoms can therefore diffuse along island edges toward the centre of an island. For the deposition rates of  $5 \times 10^{-2}$  ML/s and  $5 \times 10^{-1}$  ML/s at 300 K, islands are still fractal-like since the rate for the step diffusion process is slower at this temperature and fewer atoms can reach the center of an island. The fractal-like arms are however thicker than those at 100 K, indicating that atoms were able to diffuse along the island edges.
- Another apparent transition from fractal-like to compact islands is made between 300 K and 500 K for deposition rates of  $5 \times 10^{-2}$  ML/s and  $5 \times 10^{-1}$  ML/s. At 500 K, the rate for the step diffusion process is fast enough so that more atoms can redistribute themselves along the island edges and therefore they have a higher probability to reach the centre of an island.
- At 500 K there is not much observable change in the island morphology for the deposition rates of  $5 \times 10^{-4}$  ML/s and  $5 \times 10^{-3}$  ML/s and this is also the case for all the deposition rates at 600 K. This behaviour is understandable however, since the process with the largest probability is selected to occur during the simulation. Processes that have only a slight probability to occur (such as step-to-corner diffusion) are not allowed to and the island morphology is therefore simulated as irregularly compact.

The island density (number of atoms per number of sites) for the different deposition rates and temperatures can also be determined and is shown in Figures 5.5 and 5.6 respectively. There is a decrease in the island size (i.e the

number of sites it occupies) and an increase in the island density with increasing temperature. This indicates the apparent transition from fractal-like to compact islands.

Table 5.2: Island morphologies for monolayer growth at different temperatures and deposition rates using the assumption that only the most recently deposited atom is mobile at a given time and considering only the largest probability in the system. The coverage is 0.25 ML. A total number of 50 nucleations sites were randomly placed on the  $200 \times 200$  substrate.

	T = 100K	T = 300K	T = 500K	T = 600K
R = $5 \times 10^{-4}$ ML/s				
R = $5 \times 10^{-3}$ ML/s				
R = $5 \times 10^{-2}$ ML/s				
R = $5 \times 10^{-1}$ ML/s				



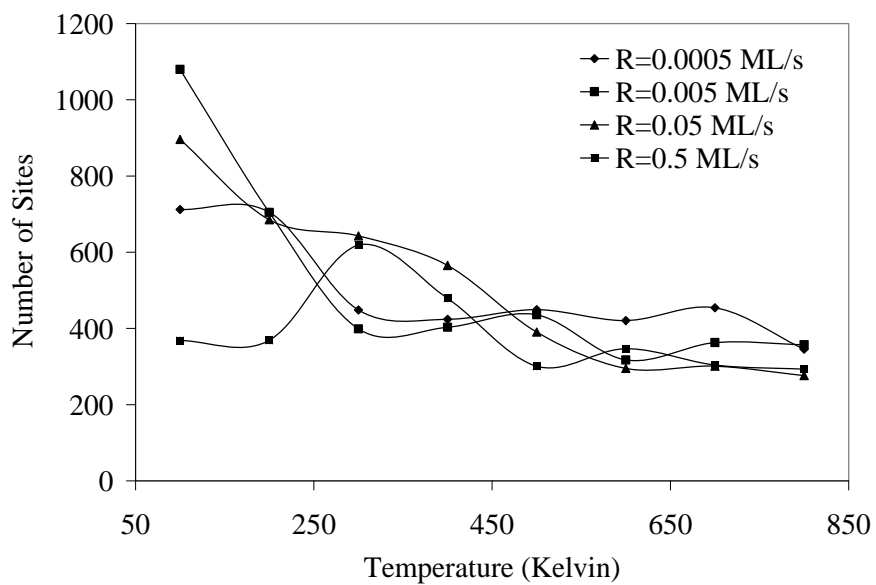


Figure 5.5: Average island size (number of sites) for different temperatures at a given deposition rate.

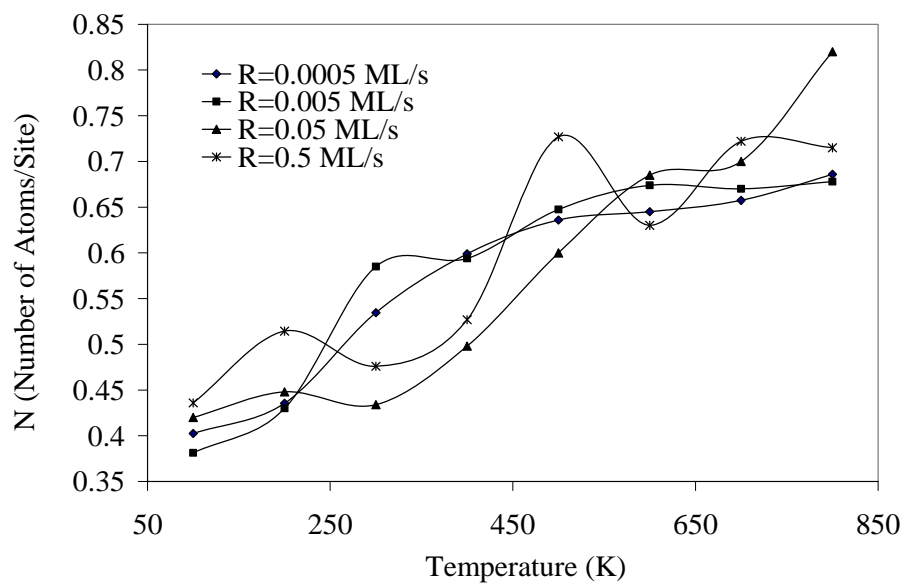


Figure 5.6: Average island density for different temperatures and deposition rates.

### 5.2.1.2 Multilayer growth

The deposition rates used in the simulation for multilayer growth were between 0.05 ML/s and 10 ML/s and temperatures ranged between 100 K and 600 K. A total number of 25 nucleation sites were randomly placed on the substrate. The surface coverage after deposition was 0.4 ML. The following deductions can be made from the simulation (see Table 5.3):

- Fractal-like islands are observed for all the deposition rates at the lowest temperature (100 K). The number of the fractal-like islands increases ( $> 25$ , which is the number of nucleation sites) with increasing deposition rate and decreasing temperature. This is particularly noticeable at a deposition rate of 10 ML/s at 100 K in which case the fractal-like islands are very small. The same explanation as for the case of monolayer deposition applies - new nucleation sites are created because the probability of atoms to reach existing islands decreases with an increasing deposition rate. The fractal-like behaviour at 100 K can however not be explained in terms of step diffusion. At this temperature, the step diffusion process is not activated; however, the Schwoebel jump process is (see Figure 5.1) but the rate for this process is too slow to allow enough atoms to jump on top of an island and thereby reduce the fractal-like morphology. Similar to the case of monolayer deposition, at high deposition rates, there are fewer atoms that can jump on islands and growth also occurs via a "hit-and-stick" mechanism. One should however not lose sight of the fact that a substantial amount of time is spent by an atom merely diffusing around on a terrace. Therefore, by the time an atom reaches an island, very little time is left for diffusion along an edge, thus enhancing the probability of "hit-and-stick" growth.

- Between 100 K and 300 K, there is an apparent transition from fractal-like to compact islands for 0.05 ML/s. At this temperature, the rate for the Schwoebel jump process is much faster and more atoms have the probability to jump on top of island, thereby decreasing the size and increasing the density of the island. The islands are still fractal-like for the other deposition rates, although they appear denser, due to the aforementioned reason.
- At 500 K, the nanostructures for the deposition rates 0.5 ML/s, 1 ML/s and 10 ML/s also appear to be more compact. Yet again, at higher temperatures the rate for the Schwoebel jump process is much faster and more atoms can jump on the nanostructures, decreasing their size.
- Lastly, at 600 K, there is not much observable change in the morphology of the nanostructures at any of the deposition rates. All that can be noticed is the number of nanostructures is slightly less than for the lower temperatures. For 0.05 ML/s and 0.55 ML/s this number is close to the number of nucleation sites, namely 25.

As before, the nanostructure density (number of atoms per number of sites) for the different deposition rates and temperatures can also be determined and is shown in Figures 5.7 and 5.8 respectively. Again, there is a decrease in the island size and increase in island density with increasing temperature indicating the transition from fractal-like to compact islands.

Table 5.3: Nanostructure morphologies for multilayer growth at different temperatures and deposition rates on a  $200 \times 200$  substrate using the assumption that only the most recently deposited atom is mobile at a given time and considering only the largest probability in the system. The coverage is 0.4 ML. A total number of 25 nucleations sites were randomly placed on the lattice.

	T = 100K	T = 300K	T = 500K	T = 600K
R = 0.05 ML/s				
R = 0.5 ML/s				
R = 1 ML/s				
R = 10 ML/s				

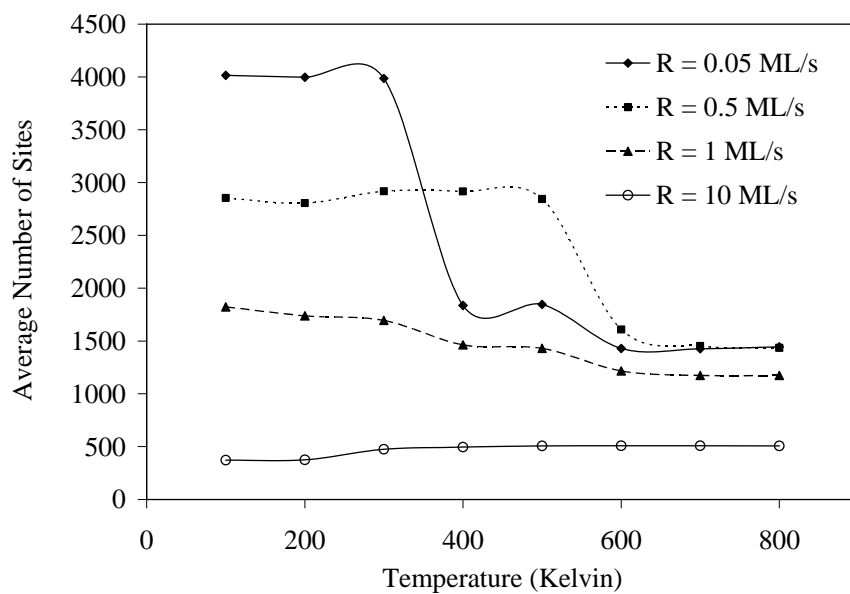


Figure 5.7: Average nanostructure size (number of sites) for different temperatures at a given deposition rate.

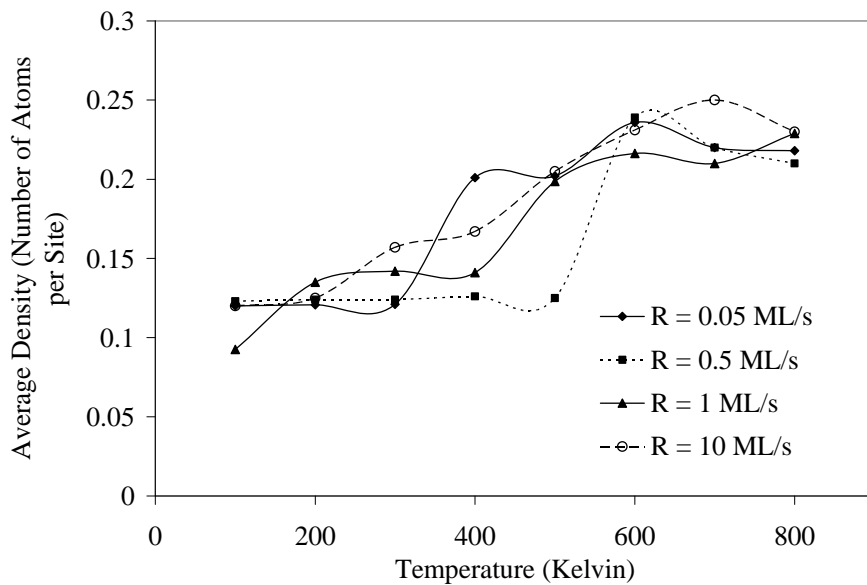


Figure 5.8: Average nanostructure density for different temperatures and deposition rates.

### 5.2.1.3 Simulation Artifacts

The reliability of the results obtained from simulations performed in this stage is hindered by what can be termed as simulation or model artifacts. Only those artifacts that can be attributed to the assumptions made are discussed here, although there may be others that are not as clear to see. These artifacts explain the somewhat unexpected morphologies of the islands/nanostructures and are as follow:

- *Largest probability process selection:* Selecting only the process with the largest probability prohibits the occurrence of processes that lead to the rounding or faceting of islands. Processes with the largest probability are step diffusion and Schwoebel jumps for monolayer and multilayer growth respectively. As a consequence of this artifact, islands/nanostructures assume an irregular morphology with protuded arms. This is because atoms that arrive at tips or corner sites cannot diffuse around to a step and merely stick where it attached.
- *Moving only the most recently deposited atom:* It is difficult to assess the influence of this assumption on the island/nanostructure morphology because of the large role played by the above assumption. If it is accepted that the above assumption holds, moving only the most recently deposited atom is not expected to have such a great effect, since, in most cases; an atom deposited on a graphite substrate will have the highest probability to occur. For example, terrace diffusion has an energy much smaller than step diffusion (0.00197 eV as opposed to 0.32 eV). However, if an atom is deposited next to a kink site (or if it attaches from the terrace), there might be an atom that is not attached to any kink site and that would therefore

have the highest probability to move. If the residence time required to execute the kink dissociation process added to the system time exceeds the lapse time, the next atom would be deposited, even though step diffusion (which will have the largest probability) could have occurred. Therefore, islands will not assume their correct morphology, and will therefore have a morphology which is more irregular than that of the physical situation. In the case of multilayer growth, if an atom is deposited on a terrace, it will also have the largest probability to occur of all the other processes in the system. Similar to the case of monolayer growth, an atom may attach or be deposited next to a kink site and, due to the assumption, be selected to occur even though there may be a larger probability process in the system. For both monolayer and multilayer deposition, a new nucleation centre (created when the system time exceeds the lapse time before an atom reaches an island) may have a higher probability to move than an atom deposited at any position on a step.

- *Considering only fcc and  $\alpha$  absorption and jumping sites:* In the case of monolayer growth, this is not expected to have an influence on the island morphology. However, during multilayer growth this assumption may lead to a pronounced effect, which will be discussed in more detail in the next section.
- *Neglecting the desorption process:* At this stage, neglect of this process does not appear to have much of an effect. It will most likely just increase the time needed to perform a simulation since more time will be needed to deposit a certain amount of atoms if a percentage of that is desorbed from the surface.

## 5.2.2 Stage 2

Evidently, selecting the process with the highest probability results in serious simulation artifacts by suppressing the fundamental physical processes that can occur during growth. Consequently, Stage 2 focused on using a random number to select the most probable process, as discussed in Chapter 4. In this stage, the data structures, random number generator and search algorithm were developed and implemented. Therefore, the assumptions made in this stage, in addition to those in Chapter 4, include the following:

- Only  $\alpha$  sites on graphite and fcc sites on Au (111) were considered as possible adsorption sites and jumps were only allowed to neighbouring  $\alpha$  or fcc sites;
- Following deposition, only the most recently deposited atom was allowed to move, with the other atoms in the systems considered fixed;
- Process selection was made by comparing the weighted probability of each process in the system with a random number in the interval  $[0, 1]$ ;
- The desorption process was neglected.

The simulations performed in Stage 1 were repeated using the modifications in the assumptions summarised above. It should be emphasised again that simulation artifacts may be present which can skew the results presented below.



### 5.2.2.1 Monolayer growth

The same deposition rates and temperatures as those used in Stage 1 for monolayer growth were used in Stage 2. Deposition rates ranged from  $5 \times 10^{-1}$  ML/s to  $5 \times 10^{-4}$  ML/s and temperatures from 100 K to 600 K. A total number of 50 nucleation sites were randomly placed on the substrate and the surface coverage after deposition was 0.25 ML. The following deductions can be made from the simulation (see Table 5.2):

















- Similar to the previous simulations, islands are fractal-like at 100 K for all the deposition rates. This can again because the step diffusion process is not activated at 100 K and atoms stick to an island edge wherever they attach (refer to Section 5.2.1.1).
- An apparent transition from compact to fractal-like islands appear between 100 K and 300 K for  $5 \times 10^{-3}$  ML/s and  $5 \times 10^{-4}$  ML/s, whereas the fractal-like arms of the islands at deposition rates of  $5 \times 10^{-1}$  ML/s and  $5 \times 10^{-2}$  ML/s at 300 K are much thicker. As mentioned in Section 5.2.1.1, the step diffusion process is activated at 300 K and atoms therefore have a probability to diffuse along an island edge which can either result in the thickening of the fractal-like arms or in a compact morphology. The step-to-corner diffusion process is not yet activated at 300 K and the island morphology at deposition rates of  $5 \times 10^{-3}$  ML/s and  $5 \times 10^{-4}$  ML/s is therefore still irregular compact.
- An apparent transition from compact to fractal-like islands for deposition rates of 0.5 ML/s and 0.05 ML/s occur between 300 K and 500 K. The faster rate for the step diffusion process is responsible for this behaviour.

The step-to-corner diffusion process, although activated at this temperature (with a step diffusion barrier of 0.32 eV, the step-to-corner diffusion process has a barrier of approximately 0.72 eV, which is indicated in Table 5.1 and Figure 5.3), is too slow to occur at a temperature of 500 K and rates of 0.5 ML/s and 0.05 ML/s. Furthermore, the arms of the fractal-like islands become thicker for the two highest deposition rates at a higher temperature. The same explanation as before applies here.

- There are other similarities with the results from Stage 1. These are the increasing island density with increasing deposition rates and decreasing temperatures. Again, this is due to the fact that more nucleation sites are created with an increasing deposition rate and decreasing temperature.
- The most important difference between Stage 1 and Stage 2 is the facetting of the islands at 600 K for deposition rates of  $5 \times 10^{-3}$  ML/s and  $5 \times 10^{-4}$  ML/s. At this temperature, the rate for the step-to-corner diffusion process is fast enough for it to have a probability to occur. The higher deposition rates still exhibit irregular morphologies, indicating that the deposition rates are too fast for the step-to-corner diffusion process to occur.

The island density (number of atoms per number of sites) for the different deposition rates and temperatures can also be determined and is shown in Figures 5.9 and 5.10 respectively. Like before, there is a decrease in the island size and increase in island density with increasing temperature indicating the transition from fractal-like to compact islands.

Table 5.4: Island morphology for various temperatures and deposition rates on a  $200 \times 200$  substrate with a coverage of 0.25 ML with 50 nucleation sites. Process selection was done by comparing the weighted probability of each process in the system with a random number in the interval  $[0, 1]$ . Only the most recently deposited atom was allowed to move between depositions.

	T = 100K	T = 300K	T = 500K	T = 600K
R = $5 \times 10^{-4}$ ML/s				
R = $5 \times 10^{-3}$ ML/s				
R = $5 \times 10^{-2}$ ML/s				
R = $5 \times 10^{-1}$ ML/s				

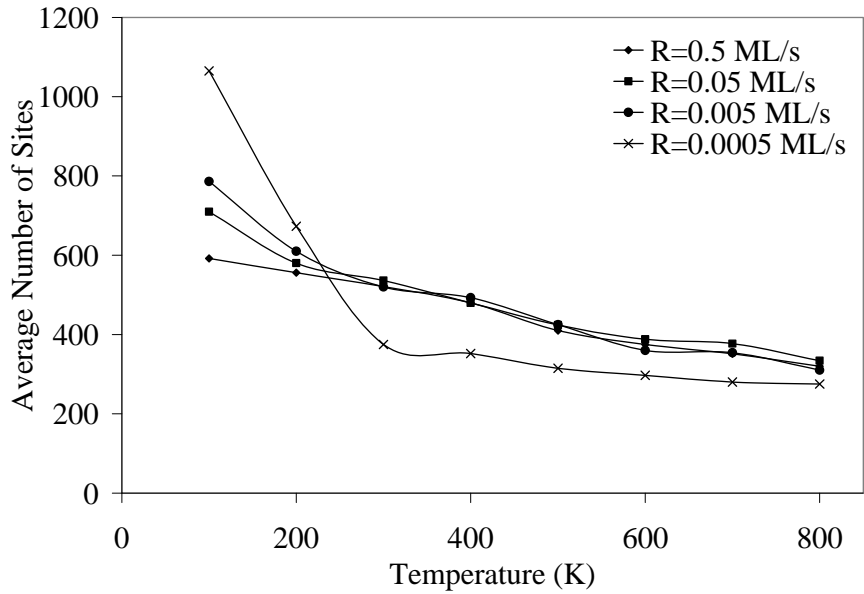


Figure 5.9: Average island size (number of sites) for different temperatures at a given deposition rate with a coverage of 0.25 ML and 50 nucleation sites.

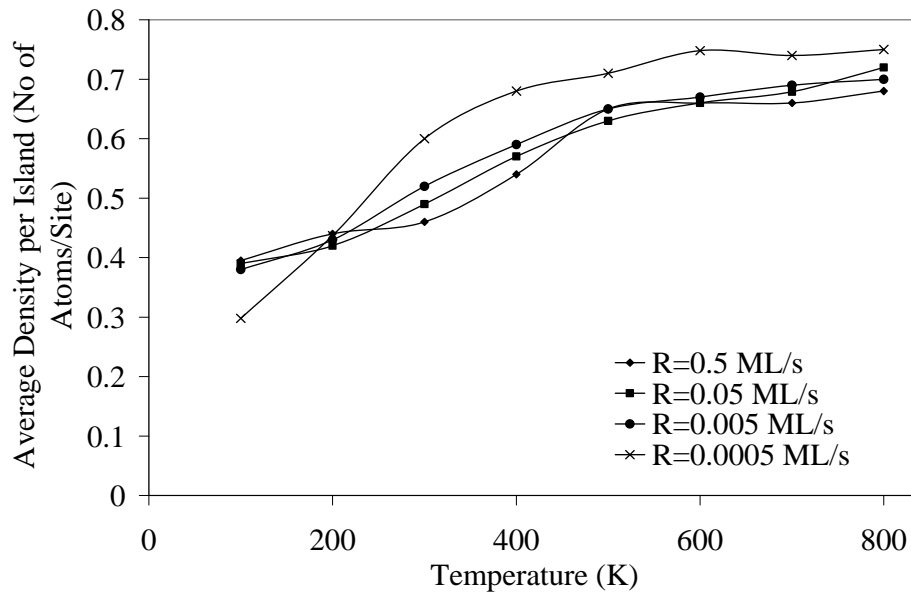
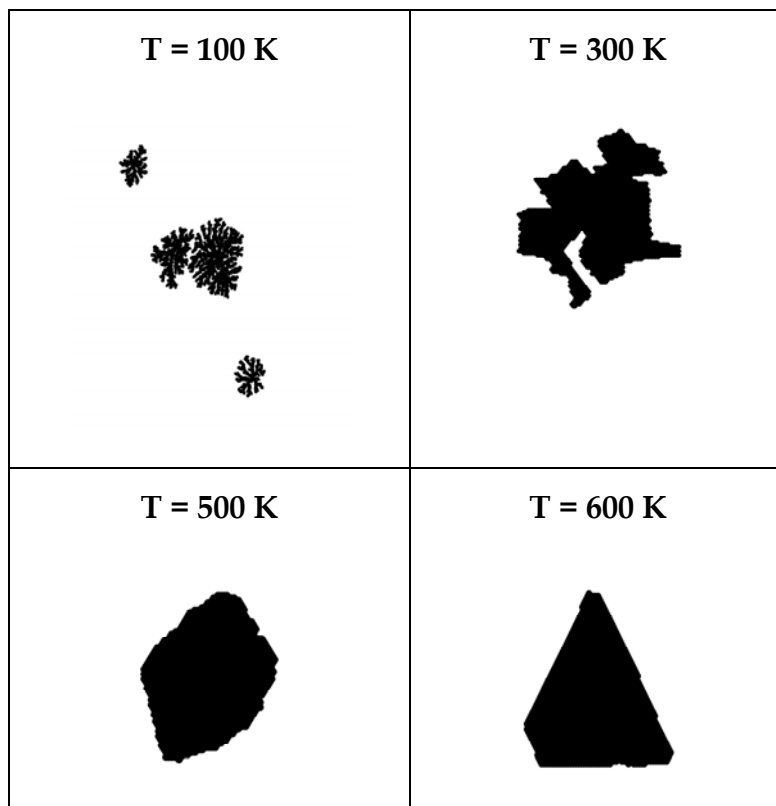


Figure 5.10: Average island density for different temperatures and deposition rates with a coverage of 0.25 ML and 50 nucleation sites.

The scaling relationship of the island growth has also been investigated. This was done to obtain a comparison between the KMC method and rate equations. According to rate equations, a linear relationship exists between the island density and the temperature or deposition rate. The scaling behaviour can be determined by depositing an atom on a substrate without any nucleation sites. From Figure 5.10 it is seen that the number of islands decreases with increasing temperature and deposition rates in a linear fashion. As example, the number of islands' decrease at a given deposition rate is shown in Table 5.5. The scaling behaviour was not investigated for the revised versions of this model.

*Table 5.5: Island morphology for various temperatures at a deposition rate of  $5 \times 10^{-4}$  ML/s and coverage of 0.05 ML with no nucleation sites.*



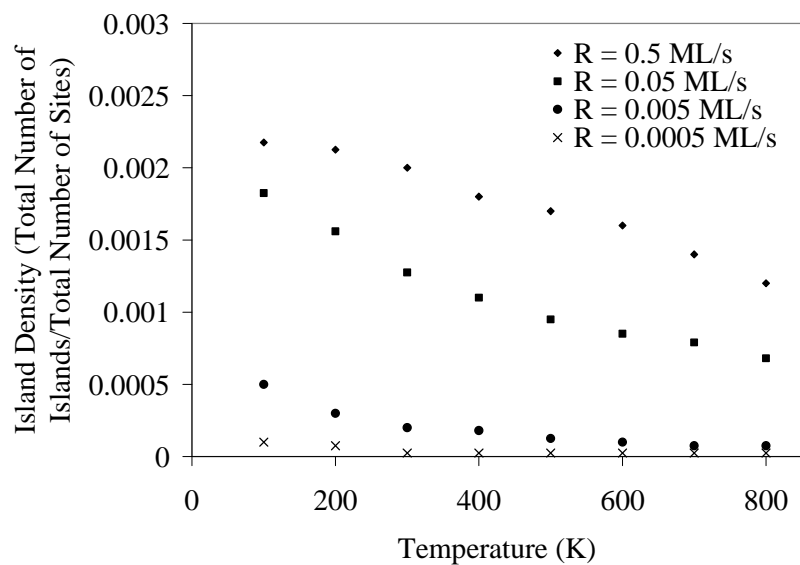


Figure 5.11: Island density for different temperatures and deposition rates with a coverage of 0.05 ML with no nucleation sites.

### 5.2.2.2 Multilayer growth

The temperatures used in the simulation for multilayer growth in Stage 2 were the same as those in Stage 1; namely between 100 K and 600 K respectively. The deposition rates differed however, ranging between  $5 \times 10^{-4}$  ML/s and  $5 \times 10^{-1}$  ML/s. The surface coverage after deposition was 0.4 ML and 25 nucleation sites were randomly placed on the substrate. The following deductions can be made from the simulation (see Table 5.5):

















- Fractal-like behavior is observed for all deposition rates at 100 K. The explanation for this is the same as in Stage 1; namely that the step

diffusion process is not activated at 100 K and atoms merely attach to an island or nanostructure. Although the Schwoebel jump process can make islands less fractal-like, it is not an issue here, since its rate is too slow to occur at  $T = 100$  K (see Figure 5.1).

- Upon increasing temperature, there is an apparent transition from fractal-like to compact islands between 100 K and 300 K at  $5 \times 10^{-4}$  ML/s and  $5 \times 10^{-1}$  ML/s, whereas the islands at  $5 \times 10^{-2}$  ML/s and  $5 \times 10^{-1}$  ML/s are still fractal-like (although the fractal-like arms are thicker, for the same reason discussed in Stage 1). Two processes are responsible for the compact behaviour of the islands namely (i) step-to-corner diffusion and (ii) Schwoebel jumps. Both these processes are activated at 300 K; however, their rates are too slow to occur at  $5 \times 10^{-2}$  ML/s and  $5 \times 10^{-1}$  ML/s, whereas they can occur at  $5 \times 10^{-4}$  ML/s. The Schwoebel jump process does however have a higher probability to occur than the step-to-corner diffusion process due to its lower activation energy and will be the driving force in determining the nanostructure morphology.
- At even higher temperatures, the nanostructures become more compact and the morphology becomes more rounded or faceted. For the lowest deposition rate of  $5 \times 10^{-4}$  ML/s, the nanostructures assume a faceted, triangular morphology at 600 K, whereas it is less faceted, but still slightly triangular at  $5 \times 10^{-3}$  ML/s. At this temperature, the step-to-corner diffusion process, partially responsible for faceting, is fast enough to occur at these two deposition rates. The Schwoebel jump process can also be responsible for faceting, since atoms jumping on top of existing islands or nanostructures leads to mass being transferred away from a particular step. For the two highest deposition rates, the islands are still irregularly compact. Thus, even though the step diffusion process has a

fast enough rate to occur, the step-to-corner diffusion process has a rate which is too slow for it to have a significant probability to occur.

Table 5.6: Nanostructure morphologies for different temperatures and deposition rates and coverages of 0.4 ML. Process selection was done by comparing the weighted probability of each process in the system with a random number in the interval [0, 1]. Only the most recently deposited atom was allowed to move between depositions.

	T = 100K	T = 300K	T = 500K	T = 600K
R = $5 \times 10^{-4}$ ML/s				
R = $5 \times 10^{-3}$ ML/s				
R = $5 \times 10^{-2}$ ML/s				
R = $5 \times 10^{-1}$ ML/s				



### 5.2.2.3 Simulation artifacts

The results presented above have to be interpreted carefully. As with Stage 1, there are simulation artifacts present that skew the results. These artifacts explain the somewhat morphologies of the islands and are as follow:

- *Considering only fcc and  $\alpha$  absorption and jumping sites:* In the case of monolayer growth, this is not expected to have an influence on the island morphology but for multilayer growth this has significant consequences. If jumps are allowed to only fcc-sites, it implies that, whenever an atom is at a B-step (or, if on graphite, a step where the next layer will be a B-step), it is not allowed to jump because the closest site in the next layer will be a hcp site. Therefore, the B-steps will grow out, hence a triangular morphology (see Figure 5.12).

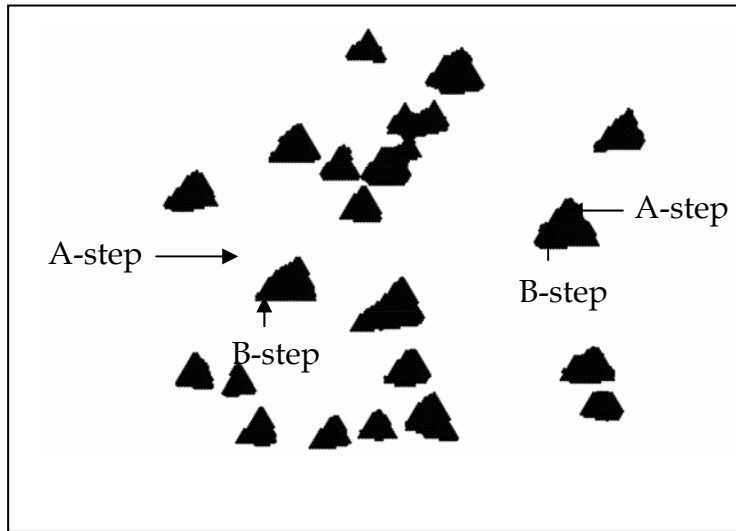


Figure 5.12 : Triangular nanostructure morphologies for a deposition rate of  $5 \times 10^{-4}$  ML/s at 600 K and a coverage of 0.4 ML. The triangular morphology is an artificial effect due to the assumption made in the model that only jumps to fcc sites are allowed.

- *Moving only the most recently deposited atom:* The effects of this assumption in Stage 2 is the same as in Stage 1 and the discussion of Section 5.2.2.1 therefore applies.
- *Neglecting the desorption process:* The same explanation given in Section 5.2.2.1 for Stage 1 is relevant here.

### 5.2.2.4 Stage 3

In light of difficulties encountered in the first two stages, the model was modified to a full diffusion model. In other words, not only the most recently deposited atom is considered mobile between deposition processes, but all of the atoms with the one that can perform the process most probable to occur (by means of a random number comparison as described in Chapter 4) selected. Furthermore, hcp and  $\beta$  sites were included as possible adsorption and jump sites in order to address the simulation artifact that leads to triangular islands, as discussed above. The assumptions made in this stage, in addition to those in Chapter 4, thus include the following:

- Adsorption sites included  $\alpha$ ,  $\beta$ , fcc and hcp sites on graphite and Au (111) respectively, with jumps only allowed to neighbouring  $\alpha$ ,  $\beta$ , fcc or hcp sites. Atoms were therefore able to jump on top of an island at either an A or a B-step. At an A-step, an atom will jump from an fcc or hcp site to a hcp or fcc site whereas from a B-step it will jump from an fcc or hcp site to a fcc or hcp site.
- Following deposition, all atoms in the system were given a probability to move;

- The process most probable to occur was selected through a random number selection process;
- The desorption process was neglected.

Simulations were performed for monolayer and multilayer growth at different deposition rates and temperatures. Deposition occurred at randomly selected sites on a 150 x 150 lattice. At this stage, new values for the energy barriers were used, either obtained from literature or calculated (see Section 5.2). The details of the simulations are given below together with the results.

#### **5.2.2.5 Monolayer growth**

The deposition rates probed in the simulation for monolayer deposition were between 0.01 ML/s and 1 ML/s and temperatures ranged between 100 K and 700 K. A total number of 25 nucleation sites were randomly placed on the substrate, however since this stage uses a full diffusion model, these nucleation sites were allowed to move. The surface coverage after deposition was 0.2 ML. The following deductions can be made from the simulation (see Table 5.7):

- Fractal-like islands are observed at 100 K for all the deposition rates, although the fractal-like islands are much smaller for the rate of 1 ML/s. This is merely because more nucleation sites are created at a higher deposition rate and more, smaller islands are formed.
- An apparent transition from fractal-like to compact islands is made between 100 K and 300 K for 0.01 ML/s and between 500 K and 700 K for

0.1 ML/s and 1 ML/s respectively. The explanation is the same as discussed in the first two stages.

- The compact islands at 700 K and 0.1 ML/s is slightly irregular whereas those at 0.01 ML/s have a faceted morphology (hexagonal as expected, since on graphite there is no "A" or "B" steps and the ratio between the step lengths should become one). The islands at 500 K and 0.01 ML/s is only slightly faceted because the step-to-edge diffusion process responsible for the faceting is activated (see Figure 5.2), but the rate may be too slow for enough of these processes to occur. In other words, even though these processes occur, there is not enough time for the atoms to diffuse along atoms along the island edges to more stable sites and thus attain a more energetically favourable morphology.

#### 5.2.2.6 Multilayer growth

The deposition rates and temperatures used in the simulation for multilayer growth varied between 0.01 ML/s and 1 ML/s and 100 K and 700 K respectively. The 25 random nucleation sites were also allowed to move during the simulation, due to the full diffusion model being employed. The surface coverage after deposition was 0.2 ML and a lattice of 150 x 150 sites was used. The following deductions can be made from the simulation (see Table 5.8):

- The nanostructures at a temperature of 100 K exhibit a fractal-like morphology for all of the deposition rates, with the fractal-like nanostructures much smaller for the rate of 1 ML/s. This is also because more nucleation sites are created at a higher deposition rate than at a

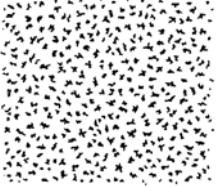


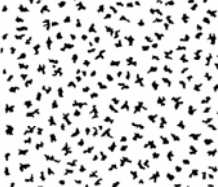





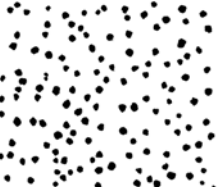
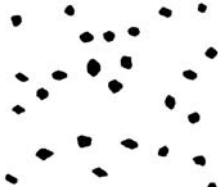

lower one. The density of the nanostructures also increases with decreasing temperature and increasing deposition rate.

- An apparent transition from fractal-like to compact islands is made between 500 K and 700 K for 1 ML/s and between 300 K and 500 K for 0.1 ML/s and 0.01 ML/s respectively. The explanation for this transition is the same as in the Stage 2.
- The compact nanostructures at 700 K and 500 K at 0.1 ML/s have a rounded morphology whereas the nanostructures at 700 K and 1 ML/s have a faceted morphology (hexagonal, quasi-hexagonal and triangular). The interpretation of the faceted behaviour for the multilayer growth should be treated with caution, although most of the explanation given in Stage 2 can be applied here. In particular since it appears that some of the islands have triangles with different orientations. This behaviour is discussed in more depth in the next section.

Table 5.7: Island morphologies for different deposition rates and temperatures (monolayer growth) varied as indicated. The total coverage after deposition was 0.2 ML. Only part of the simulated surfaces is shown.

	R=1 ML/s	R= 0.1 ML/s	R=0.01 ML/s
T=100K			
T=300K			
T=500K			
T=700K			

Table 5.8: Nanostructure morphologies (multilayer growth) for different deposition rates and temperatures varied as indicated. The total coverage after deposition was 0.2 ML. Only part of the simulated surfaces is shown.

	R=1 ML/s	R= 0.1 ML/s	R=0.01 ML/s
T=100K			
T=300K			
T=500K			
T=700K			

### 5.2.2.7 Simulation artifacts

The results presented above have yet again to be treated with caution. As with Stage 1 and 2, simulation artifacts may exist that can distort the results. The artefact that may be present during this stage may be due to neglecting the desorption process. This process is still not expected to have such a great effect on the island morphology; however, it is addressed in the next section.

### 5.2.3 Stage 4

In this stage, the desorption process was included. Furthermore, a simulation artifact not discussed in the previous sections is the size of the intervals over which deposition rates and temperatures are probed. Temperatures of 100 K may be too large to quantify the temperature at which a fractal-like to compact transition occurs. In addition, considering a large substrate at a time limits the possibility of identifying the exact processes responsible for island or nanostructure morphology. Therefore, in this stage, the interest was to investigate the influence of temperature on nanostructure morphology at a given deposition rate. A nucleation site, consisting of a hexagonal island of 7 atoms, was placed on the lattice and atoms were randomly deposited on the lattice. A hexagonal island was chosen to prevent a simulation artifact (for example, a triangular island might lead to triangular growth). A deposition rate of 1 ML/s were used in the simulation, the reason for not considering lower deposition rates is the inefficiency of the KMC method at high temperatures and low deposition rates, discussed in the next section. Different temperatures were probed in an interval of 50 K. The results for varying surface coverages are shown in Table 5.9.



Table 5.9: Nanostructure morphology (multilayer growth) at different temperatures for a deposition rate of 1 ML/s.

	$\theta = 0.01$ ML	$\theta = 0.03$ ML	$\theta = 0.08$ ML	$\theta = 0.1$ ML
T = 100 K				
T = 200 K				
T = 300 K				
T = 400 K				
T = 500 K				
T = 550 K				
T = 600 K				
T = 650 K				
T = 700 K				

From Table 5.9 it can be seen that the nanostructures obtain a more faceted morphology at the higher temperatures. The morphology tend to be triangular, however, an interesting observation is made at 650 K and 700 K. The triangular morphology of the nanostructure is inverted at 700 K compared to the morphology at 650 K. Based on the activation energies, step-to-corner diffusion on an A-step has a higher activation energy than step-to-corner diffusion on a B-step. It is therefore expected that atoms that arrive at corner sites will most likely diffuse towards the B-step. Hence, the B-steps will grow out. This is indeed what is seen at temperatures at 650 K and below. The inverted morphology at 700 K indicates that there is a process which is activated at this temperature that inverts the step-to-corner anisotropy. Indeed, upon closer inspection of the rates calculated, it appeared that the 4→1 process has an activation energy of 0.72 eV and 0.79 eV for the A-step and B-step respectively and the 4→2 process has an activation energy of 0.49 eV and 0.63 eV for the A-step and B-step respectively. This implies a reversal in the step-to-corner anisotropy; the reason for this is not clear at this stage. The effect of the Schwoebel jump processes should not be neglected either, as this barrier can vary according to the number of neighbours surrounding the jumping atom in its initial and final state. Clearly, a detailed analysis of the activation energies is needed before accurate predictions on island morphologies can be made.

The desorption process did not have a significant effect on the results except for slowing down the simulation substantially. This is especially true at higher temperatures since atoms diffusing on a graphite terrace have a very high probability of desorption due to the weak attraction with the substrate.

### 5.3 Comments on the KMC method

At this stage of the study, the method is still not fast enough to simulate systems on experimental time scales exceeding minutes. It is particularly slow at high temperatures (see Figures 5.13 and 5.14). Gold on graphite is also difficult to model using KMC because the rate of diffusion of gold on graphite is very fast ( $\sim 10^{-13}$  s) and a lot of jumps are performed between depositions. Ideally, the code should be optimized and the code parallized.

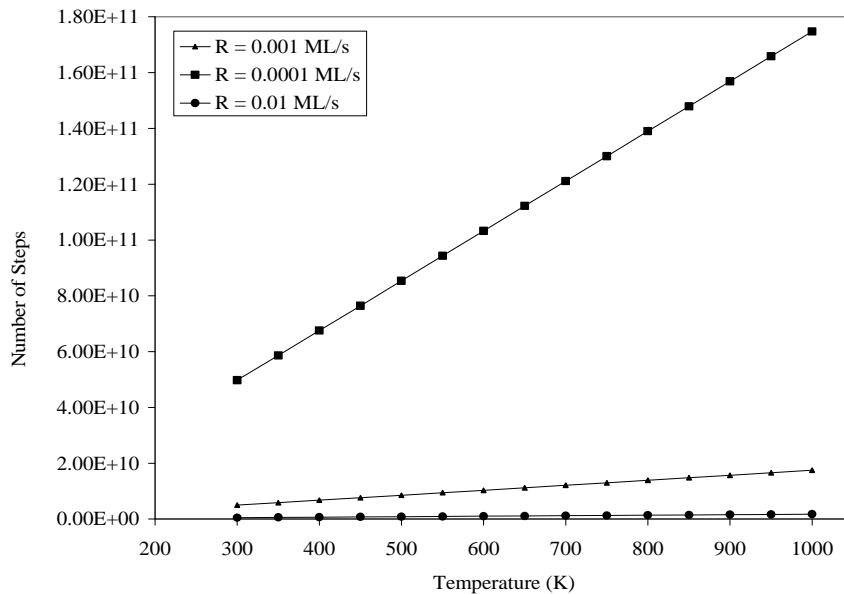
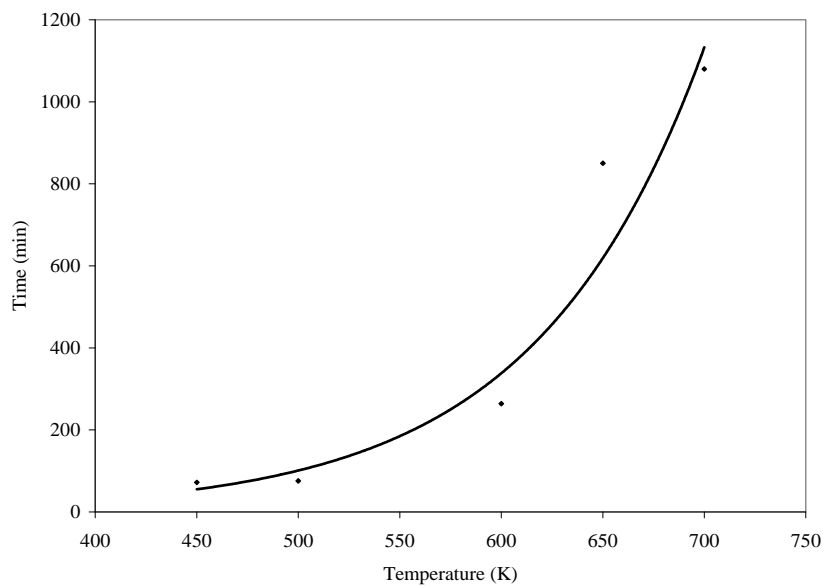


Figure 5.13: The number of steps needed for completion of a simulation (deposition of 1000 atoms in a full diffusion model) depends on the temperature and deposition rate.



*Figure 5.14: Illustration of the exponential increase of computational time with temperature for monolayer deposition using a deposition rate of 0.1 ML/s (deposition of 1000 atoms with a full diffusion model).*

## CHAPTER 6

### Conclusions and Future Work

#### 6.1 Conclusions

In this study, a kinetic Monte Carlo model was developed that is able to perform physically realistic simulations over a useful range of growth conditions. The emphasis of this work was to establish the model such that it can serve as a development platform for other, more advanced models. Indeed, modules were designed that can be used in both other MC and KMC codes. These modules are discussed in Appendix B and C and are for the data structures, the random number generator, the lattice object together with its cells and the main calculation module which contain the functions that perform the KMC method. Many of the functions may be used for MC simulations as well.

The KMC model was developed in different stages, each stage characterised by the assumptions made to describe the system and the dynamics. The assumptions made (in particular those in Stage 1) results in simulation artefacts, as discussed in Chapter 5. However, despite these disadvantages of the model at the various stages, several useful predictions can be made.

In Stage 1, only the most recently deposited atom was allowed to move and the process with the highest probability was selected to occur. Adsorption and jumps were also only allowed onto fcc and  $\alpha$  sites on Au (111) and graphite respectively. The desorption process was also neglected. Although the model at this stage was very crude, it was able to simulate the increase in the number of islands/nanostructures at increasing deposition rates and decreasing temperatures. It could also partially predict a transition from fractal-like to compact islands/nanostructures (and thus increasing density with increasing temperature and decreasing deposition rate as expected from experiment); however, selecting only the process with the largest probability places a question mark on the ability of the model to predict transition temperatures. The model was also not able to correctly predict island/nanostructure morphologies, since at higher temperatures and lower deposition rates; the islands still had an irregular compact morphology.

At Stage 2, the selection of a process to occur during a simulation was based on a random number selection, not by merely considering the one with the largest probability. At this stage, again only the most recently deposited atom was allowed to move. The model was able to obtain an increasing number of islands/nanostructures with increasing deposition rate and decreasing temperature, similar to Stage 1. The size of the islands/nanostructures could also be partially obtained from the simulation. The model was also able to partially predict the transition from fractal-like to compact islands/nanostructures.

Noticeably different from Stage 1 is the ability of the model to attain a faceted morphology of simulated islands/nanostructures. A simulation artefact was present though, with triangular nanostructures dominating due to only fcc sites considered during the simulation.

In Stage 3, a full diffusion model was adopted; however, still only the most recently deposited atom was allowed to move. Jumps to hcp sites were also included. This model was able to deduce the island/nanostructure distribution on the lattice and could attain an increase in the number of islands/nanostructures with an increase in deposition rate and decrease in temperature. The model could also partially predict the transition from fractal-like to compact islands/nanostructures and was able to better describe the simulated morphologies of the islands/nanostructures. The simulation artefact present in Stage 2 that resulted in triangular nanostructures was not a factor here. Instead, hexagonal, quasi-hexagonal and triangular morphologies were obtained.

In Stage 4, the desorption process was included, and the same achievements made by the model in Stage 3 applies here.

The above discussion is summarised in Table 6.1.

It can also be concluded that although the energy barriers used in the simulations were not calculated from first principles, that one can still make useful deductions (as discussed above). In the KMC model, the focus is more on relative process rates and thus accurate enough energy differences. For example,

Table 6.1. Summary of the ability of the KMC model at different stages to simulate a specific property correctly.

Property	Partly Achieved	Achieved
Island/nanostructure size	Stage 1 Stage 2	Stage 3
Island/nanostructure distribution	Stage 1 Stage 2	Stage 3
Island/nanostructure density	Stage 1 Stage 2	Stage 3
Transition temperatures	Stage 1 Stage 2 Stage 3	-
Island/nanostructure morphology	Stage 2 Stage 3	-

having the correct rate ratio between the A and B-steps as input into the KMC model, the correct island/nanostructure morphologies may be obtained, however, the deposition rate and temperature at which this would occur cannot be directly determined. The model and computer code is of course flexible enough to allow improved energy barriers to be used as inputs.

Although the model has a lot of qualitative predictive capability several issues have to be addressed before it will be able to perform industrially relevant simulations. These are discussed in the next section.



## 6.2 Future work

The model itself can be refined by considering the following:

- Optimization of code should be performed to ensure that the KMC method is utilized in the most efficient way. Included in increasing the speed of the simulations is parallelization of the code.
- The artefact arising because of the lapse time should be addressed in an intuitive way. Processes with very slow rates might not occur during a simulation if their residence time exceeds the lapse time. These processes may however have a probability to occur during an experiment. The constraint imposed by the lapse time should thus be addressed.
- Activation energies for all possible processes should be more accurately determined, for example by using DFT methods.
- The relation between the deposition rate,  $R$ , and the vapour pressure,  $p$ , in the system during evaporation should be addressed in the simulation. This relation is given by  $R = p/(2\pi k_B T)$ .

The system studied may also be varied and may include the following:

- Different heteroepitaxial systems: These include not only other metals on graphite, but also other substrates. A change in the lattice geometry can be done in the T\_Lattice object.

- Effect of gases on the growth conditions: Gas molecules, such as CO or O might preferentially adsorb at certain sites on growing particles and limit growth in that direction, therefore altering the morphology of the particles.
- Inclusion of more processes: On Ag(100), exchange diffusion is a process which occurs (however, not for Au (111)) and during deposition, downward funnelling effects etc. can also be included.

## Appendix A

### Derivation of the Master Equation

The derivation of the Master Equation presented here is significantly simplified and a more complete derivation can be found in [71-73]. The derivation starts by integrating equation 3.27 over  $y_2$ . Thus, for  $t_1 < t_2 < t_3$  it follows that

$$P_2(y_1, t_1; y_3, t_3) = P_1(y_1, t_1) \int P_{||}(y_2, t_2; y_1, t_1) P_{||}(y_3, t_3; y_2, t_2) dy_2. \quad (\text{A.1})$$

Dividing both sides by  $P_1(y_1, t_1)$  gives the Chapman-Kolmogorov equation

$$P_{||}(y_3, t_3 | y_1, t_1) = \int P_{||}(y_3, t_3 | y_2, t_2) P_{||}(y_2, t_2 | y_1, t_1) dy_2. \quad (\text{A.2})$$

The Chapman-Kolmogorov equation is an identity that must be obeyed by the transition probability of any Markov process. It states that a process starting at  $t_1$ ,

with value  $y_1$ , reaches  $y_3$  at  $t_3$  via any one of the possible values  $y_2$  at the intermediate time  $t_2$ . For stationary Markov processes the transition probability  $P_{1|1}$  does not depend on two times, but only on the time interval; for this case a special notation is introduced:

$$P_{1|1}(y_2, t_2 | y_1, t_1) = T_\tau(y_2 | y_1), \quad (\text{A.3})$$

with  $\tau = t_2 - t_1$ . A more explicit definition of stationary Markov processes is that their moments are unaffected by time thus

$$\langle Y(t_1 + \tau)Y(t_2 + \tau)\dots Y(t_N + \tau) \rangle = \langle Y(t_1)Y(t_2)\dots Y(t_N) \rangle, \quad (\text{A.4})$$

with  $\langle Y(t) \rangle$  the average of the stochastic process  $Y(t)$ . Therefore, the Chapman-Kolmogorov equation becomes

$$T_{\tau+\tau'}(y_3 | y_1) = \int T_{\tau'}(y_3 | y_2) T_\tau(y_2 | y_1) dy_2. \quad (\text{A.5})$$

The Master Equation can consequently be derived and it is a more convenient version of the Chapman-Kolmogorov equation. It is a differential equation obtained by taking the limit  $\tau' \rightarrow 0$ . For small  $\tau'$  one can write  $T_{\tau'}(y_2 | y_1)$

$$T_{\tau'}(y_2 | y_1) = \delta(y_2 - y_1) + \tau' W(y_2 | y_1) + O(\tau'^2), \quad (\text{A.6})$$

with  $W(y_2 | y_1)$  is the transition probability per unit time from  $y_1$  to  $y_2$ . The delta function expresses that the probability to stay at the same state after time zero is one, whereas the time to change after time zero equals zero. Equation A.6 must be normalised as follows

$$\int T_{\tau'}(y_2 | y_1) dy_2 = 1. \quad (\text{A.7})$$

This will be the case when

$$T_{\tau'}(y_2 | y_1) = (1 - a_0 \tau') \delta(y_2 - y_1) + \tau' W(y_2 | y_1) + O(\tau'^2). \quad (\text{A.8})$$

The coefficient  $(1 - a_0 \tau')$  in front of the delta function is there to correspond to the probability for no transition to take place during  $\tau'$ . The expression for  $T_{\tau'}$  can now be inserted into the Chapman-Kolmogorov equation to give

$$T_{\tau+\tau'}(y_3 | y_1) = [1 - a_0(y_3) \tau'] T_{\tau'}(y_2 | y_1) - W(y_2 | y_3) T_{\tau'}(y_2 | y_1) dy_2. \quad (\text{A.9})$$

Dividing equation A.9 by  $\tau'$  and taking the limit  $\tau' \rightarrow 0$  yields the differential version of the Chapman-Kolmogorov equation :

$$\frac{\partial}{\partial \tau} T_{\tau'}(y_3 | y_1) = \int \{W(y_3 | y_2) T_{\tau'}(y_2 | y_1) - W(y_2 | y_3) T_{\tau'}(y_2 | y_1)\} dy_2. \quad (\text{A.10})$$

This equation is valid for the transition probability of any stationary Markov process obeying equation is called the Master Equation. Rewriting the previous equation and suppressing redundant indices gives

$$\frac{\partial P(y, t)}{\partial \tau} = \int \{W(y | y') P(y', t) - W(y' | y) P(y, t)\} dy', \quad (\text{A.11})$$

which is the customary form of the Master Equation. If the range of  $Y$  is a discrete set of states with labels  $n$ , the equation reduces to

$$\frac{\partial p_n(t)}{\partial t} = \sum_{n'} \{W_{n'n} p_{n'}(t) - W_{n'n} p_n(t)\}, \quad (\text{A.12})$$

which is equation 3.28 given in Chapter 3.

## **Appendix B**

### **Data Structures**

A modular approach was followed in the design of the computer code to simulate vapour deposition. The necessity of modularity for this study is two-fold: (i) most obviously, it is easier to debug smaller programs than larger ones and (ii) a well-written modular program places certain dependencies in a single routine, making changes easier. It also allows flexibility in adding or removing routines by other programmers. In this study, abstract data types (ADT's) were employed to store the calculated rates. ADT's are nothing but a set of operations. They are mathematical abstractions; the method of implementing this set of operations is not specified. They are thus an extension of modular design. The ADT's of choice in this study is the list ADT and the AVL tree. These two ADT's are discussed below.

## B.1 The List ADT

Consider a list of the form  $a_1, a_2, a_3 \dots a_n$ . The size of this list is  $n$  with a list of size zero called a null list. For any list, except the null list, it follows that  $a_{i+1}$  succeeds  $a_i$  and that  $a_{i-1}$  precedes  $a_i$  ( $i > 1$ ). The first element of the list is  $a_1$  and the last element is  $a_n$ . The position of an element  $a_i$  in the list is  $i$ . Associated with the above definitions, there is a set of operations that can be performed, such as insert, remove, make empty and find. All of these instructions can be implemented by using an array. However, this requires that an estimate of the maximum size of the array is specified, even if this array is dynamically allocated. Evidently, this is a serious limitation when storing process rates, especially if there are many unknown rates. Keeping the number of rates fixed is therefore necessary for this type of ADT. Furthermore, the insertion and removal operations are expensive with a scaling behaviour of  $O(N)$ . The list was therefore only used to store the rates of a particular atom for different processes. Thus, for  $N$  atoms, there are  $N$  linear lists. The size of the list ADT is  $M$ , with  $M$  referring to the number of possible sites around a particular atom at which processes can occur. In the system under study (i.e. a hexagonal lattice) this corresponds to 25 – in other words, 12 processes can occur in the same level as the jumping atom, 6 in the level below, 6 in the level above and an additional process is the desorption of an atom. The reason why the list ADT was chosen for the storage of the process rates will be clarified in the next section. The array implementation of the list ADT in C++ (which is the language used in this study) is as straightforward as defining

```
double T_List [24].
```

The elements of a particular linear list are just the rates of the individual processes for the corresponding atom, which are summed to give the total rate of

the corresponding atom. This total rate is then inserted into the tree data structure.

## **B.2. The Tree ADT**

A tree can be defined in several ways. A natural way of defining a tree is by means of recursion. A tree is merely a collection of nodes, with a distinguished node, the root,  $r$ , connecting zero or more sub trees,  $T_1, T_2, \dots, T_k$ , each of whose roots are connected by a directed edge to  $r$ . [97]. The root of each sub tree is termed the child of  $r$  with  $r$  the parent of the child. In a binary tree (see Figure B.1), each root can have only two (or less) children. Nodes that have no children are called the leaf nodes of the tree. The values assigned to each node are called key values (in this study this is the rates of the individual atoms). There are various kinds of trees, like the binary tree. Binary trees are very useful due to their ability to perform efficient data searches. A binary tree can be made into a binary search tree by requiring that for every node of the tree, the values of all the keys in the left sub tree are smaller than the values of all the keys in the right sub tree (see Figure B.2).



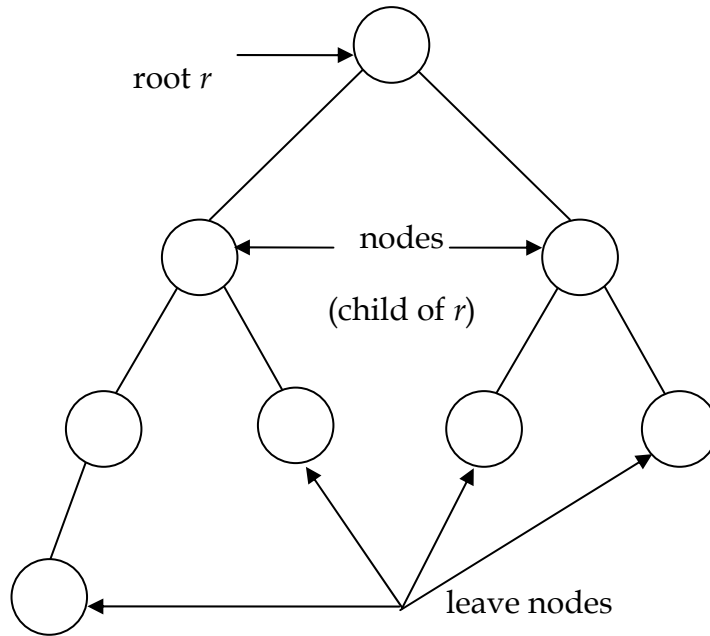


Figure B.1: Schematic illustration of a binary tree. A binary tree does not necessarily have to be balanced.

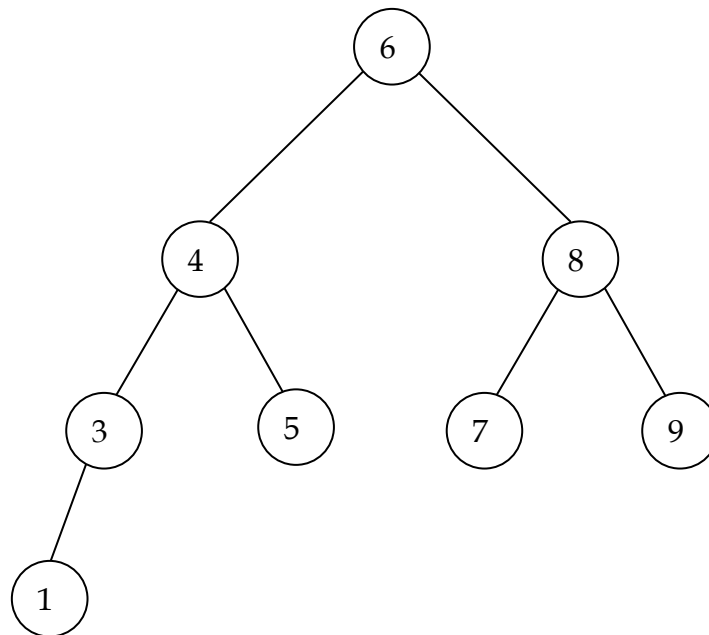
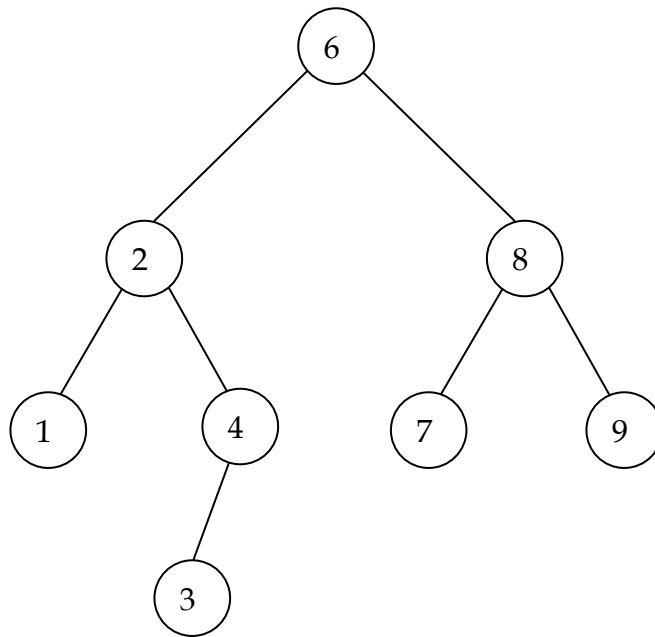


Figure B.2: Schematic illustration of a binary search tree. The key values are as indicated in the figure.

A binary search tree does not necessarily have to be balanced (i.e. have the same height for the left and right sub trees) although it can be. However, with binary search trees it can happen that the tree becomes “too deep” on one side. A search through the tree then becomes equivalent to a search through a linear list. This is especially time-consuming for very deep trees [97]. Typically scaling for binary trees can thus range from  $O(\log N)$  to  $O(N)$ . Imposing a balance condition on the binary search tree can rectify this problem. Such a binary search tree with an added balance condition is the AVL search tree [97]. In this tree, the left and right sub trees have the same height or can differ at most by 1 (see Figure B.3).



*Figure B.3: Schematic illustration of an AVL binary search tree. The key values of the nodes are as indicated in the figure.*

The balancing of the tree is done by rotating the tree around a selected node to ensure that the balance criterion is met. The other operations that were performed on the tree structure were insertions, deletions and finds. Insertions and deletions are done when the rates in the system have to be updated and finds when a most probable rate have to be selected. Implementation of a tree structure is not as trivial as an array. The AVL search tree is treated an object named as T\_Avl\_Search\_Tree. It is derived from the binary search tree, T\_Binary\_Search\_Tree. The nodes of the trees are also treated as objects, T\_Tree\_Node, for the binary search tree and T\_Avl\_Node for the AVL tree. The class structure for the binary search tree nodes contains pointers to the left and right child of each node. The class structure for the binary tree object contains pointers to the root node and the last node that was accessed. The class structure for the AVL search tree nodes contains an extra height parameter whilst the class structure for the AVL search tree contains extra functions to rotate the tree around a specified node to ensure the balance of the tree. These class structures are given in Figures A.4, to A.7 for T\_Tree\_Node, T\_Binary\_Search\_Tree, T\_AVL\_Node and T\_AVL\_Search\_Tree respectively. The class structures make use of templates in C++. This allows one to write a class for an arbitrary type and only to specify that type whenever the object is used. One can thus use floats, doubles, integers etc. as desired.

```
template <class eType>
class T_Tree_Node
{
    protected:
        eType eElement;
        T_Tree_Node *Left;
        T_Tree_Node *Right;
        T_Tree_Node *List;
```

```

T_Tree_Node(eType E = 0, T_Tree_Node *L = NULL, T_Tree_Node
*R = NULL) : eElement(E), Left(L), Right(R) {}
friend class T_Binary_Search_Tree<eType>;
};
//-----

```

Figure B.4: Class structure of T\_Tree\_Node.

```

template <class eType>
class T_Binary_Search_Tree
{
    protected:

        void MakeEmpty(T_Tree_Node <eType>* &T);
        void Insert(const eType &X, T_Tree_Node <eType>* &T, int
Number);
        void Remove(const eType &X, T_Tree_Node <eType>* &T);
        void PrintTree(T_Tree_Node<eType> *T, TCanvas* cCanvas,
double X, double Y, double ParentX, double ParentY, int Width,
bool HLight) const;
        T_Tree_Node<eType>* Copy(const T_Tree_Node<eType>* T);
        T_Tree_Node<eType>* Find(const eType &X, T_Tree_Node
<eType>* T, const eType &Value, T_Tree_Node <eType>*
Value_Pointer) const;
        T_Tree_Node<eType>* FindMin(T_Tree_Node <eType>* T) const;
        T_Tree_Node<eType>* FindMax(T_Tree_Node <eType>* T) const;
        T_Tree_Node<eType>* Root;
        T_Tree_Node<eType>* LastFind;

    public:

        T_Binary_Search_Tree();
        T_Binary_Search_Tree(T_Binary_Search_Tree &Value);
        virtual ~T_Binary_Search_Tree();
        const T_Binary_Search_Tree &operator = (const
T_Binary_Search_Tree &Value);

```

```

    eType operator () () const;
    void MakeEmpty ();
    virtual float Find(const eType &X);
    virtual int FindMin();
    virtual int FindMax();
    virtual void Insert(const eType &X, int Number);
    virtual void Remove(const eType &X);

};
//-----

```

Figure B.5: Class structure of *T\_Binary\_Search\_Tree*.

```

template <class eType>
class T_Avl_Node : public T_Tree_Node<eType>
{
    private:
        int Height;
        friend int Node_Ht(T_Avl_Node *P)
        {return P ? P->Height : -1;}
        friend int Node_Ht(T_Tree_Node <eType> *P)
        {return Node_Ht((T_Avl_Node<eType>*)P);}
        friend void Calculate_Height(T_Avl_Node *P)
        {P->Height=1+max(Node_Ht(P->Left),Node_Ht(P->Right));}

    protected:
        friend class Avl_Search_Tree<eType>;

    public:
        Avl_Node(eType E = 0, Avl_Node *L = NULL, T_Avl_Node *R =
        NULL, int H = 0) : T_Tree_Node<eType>(E,L,R), Height(H) {}
};
//-----

```

Figure B.6: Class structure of *T\_Avl\_Node*.

```

template <class eType>
class T_Avl_Search_Tree : public T_Binary_Search_Tree <eType>
{
    protected:
        T_Avl_Node <eType> *Copy(const T_Avl_Node<eType>*T);
        virtual      void Insert(const eType & X, T_Avl_Node<eType>*
        &T, int i);
        virtual      void      Remove(const      eType      &      X,
        T_Avl_Node<eType>* &T);

    public:
        T_Avl_Tree():T_Binary_Search_Tree <eType> () {}
        const T_Avl_Tree & operator = (const T_Avl_Tree & Value);
        virtual void Insert(const eType & X, int i)
        {Insert(X, (T_Avl_Node<eType> *&) Root, i);}
        virtual      void Remove(const eType &X)
        {Remove(X, (T_Avl_Node<eType> *&) Root);}
        void S_Rotate_Left(T_Avl_Node<eType>* & k2);
        void D_Rotate_Left(T_Avl_Node<eType> * & k3);
        void S_Rotate_Right(T_Avl_Node<eType>* & k1);
        void D_Rotate_Right( T_Avl_Node<eType> * & k3);
};
//-----

```

Figure B.7: Class structure of T\_Avl\_Search\_Tree.

The details on how classes are written, the role of constructors, destructors, virtual functions and so forth can be found in [97]. Implementations of the functions in the above mentioned classes will be supplied upon request to the author [105].

In order to use the AVL search tree, an instance of the T\_AVL\_Search tree has to be created first

```
T_Avl_Search_Tree <double> T_Atom;
```

T\_Atom is now the object which will be used. The term “double” refers to the data type that will be used during calculations. Insertions of rates are then done recursively by calling the function Insert

```
T_Atom->Insert (rate);
```

recursive deletions by

```
T_Atom->Remove (rate)
```

and recursive finds by

```
T_Atom->Find (rate).
```

“Rate” refers to the total rate of each atom that is either inserted, deleted or has to be found. Each node in the tree is also linked, via the pointer “List” indicated in Figure B.5, to a list that contains the rates of the various jump directions of each atom. Insertions are done recursively by comparing the value to be inserted, X, with the key values of each node in the tree, starting from the root. If X is smaller than the key value of a node, tree traversal occurs to the left child to the node else to the right child. This procedure is recursively repeated until a leaf node is reached. The same situation applies to removals. For the AVL tree, however, each insertion or removal is accompanied by a rotation of the tree to ensure that the tree remains balanced. The selection of the atom most probable to jump at a given time in the system is done by calling a find, which again, is done by

recursively searching through the tree, as with the case of the insertion and removal. The other class structures used are discussed in Appendix C.



## APPENDIX C

### Class Structures

In conjunction with the `AVL_Search_Tree`, `AVL_Search_Tree` nodes (and corresponding `Binary_Search_Tree` and `Binary_Search_Tree_Nodes`) and linear list class structures (`T_List`), structures also exist for the lattice, cells of the lattice, random number, calculation functions and for the main program. These class structures are clarified as follows:

- `T_Cell`: This is the structure for the cells of the lattice (i.e a struct). The `T_Cell` object contains information about the height of the cell, its occupancy, the type of atom that occupies it, its row and column numbers, its coordinates, its number and lastly whether it is a bulk atom or not.
- `T_Lattice`: This is the class structure for the lattice and contains all the `T_Cells` and is shown in Figure C.1.
- `T_Calculation_Engine`: This is the class structure which contains all the member functions and methods to perform the calculations (i.e simulate

vapour deposition). The T\_Calculation\_Engine class, with some of the functions and methods, is shown in Figure C.2.

- T\_Rand: This is the class structure used for the Mersenne Twister random number generator. The class structure is shown in Figure C.3.
- T\_Main: This is the main class structure (representing the main program).

The relationship between the various class structures is shown in Figure C.4.

```

class T_Lattice
{
    public:
        TLattice (int NX, int NY, int NZ); //constructor
        virtual __fastcall ~TLattice (void); //destructor
        double __fastcall Get_X_Coordinate(int i, int j, int k); // for lattice
        double __fastcall Get_Y_Coordinate(int i, int j, int k); //for lattice
        double __fastcall Get_Z_Coordinate(int i, int j, int k); //for lattice

    struct Cells
    {
        double X_Coordinate;
        double Y_Coordinate;
        double Z_Coordinate;
        bool Occupied;
        int Cell_No;
        int Neighbours;
        int Height;
        int i_I;
        int i_J;
        bool Bulk;
        int Atom_No;
    };
}; //T_Lattice
//-----

```

Figure C.1: Class structure for the T\_Lattice Object. Comments are given in italics.

```

class T_Calculation_Engine
{
    public:
        T_Calculation_Engine(int No_Atoms, double Rate, double Temp, int
        Size_X, int Size_Y, int Size_Z); //constructor
        ~ T_Calculation_Engine(void); //destructor
        void __fastcall SaveCells(AnsiString Filename); //save existing
        configuration
        void __fastcall ReadCells(AnsiString Filename); //load in a new
        configuration
        void SleepLocal(int Milliseconds); //used to emulate doevents of Visual
        Basic
        void DoEvents(); //used to emulate doevents of Visual Basic
        int __fastcall GetSCPParameters (String Element); //get the Sutton-Chen
        potential parameters for gold
        int __fastcall Random_Number(double U_X, double L_X, double
        U_Y, double L_Y); //generate a random number
        int __fastcall GetActParameters (String Element); //get the activation
        energy barriers for a specific element
        int __fastcall Get_Process(int N_i, int N_f, int No_Init, int No_Final,
        int Step_Type); //decide which process occurs in the system
        void __fastcall Get_NB(int i_No, int k, int Ineraction, int Level); //get
        the neighbours of a particular atom
        int __fastcall MoveAtom(String Element, int i_Prob); //calculate the
        rates of the atoms and which one should move
        int __fastcall MoveDirection(String Element, int SelectedAtom); //based
        on the selected atom, calculate in which direction it should move
        void __fastcall Distance(int i, int j, int k, double X, double Y, double
        Z, int k1); //calculate the distance between atoms – used for counting the
        neighbours
        double __fastcall Total_Energy (int No); //calculate the total energy for a
        specific atom in the system – includes all its neighbours
        double __fastcall Level_Energy(double r, int Neighbours); //calculate
        the energy for a specific atom in the system for one set of neighbours
        double __fastcall Get_Barriers(int i, int Step_Type, int N_i, int N_f);
        //get the activation energy barriers depending on whether on A, B step or on
        graphite
        double __fastcall Get_Activation_Energy (int i, int Step_Type, int
        No_i, int No_f, int s, int N_i, int N_f); //calculate the activation energy for
        a specific process – depends on the activation energy barrier and initial and
        final energy of the particular atom.

```

```

double __fastcall Get_Rate(int i, double Act_E, double Sim_Temp);
//calculate the rate given the activation energy
double __fastcall Delta_Energy(int No_i, int No_f, int N_i, int N_f);
//calculate the energy difference between the initial and final positions of an
atom
void __fastcall All_Jump_Rates(int No, int NoAtom); //function to put
all the rates in a list - contains the functions Get_Activation_Energy,
Get_Rate, Get_Process
void __fastcall Get_Probabilities(int SelectedAtom); //put all the rates in
lists
void __fastcall Get_ProbabilitiesAtoms(); //put all the rates for the atoms
in the AVL_Tree
int __fastcall Select_Probability(); //select which process should be
performed ie in which direction an atom should move
int __fastcall Select_ProbabilityAtoms(); //select which atom should move
int __fastcall Get_Step(int Init, int Final); //determine whether on A or B
step
void __fastcall Save(AnsiString Element_Type, AnsiString Filename);
double __fastcall Absolute(double Number);
void __fastcall Save_Desorp(AnsiString Filename);
int __fastcall Move(double X_Coordinate, double Y_Coordinate,
double Z_Coordinate, double Step_Size, double U_X, double L_X,
double U_Y, double L_Y, double U_Z, double L_Z, double theta);
int __fastcall Find_Nearest_Cell(double X2, double Y2, double Z2,
double U_X, double L_X, double U_Y, double L_Y, double U_Z,
double L_Z, double theta);
void __fastcall SaveMSD(AnsiString Filename);
void __fastcall Update_Rates(String Element, double U_X, double
L_X, double U_Y, double L_Y, int i_Prob) ; //update the rates in the
vicinity of the moving atom
T_Rand drand; // double in [0, 1) generator, already init
Avl_Search_Tree<double> L_Atom; //create AVL_Tree object
}; //T_Calculation_Engine
//-----

```

Figure C.2: Class structure for the T\_Calculation\_Engine object. Comments are given in italics.

```

class T_Rand
{
    public:
        // default constructor: uses default seed only if this is the first instance
        MTRand() { if (!init) seed(5489UL); init = true; }
        // constructor with 32 bit int as seed
        MTRand(unsigned long s) { seed(s); init = true; }
        // constructor with array of size 32 bit ints as seed
        MTRand(const unsigned long* array, int size) { seed(array,
size); init=true;}
        // the two seed functions
        void seed(unsigned long); // seed with 32 bit integer
        void seed(const unsigned long*, int size); // seed with array
        unsigned long operator()() { return rand (); }
        ~MTRand() {} // destructor
    protected:
        unsigned long rand (); // generate 32 bit random integer
    private:
        static const int n = 624, m = 397; // compile time constants
        static unsigned long state[n]; // state vector array
        static int p; // position in state array
        static bool init; // true if init function is called
        // private functions used to generate the pseudo random numbers
        unsigned long twiddle(unsigned long, unsigned long); // used
by gen_state()
        void gen_state(); // generate new state
        MTRand(const MTRand&); // copy constructor not defined
        void operator=(const MTRand&);

}; //T_Rand
//-----

```

Figure C.3: Class structure for the T\_Rand object. Comments are given in italics.

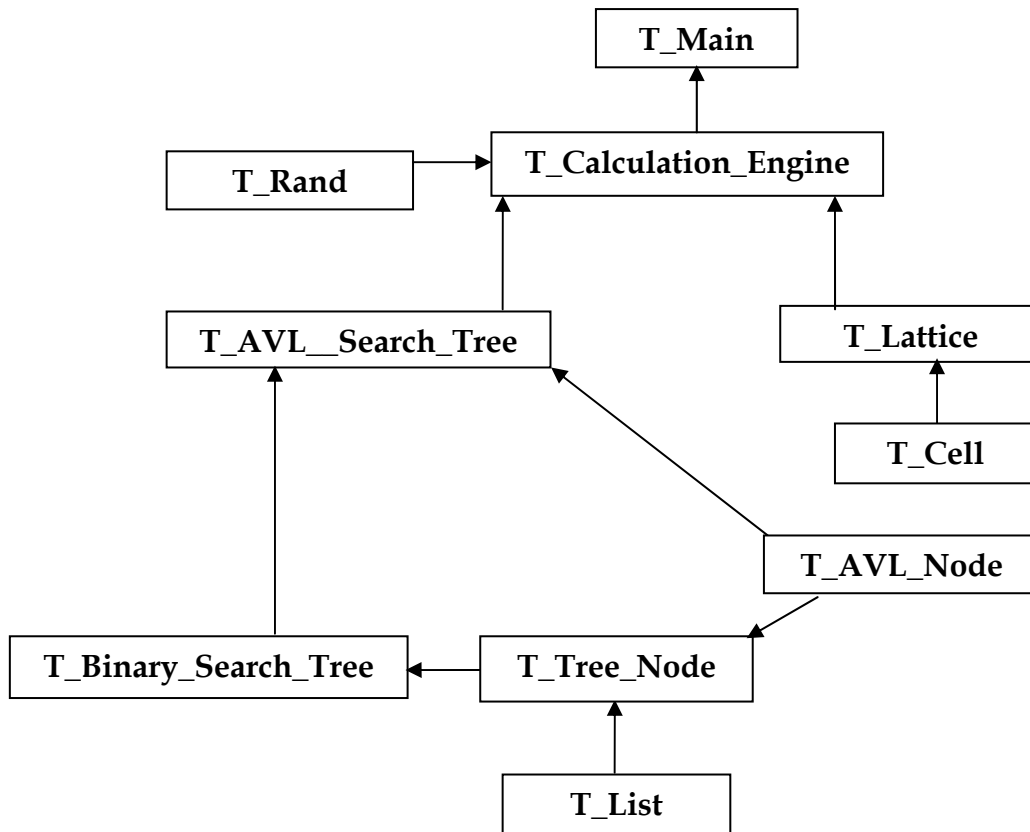


Figure C.4: Class structure of the objects used in the KMC program. The class `T_Main` is the object associated with the main program. The lattice is constructed in the class `T_Lattice`, with a `T_Cell` struct linked to the lattice. For an  $L \times L$  lattice there are  $L \times L$  `T_Cell` objects. The rates of each atom are stored in the nodes, `T_AVL_Node`, of the `T_AVL_Search_Tree` structure. The `T_AVL_Node` is derived from the `T_Tree_Node` of the `T_Binary_Search_Tree` object whereas the `T_AVL_Search_Tree` is derived from the `T_Binary_Search_Tree` object. The rates for each jump direction around an atom in the `T_AVL_Search_Tree` are stored in a list, `T_List`. A jumping atom and direction is selected by spinning a random number, generated in the `T_Rand` class structure, and

*comparing this random number with the weighted probabilities of the respective rates. The calculation and update of the rates, saving of data, reading of activation energies and so forth are done within the class T\_Calculation\_Engine. The results are output to the main program.*

## **Appendix D**

### **Random Numbers**

All stochastic simulation methods, such as the KMC method implemented in this study, are based on the use of noise produced by random number generators. These random number generators are of prominent importance since they are employed to generate the dynamics of the system under study. Random number generators are deterministic in nature, therefore, if there is any significant correlations within the random number sequence, the dynamics of the system under study will be biased and the results may become questionable. Random number generators should therefore be subjected to rigorous testing before being used in stochastic simulations.



Essentially, random numbers can be identified by the fact that their values are unpredictable. In other words, if a sequence of random number is constructed, then the probability distribution of the following random numbers has to be completely independent of all the other generated numbers. Three types of random numbers can be distinguished. These are:

- True Random Numbers: True random numbers are most often gained from physical processes such as radioactive decay, thermal noise in semiconductors and the like. It is important to ensure that, when constructing a random number generator based on a physical device, that merely “random” output is insufficient; the numbers generated must be to a large extent representative of independent random variables. This implies that a device that generates a stream of bits, each bit should be either 0 or 1 with equal probability. The bit should also be independent of all the other generated bits. Using devices for the generation of random numbers is risky for a couple of reasons. These include (a) the difficulty to install and run them; (b) cost in acquiring them; (c) slow speed; (d) inability to reproduce a sequence exactly. The latter is especially important since it is necessary to verify and debug simulation code or to compare it to similar systems. The usefulness of these random numbers is that they can be used as seeds for algorithms designed to be random number generators.
- Uniform Random Numbers: These random numbers are uniformly distributed. The aim of such numbers is, for example, to control and

reduce the errors in Monte Carlo simulations. Such uniform distributions are typically generated by predefined computer functions or random number generators (RNG's). Although there are a variety of random number generators, none of them are truly uniform, rather they generate a sequence of numbers that has a very long repeat periodicity and small correlation between numbers within one period. The probability distribution for a uniform random number on the interval (0, 1] is:

$$p(x) = 1 \text{ if } 0 \leq x \leq 1 \\ = 0 \text{ if } 1 < x < \infty$$

which means that the probability of getting a number in the interval between  $x$  and  $x + dx$  is equal to 1 when  $x$  is between 0 and 1 and zero otherwise.

- Pseudo Random Numbers: These numbers are generated by a computer or algorithm (RNG) and because of this they are not truly random. Every new number is generated from the previous ones by an algorithm. This means that the new value is fully determined by the previous ones. But, depending on the algorithm, they often have properties making them very suitable for simulations [100].

A comparison between uniform and pseudo random numbers is made in Figure D.1 and D.2. It is evident that, for the purpose of this work, a pseudo random number generator will have to be used to prevent repetitions in the process selection.

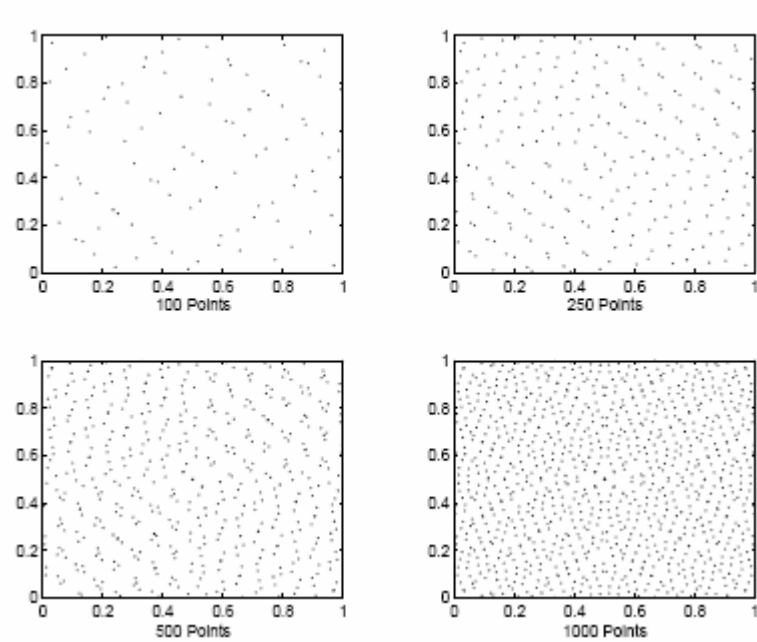


Figure D.1: An illustration of random numbers generated with a uniform random number generator.

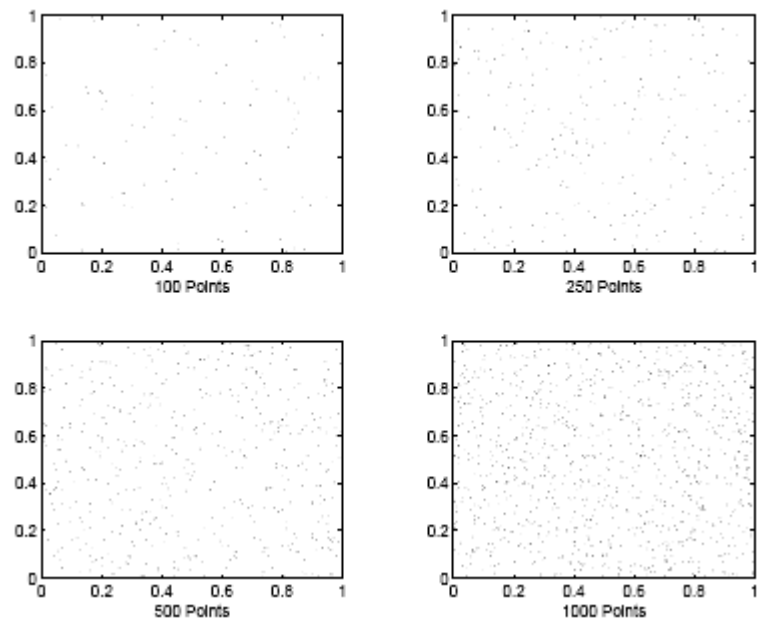


Figure D.2: An illustration of random numbers generated with a pseudo random number generator.

## D.1 The Mersenne Twister RNG

The criteria that should be met for a RNG to be classified as a “good” RNG are as follow [100]:

- It must be able to generate random numbers fast;
- It must be portable between various platforms;
- It must have a long period of repeat;
- It must have repeatable results;
- It must have a uniform distribution over the set  $(0, 1]$ .

Although most computer languages have built-in RNG's, these typically have limited period and are not truly random [100]. Consequently, one of the most advanced RNG's currently available, namely the Mersenne Twister, has been implemented to generate random numbers for the simulations. A comparison between various RNG's, including the Mersenne Twister, can be found in [100]. In short, the Mersenne Twister is a generalized feedback shift register and has a period of  $2^{19937}-1$ , a 623-dimensional equidistribution, accuracy to 32 bits and has a working area of 624 words. It thus allows for a long period and efficient use of memory. It is faster in comparison to most generators and has passed rigorous testing [100]. This RNG was specifically designed for Monte Carlo simulations and is widely used as the RNG of choice in the statistical community. The mathematics and implementation behind the Mersenne Twister algorithm is quite complex, and are therefore not covered here. A detailed explanation can be found in [100].

## **Appendix E**

### **Calculation of Activation Energy Barriers with Molecular Dynamics**

The accuracy of the results obtained via KMC simulations is influenced by the accuracy of the activation energy barriers for the various processes in the system. The activation energy barriers are needed to calculate the rates of the processes, as seen in equation 3.48. Although it is possible to get a rough estimate of the influence of various processes on island morphology, provided that the relationship between the various barriers is correct, comparison with experimental results is difficult. Ideally, first principle calculations are preferred for the determination of the activation energy barriers. On both Au(111) and

graphite; however, various DFT calculations were unable to correctly account for the gold-graphite interaction [101]. On the other hand, molecular dynamics studies on the gold-graphite system is quite popular and a modified Lennard-Jones potential [102, 103] has been extensively used, with much success, in these calculations. Unfortunately, no literature is currently available on the activation energy barriers for diffusion of a gold adatom around a gold island on graphite and very little for the diffusion of a gold adatom on bare graphite surfaces. Consequently, an energy-minimization routine was written, employing the modified Lennard-Jones potential, to calculate these barriers. In contrast to the sparsity of information on diffusion of adatoms on graphite, diffusion on and along steps on fcc(111) surfaces, and on Au(111) is quite well-studied. As such, activation energy barriers for step and terrace diffusion were obtained from the literature, calculated with molecular dynamics using a many-body RGL potential [104]. In the next section, a brief discussion is thus given on the methodology to calculate the diffusion barriers on graphite and on Au(111).

### **E.1. Calculation of the activation energy barriers on graphite**

The system used to calculate the activation energy barriers for the edge diffusion of a gold adatom along a gold island on graphite, consisted of two sheets of graphite, with 120 rows and 110 columns. A gold island of 50 rows was placed on the centre of the graphite sheet. Figure E.1 shows a schematic illustration of a gold island on the graphite substrate. For clarity, only one graphite layer is shown and the number of columns and rows has been reduced. The interaction between the gold atoms was described by the Sutton-Chen potential [106] since this potential can be readily implemented and gives reasonable results with experiment [107].

The interaction between the gold and graphite atoms was described by a modified Lennard-Jones potential [108] whereas the carbon atoms' interactions are described by the Tersoff-Brenner potential which is well-suited for carbon based systems [109]. The activation energy for step diffusion was calculated as the energy difference between the initial and transition state position of the diffusing adatom. At each of these two positions the energy of the system was minimized using an energy minimization routine. This entailed constraining the  $x$ -coordinate of the adatom and allowing it to relax in the  $y$  and  $z$  direction. At each step of the relaxation, the energy of the system was calculated. The calculated energy after relaxation was compared with that before relaxation. If it was less, relaxation in a particular direction was allowed, else not. This procedure was repeated for all the atoms within a specified interaction area, taken to be 10 Å. The code used for this procedure is available from ref. [105]. Following this methodology, the activation energy barrier for diffusion along a step was found to be 0.154 eV. Calculation of the diffusion energy barrier of a single gold adatom on the graphite surface as well as the Schoewbel barrier was done following the same method as above. It was found to be 0.005 eV and 0.126 eV respectively.

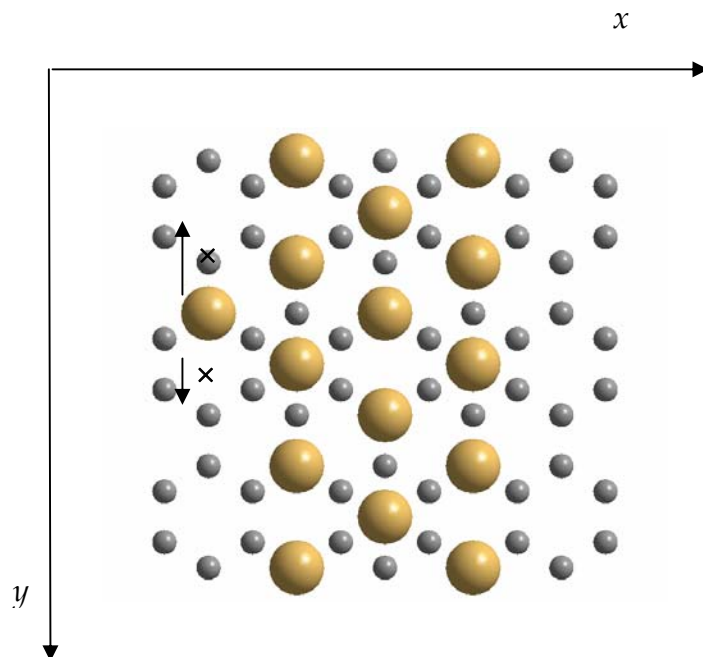


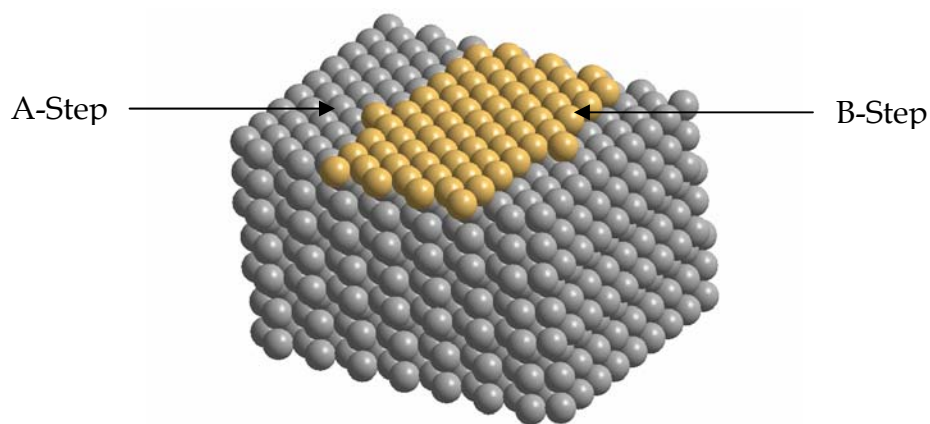
Figure E.1: Schematic illustration of the graphite layer and gold atoms used for determining the activation energies along the island edges. For clarity, the second layer of graphite is not shown. The steps on either side of the island are the same. The transition states for the diffusing atom are indicated by crosses.

## E.2. Calculation of the activation energy barriers on Au(111)

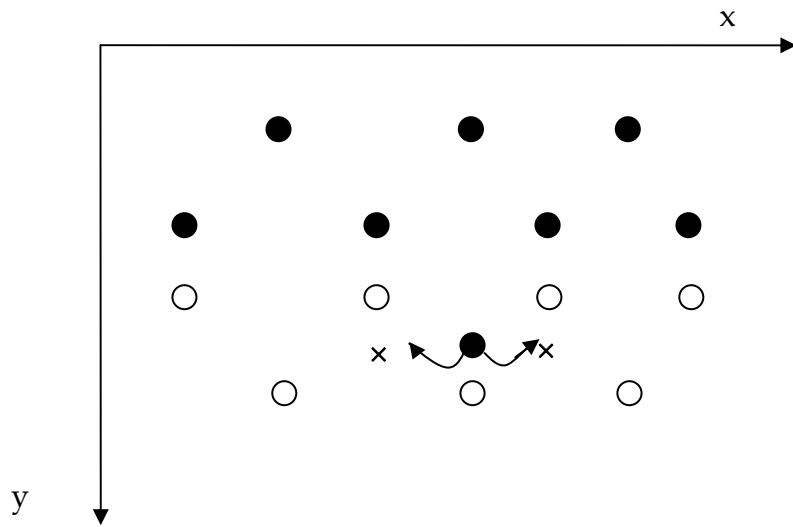
The values that are presented here were obtained from Ferrando et al [104] who performed an extensive molecular dynamics study of the anisotropic diffusion along the (111) steps of gold and silver. Therefore, only the aspects of the methodology followed in the calculations relevant to this study, are addressed here. The system used to investigate the dynamics of a gold adatom along the (111) steps of gold were described by a (111) unreconstructed slab with 9 layers in the z-direction, 9 rows (x-direction) and 15 columns (y-direction). An island on the slab was represented by a terrace of 7 columns, delimited by an A-step and B-



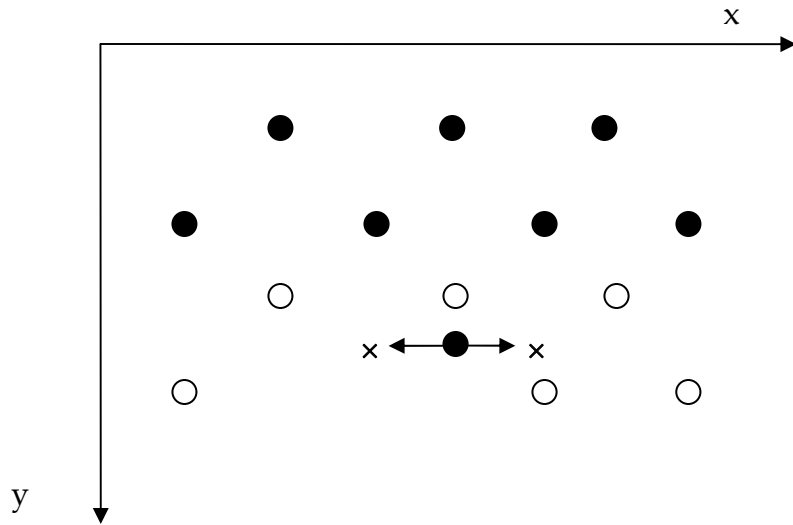
step on the left and right hand side respectively (as shown in figure E.1). An adatom was placed alongside either step. The minimization of the energy at 0 K for this structure was done following a quenching procedure [104]. The calculation of the energy barriers was done by placing the adatom on the transition state (for that particular step) and performing the quenching mentioned above. Figure E.2 shows the diffusion paths along the two steps on the Au (111) surface, as well as the transition states. From these simulations it was found that the activation energy barrier for diffusion of a single adatom along an A-step is 0.34 eV and along a B-Step is 0.22 eV. The Schwoebel barrier was however not determined in this study [104] and no values could be found in the literature either. Thus, using the same simulation cell as shown in Figure E.1, the Schoewbel barrier was calculated in the same fashion as discussed in section E.1 yielding a value of 0.504 eV. As a comparison, the diffusion barriers along the A step and B step were also calculated and found to be 0.302 eV and 0.27 eV. Although the values are close to those determined in [104], the difference between A-step and B-step diffusion is not that prominent.



*Figure E.2: The slab used in the molecular dynamics simulations [104]. The terrace on the topmost layer is bounded by an A-step and B-step on the left and right hand sides respectively. Also shown is adatoms on both steps.*



(a)



(b)

Figure E.3: Schematic illustration of the diffusive paths along the (a) A step and (b) B step on Au(111). The crosses represent the transition states. Open circles denote the atoms in the lower layer whereas the black circles denote atoms in the top layer.

# Appendix F

## Supplementary Code

In this appendix, extracts from the programs written in this study is given. The following programs were written:

- *pDiffusion*: This program calculated the activation energy barriers for diffusion along the different steps of a Au island on Au(111) and graphite as well as the activation energy barrier for diffusion of a Au atom on a graphite terrace. The program was written in Visual Basic 6.0.
- *pKMC*: This program is the main program performing the KMC simulation. The program was written in C++.
- *pDisplay*: This program utilizes OpenGL routines to display surface morphology as determined by KMC. The program was written in Visual Basic 6.0.

- *pIsland*: This program calculates island sizes and densities. The program was written in Visual Basic 6.0.

In the next three sections, some of the code for the pDiffusion, pDisplay and pIsland programs are given. The pKMC code was discussed in Appendix . Further information can be obtained from [].

## F.1 pDiffusion

Option Explicit

```
Type AtomVel
  X As Double
  Y As Double
  Z As Double
End Type
```

```
Type AtomForce
  X As Double
  Y As Double
  Z As Double
End Type
```

```
Type Atoms
  X As Double
  Y As Double
  Z As Double
End Type
```

```
Global nn As Integer
Global NoAtomsC As Integer
Global mass_Au As Double
Global E_Au As Double
Global AtomForAu() As AtomForce
Global AtomForC() As AtomForce
Global Energy(1000) As Double
```

Global Nx As Double  
Global Ny As Double  
Global Nz As Double  
Global No\_Substrate As Double  
Global AtomPosC() As Atoms  
Global AtomPosAu() As Atoms  
Global f As Double  
Dim fstart As Double  
Global Loc\_Dens() As Double  
Dim Local\_Density\_Sub() As Double  
Dim Local\_Density\_Surf() As Double  
Dim a As Double  
Dim alfa As Double  
Dim D\_t As Double  
Dim D\_e As Double  
Dim r\_e As Double  
Dim m\_t As Double  
Dim Rb As Double  
Dim E As Double  
Dim c As Double  
Dim n As Double  
Dim m As Double  
Dim N\_N As Single  
Dim A\_t As Double  
Dim B\_t As Double  
Dim l\_mu As Double  
Dim mu As Double  
Dim beta As Double  
Dim n\_t As Double  
Dim c\_t As Double  
Dim d As Double  
Dim h As Double  
Dim R\_t As Double  
Dim S\_t As Double  
Global No\_Atoms As Double  
Dim psi As Integer

---

Public Function SC\_Ui(i As Double, N\_Atoms As Double)

```

Dim j As Double
Dim u As Double
Dim S1 As Double
Dim Vr As Double
Dim Sum_Vr As Double
Dim r_ij As Double
Dim p_i As Double
Dim s As Single
Dim a_op_r_ij As Double
Dim rx_ij As Double
Dim ry_ij As Double
Dim rz_ij As Double
Sum_Vr = 0
p_i = 0

For j = 1 To N_Atoms
    If (AtomPosAu(j).X <> 0 And AtomPosAu(j).Y <> 0 And AtomPosAu(j).Z
<> 0) Then
        If j <> i Then
            rx_ij = (rx(i, j, "Au", "Au"))
            ry_ij = (ry(i, j, "Au", "Au"))
            rz_ij = (rz(i, j, "Au", "Au"))
            r_ij = (rx_ij ^ 2 + ry_ij ^ 2 + rz_ij ^ 2) ^ 0.5
            a_op_r_ij = a / r_ij
            p_i = p_i + s * (a_op_r_ij) ^ m
            Sum_Vr = Sum_Vr + s * (a_op_r_ij) ^ n
        End If
    End If
Next j
SC_Ui = 0.5 * Sum_Vr - c * (p_i ^ 0.5)

End Function

```

---

```

Public Function SC_Crystal_Energy(No_Atoms As Double, Element As String)

```

```

Dim Sum_U As Double
Dim i As Double
Sum_U = 0

```

```

For i = 1 To No_Atoms
  If (AtomPosAu(i).X <> 0 And AtomPosAu(i).Y <> 0 And AtomPosAu(i).Z
<> 0) Then
    Sum_U = Sum_U + SC_Ui(i, No_Atoms)
  End If
Next i
SC_Crystal_Energy = E * Sum_U

End Function

```

---

```

Public Function Forces(NoAtoms As Double, j As Double, AtomType As String)

```

```

  Dim rx_ij As Double
  Dim ry_ij As Double
  Dim rz_ij As Double
  Dim i As Double
  Dim k As Double
  Dim fx As Double
  Dim fy As Double
  Dim fz As Double
  Dim a_on_r As Double
  Dim r As Double
  Dim RCutOff As Double
  Dim Diff As Double
  Dim p_n As Double
  Dim p_m As Double
  Dim Inv_R As Double
  Dim p As Double
  Dim f As Double
  Dim Loc_Dens(10001) As Double
  Dim Inv_Loc_Dens_j As Double
  Dim Inv_Loc_Dens_i As Double
  Dim Sum_Inv_Loc_Dens As Double
  Dim cm_Sum_Inv_Loc_Dens As Double
  Dim cm_Sum_p_m As Double
  Dim np_n As Double
  Dim Rx_on_R As Double
  Dim Ry_on_R As Double
  Dim Rz_on_R As Double

```

```

RCutOff = 10
Dim cmop2 As Double
Dim R_x_on_R As Double
Dim R_y_on_R As Double
Dim R_z_on_R As Double

f = 0
fx = 0
fy = 0
fz = 0
p = 0
p_n = 0
p_m = 0

If (AtomType = "Au") Then
LoadParameters ("Au")

For i = 1 To NoAtoms
    p = 0
    For k = 1 To NoAtoms
        If (AtomPosAu(k).X <> 0 And AtomPosAu(k).Y <> 0 And
AtomPosAu(k).Z <> 0) Then
            If k <> i Then
                rx_ij = (rx(i, k, "Au", "Au"))
                ry_ij = (ry(i, k, "Au", "Au"))
                rz_ij = (rz(i, k, "Au", "Au"))
                r = (rx_ij ^ 2 + ry_ij ^ 2 + rz_ij ^ 2) ^ 0.5
                a_on_r = a / r
                p = p + a_on_r ^ m
            End If
        End If
    Next k
    Loc_Dens(i) = p
Next i
cmop2 = c * m / 2

For i = 1 To NoAtoms
    f = 0
    PosCounterSC = PosCounterSC + 1

```



```

If i <> j And i <= No_Atoms Then
  rx_ij = (rx(i, j, "Au", "Au"))
  ry_ij = (ry(i, j, "Au", "Au"))
  rz_ij = (rz(i, j, "Au", "Au"))
  r = (rx_ij ^ 2 + ry_ij ^ 2 + rz_ij ^ 2) ^ 0.5
  If (AtomPosAu(i).X <> 0 And AtomPosAu(i).Y <> 0 And
AtomPosAu(i).Z <> 0) Then
    If (r < RCutOff) Then
      a_on_r = a / r
      p_n = a_on_r ^ n
      p_m = a_on_r ^ m
      Inv_R = 1 / r
      Inv_Loc_Dens_j = (1 / Loc_Dens(j)) ^ 0.5
      Inv_Loc_Dens_i = (1 / Loc_Dens(i)) ^ 0.5
      Sum_Inv_Loc_Dens = Inv_Loc_Dens_j + Inv_Loc_Dens_i
      cm_Sum_Inv_Loc_Dens = cmop2 * Sum_Inv_Loc_Dens
      cm_Sum_p_m = cm_Sum_Inv_Loc_Dens * p_m
      np_n = n * p_n
      Diff = np_n - cm_Sum_p_m
      f = Diff * Inv_R
      f = E_Au * f
      R_x_on_R = ((rx_ij)) / r
      R_y_on_R = ((ry_ij)) / r
      R_z_on_R = ((rz_ij)) / r
      fx = fx + f * R_x_on_R
      fy = fy + f * R_y_on_R
      fz = fz + f * R_z_on_R
    End If
  End If
End If
Next i
End If
AtomForPot1(j).X = fx
AtomForPot1(j).Y = fy
AtomForPot1(j).Z = fz
Forces = f

End Function

```

---

Public Function LoadParameters(Element As String)

    If Element = "Au" Then

        E\_Au = 0.012793

        a = 4.08

        c = 34.408

        n = 10

        m = 8

        mass\_Au = 196.97

    End If

    If Element = "C" Then

        a\_C = 2 \* 0.77

        mass\_C = 12.01

    End If

End Function

---

Public Function Load\_Brenner\_Parameters(Element As String)

    If Element = "C" Then

        beta = 15

        n\_t = 0.8047

        c\_t = 19

        d = 2.5

        h = 1

        R\_t = 2.1

        S\_t = 1.29

        m\_t = 2.25

        D\_t = 0.2

        D\_e = 6.325

        r\_e = 1.28

        alfa = 0.0113

    End If

End Function

---

Public Function r(i As Double, j As Double, Atom\_Type As String)

    Dim dx As Double

```

Dim dy As Double
Dim dz As Double
If Atom_Type = "Sub" Then
    dx = AtomPosC(j).X - AtomPosC(i).X
    dy = AtomPosC(j).Y - AtomPosC(i).Y
    dz = AtomPosC(j).Z - AtomPosC(i).Z
End If

If Atom_Type = "Surf" Then
    dx = AtomPosAu(j).X - AtomPosAu (i).X
    dy = AtomPosAu (j).Y - AtomPosAu (i).Y
    dz = AtomPosAu (j).Z - AtomPosAu (i).Z
End If

If Atom_Type = "Both" Then
    dx = AtomPosC(j).X - AtomPosAu (i).X
    dy = AtomPosC(j).Y - AtomPosAu (i).Y
    dz = AtomPosC(j).Z - AtomPosAu (i).Z
End If

r = (dx ^ 2 + dy ^ 2 + dz ^ 2) ^ 0.5

```

End Function

---

```

Public Function LJ_U(i As Double, No_Atoms As Double, Element_Sub As
String, Element_Surf As String)

```

```

    Dim Epsilon As Double
    Dim Sigma As Double
    Dim u As Double
    Dim Sigma_on_r As Double
    Dim j As Double
    Dim r_d As Double

```

```

    Epsilon = 0.022
    Sigma = 2.74

```

```

    For j = 1 To No_Atoms
        r_d = r(i, j, "Both")
    
```

```

    If r_d < 7 Then
        Sigma_on_r = Sigma / r_d
        u = u - 4 * Epsilon * ((Sigma_on_r) ^ 12 - (Sigma_on_r) ^6)
    End If
Next j

```

```

LJ_U = -u

```

```

End Function

```

---

```

Public Sub FCC_111(i_max As Double, j_max As Double, k_max As Double, a As
Single)

```

```

    Dim i As Integer
    Dim j As Integer
    Dim k As Integer
    Dim n As Integer
    Dim f As Integer
    Dim ff As Integer
    Dim ak As Single
    Dim s As Single
    Dim ss As Single
    Dim xx As Single

```

```

    ak = a / Sqr(2)
    s = Sqr(3) * (ak / 2)
    ss = (Sqr(3) / 3) * (ak / 2)
    xx = a / Sqr(3)
    For k = 1 To k_max
        For j = 1 To j_max
            For i = 1 To i_max
                ff = ((k - 1) Mod 3) - 1
                f = j Mod 2
                If f = 0 Then
                    f = 1
                Else
                    f = 0
                End If
                n = n + 1
            Next i
        Next j
    Next k

```

```

    AtomPosAu(n).X = i * ak + f * (ak / 2) + ff * (ak / 2)
    AtomPosAu(n).Y = j * s + ff * ss
    AtomPosAu(n).Z = k * xx
  Next i
Next j
Next k

End Sub

```

---

```

Public Sub MoveSubstrate(Nz_move As Integer, Move_Step As Double, Nx As
Double, Ny As Double, Nz As Double, U_Start As Double, E_min As Double,
Element_Sub As String, Element_Surf As String, No_Atoms_Surf As Double,
Sub_Potential As String, Surf_Potential As String, Surf_Sub_Potential As String)

  Dim En As Double
  Dim u As Double
  Dim i As Integer
  Dim j As Integer
  Dim k As Integer
  Dim Atom_Nr As Integer
  Dim U_Start2 As Double
  u = Total_Energy(Nx * Ny * Nz, No_Atoms_Surf, Element_Surf,
Element_Sub, Surf_Potential, Sub_Potential, Surf_Sub_Potential)
  U_Start2 = u
  For i = 1 To Nx * Ny
    Atom_Nr = Atom_Nr + 1
    AtomPosC(Atom_Nr).X = AtomPosC(Atom_Nr).X + Move_Step
    DoEvents
    u = Total_Energy(Nx * Ny * Nz, No_Atoms_Surf, Element_Surf,
Element_Sub,
Surf_Potential, Sub_Potential, Surf_Sub_Potential)
    If U_Start < 0 Then
      En = u - U_Start
    Else
      En = U_Start - u
    End If

    If En < E_min Then
      E_min = En
    End If
  Next i

```

```

Else
  AtomPosC(Atom_Nr).X = AtomPosC(Atom_Nr).X - Move_Step * 2
  u = Total_Energy(Nx * Ny * Nz, No_Atoms_Surf, Element_Surf,
  Element_Sub,
  Surf_Potential, Sub_Potential, Surf_Sub_Potential)
  En = u - U_Start
  If En < E_min Then
    E_min = En
  Else
    AtomPosC(Atom_Nr).X = AtomPosC(Atom_Nr).X + Move_Step
  End If
End If
Next i
End Sub

```

---

```

Public Function Total_LJ(No_Atoms_Sub As Double, No_Atoms_Surf As Double,
Element_Sub As String, Element_Surf As String)

```

```

  Dim i As Double
  Dim u As Double
  For i = 1 To No_Atoms_Surf
    u = LJ_U(i, No_Atoms_Sub, Element_Sub, Element_Surf)
  Next i
  Total_LJ = u

```

```

End Function

```

---

```

Public Function Total_Energy(No_Atoms_Sub As Double, No_Atoms_Surf As
Double, Element_Surf As String, Element_Sub As String, Surf_Potential As
String, Sub_Potential As String, Surf_Sub_Potential As String)

```

```

  Dim SC As Double
  Dim B As Double
  Dim LJ As Double
  SC = SC_Crystal_Energy(No_Atoms_Surf, Element_Surf)
  B = total_brenner(No_Atoms_Sub, Element_Sub)
  LJ = Total_LJ(No_Atoms_Sub, No_Atoms_Surf, Element_Sub,
  Element_Surf)
  Total_Energy = SC + B + LJ

```

End Function

---

```
Public Sub MoveSurfaceX(Nz_move As Integer, Move_Step As Double, Nx As Integer, Ny As Integer, Nz As Integer, U_Start As Double, E_min As Double, Element As String)
```

```
    Dim E As Double
    Dim u As Double
    Dim i As Integer
    Dim j As Integer
    Dim k As Integer
    Dim Atom_Nr As Integer
    Atom_Nr = (Nz_move - 1) * Nx * Ny
    For j = 1 To Ny
        For i = 1 To Nz
            Atom_Nr = Atom_Nr + 1
            AtomPosAu(Atom_Nr).Z = AtomPosAu(Atom_Nr).Z + Move_Step
            u = SC_Crystal_Energy(Nx * Ny * Nz, Element)
            E = u - U_Start
            If E < E_min Then
                E_min = E
            Else
                AtomPosAu(Atom_Nr).Z = AtomPosAu(Atom_Nr).Z - Move_Step^2
                u = SC_Crystal_Energy(Nx * Ny * Nz, Element)
                E = u - U_Start
                If E < E_min Then
                    E_min = E
                Else
                    AtomPosAu(Atom_Nr).Z = AtomPosAu(Atom_Nr).Z + Move_Step
                End If
            End If
        Next i
    Next j
End Sub
```

---

```
Public Sub MoveSurfaceY(Nz_move As Integer, Move_Step As Double, Nx As Integer, Ny As Integer, Nz As Integer, U_Start As Double, E_min As Double, Element As String)
```

```

    Dim E As Double
    Dim u As Double
    Dim i As Integer
    Dim j As Integer
    Dim k As Integer
    Dim Atom_Nr As Integer
    Atom_Nr = (Nz_move - 1) * Nx * Ny
    For j = 1 To Nz
        For i = 1 To Nx
            Atom_Nr = Atom_Nr + 1
            AtomPosAu(Atom_Nr).Y = AtomPosAu(Atom_Nr).Y + Move_Step
            u = SC_Crystal_Energy(Nx * Ny * Nz, Element)
            E = u - U_Start
            If E < E_min Then
                E_min = E
            Else
                AtomPosAu(Atom_Nr).Y = AtomPosAu(Atom_Nr).Y - Move_Step^2
                u = SC_Crystal_Energy(Nx * Ny * Nz, Element)
                E = u - U_Start
                If E < E_min Then
                    E_min = E
                Else
                    AtomPosAu(Atom_Nr).Y = AtomPosAu(Atom_Nr).Y + Move_Step
                End If
            End If
        Next i
    Next j
End Sub
```

---

```
Public Function Optimize(i As Double, StepS As Double, Nx As Double, Ny As Double, Nz As Double, Emin As Double, Substrate_Element As String, Surface_Element As String, Uvoor As Double, No_Surf_Atoms As Double, fmin As Double)
```



```

MoveSubstrate Nz - (i - 1), StepS, Nx, Ny, Nz, Uvoor, Emin,
Substrate_Element, Surface_Element, No_Surf_Atoms, "", "", ""
MoveSurfaceZ Nz - (i - 1), StepS, Nx, Ny, Nz, Uvoor, Emin,
Substrate_Element, Surface_Element, No_Surf_Atoms, "", "", ""
MoveSurfaceY Nz - (i - 1), StepS, Nx, Ny, Nz, Uvoor, Emin,
Substrate_Element, Surface_Element, No_Surf_Atoms, "", "", ""

```

End Function

---

```

Public Function Brenner(j As Double, No_Atoms As Double, Element_Type As
String)

```

```

    Dim rd_ij As Double
    Dim W As Double
    Dim fc_ij As Double
    Dim g_theta As Double
    Dim i As Double
    Dim count As Integer
    Dim R1(1, 1, 1) As Double
    Dim r2(1, 1, 1) As Double
    Dim k As Double
    Dim cos_theta As Double
    Dim dot As Double
    Dim fc_ik As Double
    Dim V_R As Double
    Dim V_A As Double
    Dim rd_ik As Double
    Dim z_ij As Double
    Dim b_ij As Double
    Dim denominator As Double
    If j < NoAtomsC Then

```

```

For i = 1 To NoAtomsC

```

```

    count = 0

```

```

    If j > i Then

```

```

        rd_ij = r(i, j, "Sub")

```

```

        fc_ij = Calculate_fc_Brenner(rd_ij)

```

```

        If fc_ij > 0 Then

```

```

            V_R = (D_e / (S_t - 1)) * Exp(-beta * ((2 * S_t) ^ 0.5) * (rd_ij

```

```

- r_e))
V_A = (S_t * D_e / (S_t - 1)) * Exp(-beta * ((2 / S_t) ^ 0.5) *
(rd_ij - r_e))
z_ij = 0

For k = 1 To NoAtomsC
  If k <> i And k <> j Then
    rd_ik = r(i, k, "Sub")
    fc_ik = Calculate_fc_Brenner(rd_ik)
  If fc_ik > 0 Then
    R1(1, 0, 0) = (AtomPosC(i).X - AtomPosC(j).X)
    R1(0, 1, 0) = (AtomPosC(i).Y - AtomPosC(j).Y)
    R1(0, 0, 1) = (AtomPosC(i).Z - AtomPosC(j).Z)
    r2(1, 0, 0) = (AtomPosC(i).X - AtomPosC(k).X)
    r2(0, 1, 0) = (AtomPosC(i).Y - AtomPosC(k).Y)
    r2(0, 0, 1) = (AtomPosC(i).Z - AtomPosC(k).Z)
    dot = dot_product(R1(1, 0, 0), R1(0, 1, 0), R1(0, 0, 1),
r2(1, 0, 0), r2(0, 1, 0), r2(0, 0, 1))
    denominator = (r(i, j, "Sub") * r(i, k, "Sub"))
    cos_theta = dot / denominator
    g_theta = alfa * (1 + c_t ^ 2 / d ^ 2 - c_t ^ 2 / (d ^ 2 +
(h - cos_theta) ^ 2))
    z_ij = z_ij + fc_ik * g_theta * Exp(m * (rd_ij - rd_ik))
  End If
End If
Next k
End If
b_ij = (1 + z_ij) ^ (-n_t)
W = W + fc_ij * (V_R - b_ij * V_A)
End If
Next i
End If
Brenner = W

End Function

```

---

```

Public Function total_brenner(No_Atoms As Double, Element_Sub As String)

```

```

  Dim i As Double

```

```
Dim E As Double
For i = 1 To No_Atoms
    E = E + Brenner(i, No_Atoms, Element_Sub)
Next i
total_brenner = E
```

End Function

---

```
Public Function Calculate_fc_Brenner(rd_ij As Double)
```

```
Dim fc As Double
If rd_ij < R_t Then fc = 1
If rd_ij < R_t + D_t And rd_ij > R_t - D_t Then fc = 0.5 - 0.5 * Sin((22 / 7) / 2
* (rd_ij - R_t) / (D_t))
If rd_ij > R_t + D_t Then fc = 0
Calculate_fc_Brenner = fc
```

End Function

---

```
Public Function dot_product(X1 As Double, Y1 As Double, Z1 As Double, X2 As
Double, Y2 As Double, Z2 As Double)
```

```
Dim x_sum As Double
Dim y_sum As Double
Dim z_sum As Double

x_sum = X1 * X2
y_sum = Y1 * Y2
z_sum = Z1 * Z2
dot_product = x_sum + y_sum + z_sum
```

End Function

---

```
Public Sub C_Substrate(i_max As Double, j_max As Double, k_max As Double, a
As Single)
```

```

Dim s As Integer
Dim i As Integer
Dim No As Double
Dim j As Integer
Dim counter3 As Double
Dim counter4 As Double
Dim counter6 As Integer
Dim k As Integer
Dim r As Double
r = 1 * 0.70899
counter6 = 1
counter3 = 0
counter4 = 0

For k = 0 To k_max - 1
  For j = 0 To j_max - 1
    counter6 = counter6 + 1
    For i = 0 To (i_max) / 2 - 1
      No = No + 1
      If k Mod 2 = 0 Then
        counter3 = 0
        counter4 = 0
      Else
        counter3 = (4 * r ^ 2 - r ^ 2) ^ 0.5
        counter4 = r
      End If
      If counter6 <= 2 Then
        AtomPosAu(No).X = 2 * i * ((4 * r ^ 2 - r ^ 2) ^ 0.5) + counter3
      Else
        AtomPosAu(No).X = 2 * i * ((4 * r ^ 2 - r ^ 2) ^ 0.5) + counter3 + (4 * r
          ^ 2 - r ^ 2) ^ 0.5
      End If
      AtomPosAu(No).Y = Fix(j / 2) * 2 * r + (j - Fix(j / 2)) * r + counter4 + (j -
        Fix(j / 2)) * 0.70899 + counter4
      AtomPosAu(No).Z = k * 3.34
    Next i
    If counter6 = 4 Then counter6 = 0
  Next j

```

Next k  
End Sub

---

## F.2 pDisplay

Option Explicit

```
Public iFlatSmooth As Integer
Public Sphere1 As GLUquadric
Public Sphere2 As GLUquadric
Public dBallRadiusC As Double
Public dBallRadiusAu As Double
Public xRot, yRot As GLfloat
Public bAppRunning As GLboolean
```

Public Type PIXELFORMATDESCRIPTOR

```
nSize As Integer
nVersion As Integer
dwFlags As Long
iPixelFormat As Byte
cColorBits As Byte
cRedBits As Byte
cRedShift As Byte
cGreenBits As Byte
cGreenShift As Byte
cBlueBits As Byte
cBlueShift As Byte
cAlphaBits As Byte
cAlphaShift As Byte
cAccumBits As Byte
cAccumRedBits As Byte
cAccumGreenBits As Byte
cAccumBlueBits As Byte
cAccumAlpgaBits As Byte
cDepthBits As Byte
cStencilBits As Byte
```

```
cAuxBuffers As Byte
iLayerType As Byte
bReserved As Byte
dwLayerMask As Long
dwVisibleMask As Long
dwDamageMask As Long
End Type
```

```
Public Const PFD_TYPE_RGBA = 0
Public Const PFD_TYPE_COLORINDEX = 1
Public Const PFD_MAIN_PLANE = 0
Public Const PFD_DOUBLEBUFFER = 1
Public Const PFD_DRAW_TO_WINDOW = &H4
Public Const PFD_SUPPORT_OPENGL = &H20
Public Const PFD_NEED_PALETTE = &H80
Public Declare Function ChoosePixelFormat Lib "GDI32" (ByVal hdc As Long,
pfd As PIXELFORMATDESCRIPTOR) As Long
Public Declare Function CreatePalette Lib "GDI32" (pPal As LOGPALETTE) As
Long
Public Declare Sub DeleteObject Lib "GDI32" (hObject As Long)
Public Declare Sub DescribePixelFormat Lib "GDI32" (ByVal hdc As Long,
ByVal PixelFormat As Long, ByVal nBytes As Long, pfd As
PIXELFORMATDESCRIPTOR)
Public Declare Function GetDC Lib "GDI32" (ByVal hWnd As Long) As Long
Public Declare Function GetPixelFormat Lib "GDI32" (ByVal hdc As Long) As
Long
Public Declare Sub GetSystemPaletteEntries Lib "GDI32" (ByVal hdc As Long,
ByVal start As Long, ByVal entries As Long, ByVal ptrEntries As Long)
Public Declare Sub RealizePalette Lib "GDI32" (ByVal hPalette As Long)
Public Declare Sub SelectPalette Lib "GDI32" (ByVal hdc As Long, ByVal
hPalette As Long, ByVal bIn As Long)
Public Declare Function SetPixelFormat Lib "GDI32" (ByVal hdc As Long, ByVal
i As Long, pfd As PIXELFORMATDESCRIPTOR) As Boolean
Public Declare Sub SwapBuffers Lib "GDI32" (ByVal hdc As Long)
Public Declare Function wglCreateContext Lib "OpenGL32" (ByVal hdc As
Long) As Long
Public Declare Sub wglDeleteContext Lib "OpenGL32" (ByVal hContext As Long)
Public Declare Sub wglMakeCurrent Lib "OpenGL32" (ByVal l1 As Long, ByVal
l2 As Long)
```

```
Public hPalette As Long
Public hGLRC As Long
```

```
Public sRenderer As String, sVendor As String, sExtensions As String, sVersion
As String
```

---

```
Public Sub FatalError(ByVal sMessage As String)
```

```
    MsgBox "Fatal Error: " & sMessage, vbCritical + vbApplicationModal +
vbOKOnly + vbDefaultButton1, "Fatal Error In " & App.Title
    Unload frmDisplay
    Set frmDisplay = Nothing
    End
```

```
End Sub
```

---

```
Private Sub SetupPixelFormat(ByVal lhDC As Long)
```

```
    Dim pfd As PIXELFORMATDESCRIPTOR
    Dim PixelFormat As Integer

    pfd.nSize = Len(pfd)
    pfd.nVersion = 1
    pfd.dwFlags = PFD_SUPPORT_OPENGL Or PFD_DRAW_TO_WINDOW
Or PFD_DOUBLEBUFFER Or PFD_TYPE_RGBA
    pfd.iPixelFormat = PFD_TYPE_RGBA
    pfd.cColorBits = 24
    pfd.cDepthBits = 16
    pfd.iLayerType = PFD_MAIN_PLANE

    PixelFormat = ChoosePixelFormat(lhDC, pfd)
    If PixelFormat = 0 Then FatalError "Could not retrieve pixel format!"
    SetPixelFormat lhDC, PixelFormat, pfd
```

```
End Sub
```

---

```
Private Sub SetupPalette(ByVal lhDC As Long)
```

```
    Dim PixelFormat As Long
    Dim pfd As PIXELFORMATDESCRIPTOR
```

```

Dim pPal As LOGPALETTE
Dim PaletteSize As Long
PixelFormat = GetPixelFormat(lhDC)
DescribePixelFormat lhDC, PixelFormat, Len(pfd), pfd
If (pfd.dwFlags And PFD_NEED_PALETTE) <> 0 Then
    PaletteSize = 2 ^ pfd.cColorBits
Else
    Exit Sub
End If

pPal.palVersion = &H300
pPal.palNumEntries = PaletteSize
Dim redMask As Long
Dim GreenMask As Long
Dim BlueMask As Long
Dim i As Long
redMask = 2 ^ pfd.cRedBits - 1
GreenMask = 2 ^ pfd.cGreenBits - 1
BlueMask = 2 ^ pfd.cBlueBits - 1
For i = 0 To PaletteSize - 1
    With pPal.palPalEntry(i)
        .peRed = i
        .peGreen = i
        .peBlue = i
        .peFlags = 0
    End With
Next
GetSystemPaletteEntries lhDC, 0, 256, VarPtr(pPal.palPalEntry(0))
hPalette = CreatePalette(pPal)
If hPalette <> 0 Then
    SelectPalette lhDC, hPalette, False
    RealizePalette lhDC
End If
End Sub

```

---

```

Public Sub StartOpenGL(ByVal lTargetHDC As Long)

```

```

    SetupPixelFormat lTargetHDC
    SetupPalette lTargetHDC

```



```
hGLRC = wglCreateContext(ITargetHDC)
wglMakeCurrent ITargetHDC, hGLRC
```

End Sub

---

```
Public Sub StopOpenGL()
```

```
    If hGLRC <> 0 Then
        wglMakeCurrent 0, 0
        wglDeleteContext hGLRC
    End If
    If hPalette <> 0 Then
        DeleteObject hPalette
    End If
```

End Sub

---

```
Public Sub glRGB(r%, G%, B%)
```

```
    glColor3f r / 255, G / 255, B / 255
```

End Sub

---

```
Public Sub setuprc()
```

```
    glClearColor 1#, 1#, 1#, 0#
    glFrontFace ffCCW
    RenderScene
```

End Sub

---

```
Public Sub Main()
```

```
    dBallRadiusC = 0.77
    dBallRadiusAu = 4.08 / (2 * 2 ^ 0.5)
    xRot = 90
    yRot = 90

    quadSphere = gluNewQuadric()
```

```
Sphere1 = gluNewQuadric()
Sphere2 = gluNewQuadric()

gluQuadricOrientation Sphere1, qoInside
gluQuadricDrawStyle Sphere1, qdsFill
gluQuadricNormals Sphere1, qnSmooth
gluQuadricTexture Sphere1, True

gluQuadricOrientation Sphere2, qoInside
gluQuadricDrawStyle Sphere2, qdsFill
gluQuadricNormals Sphere2, qnSmooth
gluQuadricTexture Sphere2, True

StartOpenGL frmDisplay.hDC
frmDisplay.Show
Call setuprc

bAppRunning = True
While bAppRunning = True
    Call RenderScene
    DoEvents
Wend

Unload frmDisplay
Set frmDisplay = Nothing
```

End Sub

---

Public Sub RenderScene()

```
glClear clrColorBufferBit + clrDepthBufferBit
DrawObjects
glFlush
```

End Sub

---

```
Public Sub DrawObjects()  
  
    CSphere  
    AuSphere  
    SwapBuffers frmDisplay.hDC  
  
End Sub
```

---

```
Public Sub CSphere()  
  
    Dim red As GLubyte  
    Dim green As GLubyte  
    Dim blue As GLubyte  
    Dim h As Integer  
    h = 1  
  
    For i = 1 To NoAtomsC  
  
        glPushMatrix  
        glRotatef xRot - 90, 1, 0, 0  
        glRotatef yRot - 90, 0, 0, 1  
        glTranslatef AtomPosC(i).X, AtomPosC(i).Y, AtomPosC(i).Z  
  
        red = 0  
        green = 50  
        blue = 255  
        glColor3ub red, green, blue  
        gluSphere Sphere2, dBallRadiusC, 50, 50  
        glPopMatrix  
    Next I  
  
End Sub
```

---

```
Public Sub AuSphere()  
    Dim red As GLubyte  
    Dim green As GLubyte
```

Dim blue As GLubyte

For i = 1 To NoAtomsAu '+ NoAtomsAu2 'NoAtomsAu \* 2

glPushMatrix

glRotatef xRot - 90, 1, 0, 0

glRotatef yRot - 90, 0, 0, 1

glTranslatef AtomPosAu(i).X, AtomPosAu(i).Y, AtomPosAu(i).Z

red = 0

green = 200

blue = 0

glColor3ub red, green, blue

gluSphere Sphere1, dBallRadiusAu, 50, 50

glPopMatrix

Next i

End Sub

---

Public Sub SetUpLighting()

glClearColor 0, 0, 0, 0

glClearDepth 1

glEnable glcDepthTest

glDepthFunc cfLEqual

glMaterialfv faceFront, mprAmbient, 1#

glMaterialfv faceFront, mprDiffuse, 1.5

glMaterialfv faceFront, mprSpecular, 0.5

glMaterialfv faceFront, mprShininess, 20

glLightModelfv lmAmbient, 5

glEnable glcLighting 'gives an outside ring/darker colour

glEnable glcLight0 'gives lighter colour on the inside

glColorMaterial faceFront, cmmAmbientAndDiffuse

glEnable glcColorMaterial

glShadeModel smSmooth

End Sub

---

### F.3 pIsland

Option Explicit

Dim counts As Integer

Dim Island\_Count As Integer

Dim Tot\_Dens As Double

Dim Total\_Sites As Double

Dim Total\_Atoms As Double

---

Private Sub Command1\_Click()

On Error GoTo SaveError:

CMD1.Action = 1

Dim info As Integer

Dim Element As String

Load\_Crystal CMD1.filename

Plot\_XY Pic, 4.08, 1

SaveError:

Exit Sub

Resume

End Sub

---

Private Sub Command2\_Click()

Dim Island\_Sites(10) As Double

Dim Avg\_Sites As Double

Dim Island\_Atoms(10) As Double

Dim Avg\_Size As Double

Dim Avg\_Density As Double

Dim Sites As Double

Dim Atoms As Double

```

If Island_Count = 0 Then
    ebAvgDensity = " "
    ebAvgSize = " "
End If

Island_Count = Island_Count + 1
Island_Sites(Island_Count) = No_Sites_Size(CDbl(Text1.Text), CDbl(Text2.Text),
CDbl(Text3.Text), CDbl(Text4.Text))
ebNoSites = Island_Sites(Island_Count)
Island_Atoms(Island_Count) = No_Atoms_Size(CDbl(Text1.Text),
CDbl(Text2.Text), CDbl(Text3.Text), CDbl(Text4.Text))
ebNoAtoms = Island_Atoms(Island_Count)
Total_Sites = Total_Sites + Island_Sites(Island_Count)
Total_Atoms = Total_Atoms + Island_Atoms(Island_Count)
ebDensity = Island_Atoms(Island_Count) / Island_Sites(Island_Count)
Tot_Dens = Tot_Dens + Island_Atoms(Island_Count) /
Island_Sites(Island_Count)
ebAvgDensity = Tot_Dens
ebAvgSize = Total_Sites
If Island_Count = 5 Then
    Avg_Size = Total_Sites / 5
    Avg_Density = Tot_Dens / 5
    ebAvgDensity = Avg_Density
    ebAvgSize = Avg_Size
    Island_Count = 0
End If

End Sub

```

---

```

Private Sub Pic_MouseDown(Button As Integer, Shift As Integer, X As Single, Y
As Single)
    Dim index As Integer
    Dim nn As Integer
    Dim d As Single
    Dim a As Double
    a = 4.08

```

$d = (a / \text{Sqr}(2)) / 2$

index = 1

If index = 0 Then

    nn = Find\_atoom\_Nr(CDbl(X), CDbl(Y), a)

    If Button = 1 Then

        Pic.Circle (Atoms(nn).X, Atoms(nn).Y), d, QBColor(12))

    End If

    If Button = 2 Then

        Plot\_XY Pic, 4.08, 1

    End If

End If

If index = 1 Then

nn = Find\_atoom\_Nr(CDbl(X), CDbl(Y), a)

If Button = 1 Then

    Lbl\_muis.Caption = nn

    Pic.Circle (Atoms(nn).X, Atoms(nn).Y), d, QBColor(12))

End If

If Button = 2 Then

    Plot\_XY Pic, 4.08, 1

End If

End If

counts = counts + 1

If counts = 1 Then

    Text1.Text = CDbl(X) 'Atoms(nn).X

    Text3.Text = CDbl(Y) 'Atoms(nn).X

End If

If counts = 2 Then

    Text2.Text = CDbl(X) 'Atoms(nn).X

End If

If counts = 3 Then

    Text4.Text = CDbl(Y) 'Atoms(nn).X

End If

If counts = 3 Then counts = 0

End Sub

---

```
Private Sub Pic_MouseMove(Button As Integer, Shift As Integer, X As Single, Y  
As Single)
```

```
    Static x_mouse As Single
```

```
    Static y_mouse As Single
```

```
    Static z_mouse As Single
```

```
    Dim index As Integer
```

```
    index = 1
```

```
    If index = 0 Then
```

```
        x_mouse = X
```

```
        y_mouse = Y
```

```
    End If
```

```
    If index = 1 Then
```

```
        x_mouse = X
```

```
        z_mouse = Y
```

```
    End If
```

```
    If index = 2 Then
```

```
        y_mouse = X
```

```
        z_mouse = Y
```

```
    End If
```

```
    Lbl_muis.Caption = "(" & Str(x_mouse) & "," & Str(y_mouse) & "," &  
Str(z_mouse) & ")"
```

End Sub

---

```
Option Explicit
```

```
Type Atom
```

```
    X As Double
```

```
    Y As Double
```

```
    Z As Double
```

```
End Type
```



```
Global Atoms(500001) As Atom
Dim No_Atoms As Double
Dim Atom_Type As String
Dim a As Double
```

---

```
Sub Load_Crystal(filename As String)
```

```
Dim i As Double
Dim j As Integer
Dim No As Double
```

```
'Input #1, j
Dim ab As Double
```

```
Open filename For Input As #1
```

```
Input #1, No_Atoms
'ReDim Atoms(No_Atoms)
Input #1, j
For i = 1 To No_Atoms
    No = No + 1
    If No > No_Atoms - 1 Then Exit For
    Input #1, i, Atoms(No).X, Atoms(No).Y, Atoms(No).Z
Next i
```

```
Close #1
```

```
Load_Parameters
```

```
End Sub
```

```
Sub Load_Parameters()
```

```
If Atom_Type = "Au" Then
    a = 4.08
End If
```

End Sub

---

Public Sub Plot\_XY(PF As Control, a As Double, Depth\_Scale As Integer)

Dim d As Single  
Dim ds As Single  
Dim i As Double  
Dim j As Integer  
Dim k As Integer  
Dim n As Integer

d = (a / Sqr(2)) / 2  
ds = d

PF.Cls  
n = 0

For i = 1 To No\_Atoms  
    PF.Circle (Atoms(i).X, Atoms(i).Y), d, QBColor(10)  
Next i

End Sub

Public Function No\_Sites\_Size(X1 As Double, X2 As Double, Y1 As Double, Y2  
As Double)

Dim X\_Size As Double  
Dim Y\_Size As Double  
Dim No\_Sites\_X As Integer  
Dim No\_Sites\_Y As Integer  
Dim No\_Sites As Integer

X\_Size = Abs(X2 - X1)  
Y\_Size = Abs(Y2 - Y1)

No\_Sites\_X = X\_Size / 2.885  
No\_Sites\_Y = Y\_Size / 2.885

```
No_Sites = No_Sites_X * No_Sites_Y
```

```
No_Sites_Size = No_Sites
```

```
End Function
```

---

```
Public Function Find_atoom_Nr(X As Double, Y As Double, a As Double)
```

```
    Dim dx As Double
```

```
    Dim dy As Double
```

```
    Dim dz As Double
```

```
    Dim d As Single
```

```
    Dim i As Double
```

```
    Dim r As Double
```

```
    d = (a / Sqr(2)) / 2
```

```
    For i = 1 To No_Atoms
```

```
        dx = Atoms(i).X - X
```

```
        dy = Atoms(i).Y - Y
```

```
        r = (dx ^ 2 + dy ^ 2) ^ 0.5
```

```
        If r < d / 1.5 Then
```

```
            Find_atoom_Nr = i
```

```
            Exit For
```

```
        End If
```

```
    Next i
```

```
End Function
```

---

Public Function No\_Atoms\_Size(X1 As Double, X2 As Double, Y1 As Double, Y2  
As Double)

Dim i As Double

Dim Atom\_Count As Double

For i = 1 To No\_Atoms

    If Atoms(i).X >= X1 And Atoms(i).X <= X2 And Atoms(i).Y <= Y1 And  
        Atoms(i).Y >= Y2 Then

        Atom\_Count = Atom\_Count + 1

    End If

Next i

No\_Atoms\_Size = Atom\_Count

End Function

---

## REFERENCES

- [1] Feynman, R. P. There's plenty of room at the bottom. *Eng. Sci.* 23, 22–36, 1960.
- [2] G. Binning, H. Rohrer, Ch. Gerber and E. Heibel, *Phys. Rev. Lett.*, 49, 57, 1982.
- [3] G. Binning, H. Rohrer, Ch. Gerber and E. Heibel, *Appl. Phys. Lett.*, 40, 178, 1982.
- [4] Hollistar, P., Weener, J., Vas, C., Harper, T., Scientifica White Paper, Nanoparticles, 2003. Available at:  
[http://nanotechweb.org/dl/wp/nanoparticles\\_WP.pdf](http://nanotechweb.org/dl/wp/nanoparticles_WP.pdf)
- [5] J.M. Kohler, *Nanotechnology: An introduction into nanostructuring techniques*, Wiley, 2004.
- [6] Gates, B. D. *et al.* New approaches to nanofabrication: molding, printing, and other techniques. *Chem. Rev.* 105, 1171–1196, 2005.
- [7] K. S. Johnson, J. H. Thywissen, N. H. Dekker, K. K. Berggren, A. P. Chu, R. Younkin and M. Prentiss, *Science*, 280, 1583, 1998.
- [8] H. Dai, N. Franklin and J. Han, *Appl. Phys. Lett.*, 73, 1508, 1998; S. Hong, J. Zhu and C. A. Mirkin, *Science*, 286, 523, 1999.
- [9] Y.N. Xia, J.A. Rogers, K.E. Paul, G.M. Whitesides, *Chem. Rev.* 99, 1823–1848, 1999.
- [10] B.H. Kear, R.K. Sadangi, S.C. Liao. Synthesis of WC/Co/diamond nanocomposites. In *Proc. of the Joint NSF-NIST Conf. on Nanoparticles*, 1997.

- [11] S. Amoruso, G. Ausanio, S. Bruzzese, N. Lanotte, P. Scardi, M. Vitiello, X. Wang, J. Phys.: Condens. Matter 18 No 4, L49-L53, 2006.
- [12] Messing, G.L., S. Zhang, U. Selvaraj, R.J. Santoro, T. Ni. Synthesis of composite particles by spray pyrolysis. In Proc. of the Joint NSF-NIST Conf. on Ultrafine Particle Engineering (May 25-27, Arlington, VA), 1994.
- [13] Rao, N.P., N. Tymiak, J. Blum, A. Neuman, H.J. Lee, S.L. Girshick, P.H. McMurry, J. Heberlein, Nanostructured materials production by hypersonic plasma particle deposition. In Proc. of the Joint NSF-NIST Conf. on Nanoparticles, 1997.
- [14] H.H. Kung, E.I. Ko, Chem. Eng. J. 64, p 203, 1996.
- [15] A.G. Gnedovets, J. Phys. D: Appl. Phys. 32, pp 2162-2168, 1999.
- [16] Frenkel, A.I., Nemzer, S., Pister, I., Soussan, L., Harris, T., Sun, Y., Rafailovich, M.H., Chem. Phys. 123, 184701, 2005.
- [17] J. M. Auerbach, P.A. Monson, J Am Chem Soc. 41, 127, pp 14388-400, 2005.
- [18] O. Dehaese, X. Wallart, and F. Mollot, Applied Physics 66, Issue 1, pp. 52-54, 1995.
- [19] M. Siegert, M. Plischke, Phys Rev E Stat Phys Plasmas Fluids Relat Interdiscip Topics. 50(2), p 917-931, 1994.
- [20] P.V. Kamat, J. Phys. Chem. B. 2002, 106(32), pp 7729-7744.
- [21] R. Narayanan, M.A. El-Sayed, J. Phys. Chem. B. 2005, 109(26), pp 12663-12676.

- [22] B. Kim, S.L. Tripp, A. Wei, *Mat. Res. Soc. Symp. Proc. Vol. 676*, p. Y6.1.1 – Y6.1.6 2001.
- [23] K.L. Kelly, E. Coronado, L. Zhao, G.C. Schatz, *J. Phys. Chem. B.* 2003, 107(3), pp 668-677.
- [24] M. Faraday, *Philos. Trans. R. Soc. London* 147 (1857) 145.
- [25] K. Tanaka, *Nanotechnology Towards the 21st Century, Thin Solid Film*, 341, pp. 120-125, 1999.
- [26] J.M. Kohler, A. Csaki, J. Reichert, R. Moller, W. Straube, W. Fritzsche, *Selective Labeling of Oligonucleotide Monolayers by Metallic Nanobeads for Fast Optical Readout of DNA Chips, Sens. Act. B.*, 76, pp. 166-172, 2001.
- [27] G.C. Schatz, A.A. Lazarides, K.L. Kelly, T.R. Jensen, *Optical Properties of Metal Nanoparticles Aggregates Important in Biosensors, J. Mol. Struct. (Theochem)*, 529, pp. 59-63. Tanaka. K, 1999. *Nanotechnology Towards the 21st Century, Thin Solid Film*, 341, pp. 120-125, 2001.
- [28] K. Youngjin, R.C. Johnson, J.T. Hupp, *Gold Nanoparticle-Based Sensing of “Spectroscopically Silent” Heavy Metal Ions, Nano Lett.*, 1, pp. 165-167, 2001.
- [29] W.P. McConnell, J.P. Novak, L.C. Brousseau, R.R. Fuierer, R.C. Tenent, D.L. Feldheim, *Electronic and Optical Properties of Chemically Modified Metal Nanoparticles and Molecularly Bridged Nanoparticles, J. Phys. Chem*, 104, pp. 8925-8930, 2000.
- [30] M. Brust, D. Bethell, C.J. Kiely, D.J. Schiffrin, 1998. *Self-Assembly of Gold Nanoparticle Thin Film with Nonmetallic Optical and Electronic Properties, Langmuir*, 14, pp. 5425-5429.

- [31] R.M. Tromp, F.M. Ross, *Annual Review of Materials Science*, Vol. 30, p 431-449, 2000.
- [32] C. Schelling, J. Myslivecek, M. Mühlberger, H. Lichtenberger, Z. Zhong, B. Voigtländer, G. Bauer, F. Schäffler, *Physica Status Solide A*, Vol. 201, Iss. 2, p 324-328, 2004.
- [33] J.V. Barth, G. Constantini, K. Kern, *Nature*, Vol. 437, p 671-679, 2005.
- [34] G. Muller, H. Mettois, R. Rudolph, *Crystal Growth: From Fundamentals to Technology*, Elsevier, 2004.
- [35] A. Pimpinelli, J. Villain, *Physics of Crystal Growth*, Cambridge University Press, 1998.
- [36] F.V. Arfken, H.J. Weber, *Mathematical Methods for Physicists*, Academic Press, 1995.
- [37] R. Kaischew, *Arbeitstagung Festkörper Physik*, Dresden, p. 85, 1952.
- [38] B.R. Pamplin, R.Z. Bachrach, *Progress in Crystal Growth and Characterization*, Vol. 12, Pergamon Press, Oxford, 1986.
- [39] K.A. Jackson, *Kinetic Processes: Crystal Growth, Diffusion and Phase Transformation in Materials*, Wiley InterScience, 2004.
- [40] J.S. Rowlinson, B. Widom, *Molecular Theory of Capillarity*, Dover, New York, 2002.
- [41] M. Ohring, *The Materials Science of Thin Films 2<sup>nd</sup> Edition*, Academic Press, 2001.



- [42] J.B. Romé de l'Isle, *Cristallographie*, Paris, p 1783.
- [43] M. Bienfait, R. Boistelle, R. Kern, *Adsorption et Croissance, Cristalline*, CNRS, 1965, p 577.
- [44] P. Hartman, *Z, Kryst.* 107, 1956, p 721.
- [45] N. Ming, I. Sunagawa, *Journa of Crystal Growth*, Vol. 87, p 13, 1988.
- [46] K. Kimoto, I. Nishida, *Jpn Journal of Applied Physics*, 6, p 1047, 1967.
- [47] C. Herring, W. E. Kingston (Ed), *The Physics of Powder Metallurgy*, McGraw-Hill, New York, 1951.
- [48] H. Graoui, S. Giorgio, C.R. Henry, *Philos. Mag. B* 81, p 1649, 2001.
- [49] W. Vervisch, C. Mottet, J. Goniakowski, *Phys. Rev.* 65, p 245411, 2002.
- [50] T. Helgaker, P. Jørgensen, J. Olsen, *Molecular Electronic-Structure Theory*, Wiley, New York, 2000.
- [51] R. Martin, *Electronic Structure Methods*, Cambridge University Press, 2004.
- [52] B. L. Hammond, W. A. Lester, P. J. Reynolds, *Monte Carlo Methods in Ab Initio Chemistry*, World Scientific, 1994.
- [53] R. G. Parr, W. Yang, *Density-Functional Theory of Atoms and Molecules*, Oxford University Press, New York, 1989.
- [54] P. Ordejón, E. Artacho, R. Cachau, J. Gale, A. García, J. Junquera, J. Kohanoff, *Mat. Res. Soc. Symp. Proc.*, Vol. 677, 2001.

- [55] E. Artacho, D. Sánchez-Portal, P. Ordejón, A. García, J.M. Soler, *Physics Status Solidi B*, Volume 215, Issue 1, p 809 – 817, 1999.
- [56] D. R. Bowler, T. Miyazaki, M. J. Gillan, *J. of Physics, Condensed Matter* 14, p. 2781 – 2789, 2002.
- [57] D.C. Rappaport, *The Art of Molecular Dynamics Simulation*, Cambridge University Press, 2<sup>nd</sup> Edition, New York, 2004.
- [58] A. Tildesley, *Computer Simulation of Liquids*, Oxford University Press, New York, 1988.
- [59] P. Jensen, R. Bunker, *Computational Molecular Spectroscopy*, John Wiley & Sons, United Kingdom, 2000.
- [60] S. Gasiorowicz, *Quantum Physics*, John Wiley, 3<sup>rd</sup> Edition, 2003.
- [61] P. Hohenberg, W. Kohn, *Phys. Rev. B* 136, p. 864, 1964.
- [62] W. Kohn, L. J. Sham, *Phys. Rev. A* 140, p. 1133, 1965.
- [63] A.F. Voter, *Phys. Rev. B* 57, p. 13985, 1998.
- [64] A.F. Voter, *J. Chem. Phys.* 106, p. 4665, 1997.
- [65] A.F. Voter, *Phys. Rev. Lett.* 78, p. 3908, 1997.
- [66] M.R. Sørensen, A.F. Voter, *J. Chem. Phys.* 112, p. 9599, 2000.
- [67] K. Tominaga, *Heterogeneous Kinetics*, Springer Series in Chemical Physics, Vol 77, 2004.

[68] W. Bryc, Applied Probability and Stochastic Processes, Lecture Notes for 577/578 Class, 1996, Available online:

<http://math.uc.edu/~brycw/probab/books/applprob.pdf>

[69] J.L. Doob, Stochastic Processes, Wiley InterScience, 1990.

[70] A.T. Barucha-Reich, Elements of the Theory of Markov Processes and Their Applications, Dover Publications, 1987.

[71] M. Iosufesco, Finite Markov Chains and Their Applications, Wiley InterScience, Bucuresti, 1980.

[72] K. Binder, Monte Carlo Methods in Statistical Physics, Springer-Verlag, Berlin, 1979.

[73] A. Papulis, Probability, Random Variables and Stochastic Processes, McGraw-Hill, Tokyo, 1965.

[74] C.J. Atkins, Equilibrium Thermodynamics, 3<sup>rd</sup> Edition, Cambridge University Press, Cambridge, 1980.

[75] A. Papulis, Probability, Random Variables and Stochastic Processes, McGraw-Hill, Tokyo, 1965.

[76] K.A. Fichthorn, W.H. Weinberg, J. Chem. Phys. 95, p 1090, 1991.

[77] J.H. Beynon, J.R. Gilbert, Transition State Theory and its Application to Unimolecular Reactions: An Introduction, John Wiley and Sons, Chichester, 1983.

[78] M. N. Popescu, J. G. Amar, F. Family, Physical Review B 64, p 20504, 2001.

- [79] Computer Physics Communications: proceedings of STATPHYS Satellite Conference, Challenges in Computational Statistical Physics in the 21<sup>st</sup> Century, 2001 Available online: <http://www.physics.utoledo.edu/~jamar/ph/paper-jamarwfigs.pdf>
- [80] H. Luth, Solid Surfaces, Interfaces and Thin Films 4<sup>th</sup> Edition, Springer, 2001.
- [81] M. Biehl, Multiscale Methods in Epitaxial Growth, Birkhauser, 2005.
- [82] M. Biehl, F. Much, W. Kinzel, M. Ahr, Europhys. Lett., **56** (6) , p. 791-796, 2001.
- [83] A.C. Schindler, M.F. Gyure, G.D. Simms, D.D. Vvedensky, R.E. Caflisch, C. Connell, E. Lou, Theory of strain relaxation in heteroepitaxial systems, Phys. Rev. B **67** (2003) 075316.
- [84] F. Much, M. Biehl, Simulation of wetting-layer and island formation in heteroepitaxial growth, Europhys. Lett. **63** (2003) 14-20.
- [85] W. Rudzinski, Czechoslovak Journal of Physics **23**, p. 488-490, 1973.
- [86] Infoplease, Gold. Available at:  
<http://www.infoplease.com/periodictable.php?id=79>. Last accessed 1 May 2006.
- [87] Infoplease, Gold. Available at:  
<http://www.infoplease.com/periodictable.php?id=79>. Last accessed 1 May 2006.
- [88] P. Jensen, X. Blasé, P., Ordejon, Surface Science **564**, p.173-178, 2004.
- [89] G.M. Wang, J.J. BelBruno, S.D. Kenny, R. Smith, Surface Science **541**, Iss 1-3, p. 91-100, 2003.

- [90] A.F. Voter, Phys. Rev. B **34**, 6819 (1986).
- [91] M. Shcroeder, P. Smilauer, D.E. Wolf, D.E., Phys. Rev. B 55, Issue 16, p. 10814–10818, 1997.
- [92] A.F. Voter, Phys. Rev. B **34**, 6819 (1986).
- [93] M. Shcroeder, P. Smilauer, D.E. Wolf, Phys. Rev. B 55, Issue 16, p. 10814–10818, 1997.
- [94] H. Henkelman, H. Jonnson, Journal of Chemical Physics 115, No 21, 2001.
- [95] A.F. Voter, Phys Rev B 34, p. 6819, 1986.
- [96] K.A. Fichthorn, M.L. Merrick, M. Scheffler, Applied Physics A 75 Number 1, p. 17-23, 2002.
- [97] M.A. Weiss, Algorithms, Datastructures and Problem Solving with C++, Addison Wesley, 1996.
- [98] J.C. Heyraud, J.J. Metois, Acta Metallurgia, 1980, Vol 28, p 1789-1797.
- [99] J. Abrahamson, Carbon, 1973, Vol. 11, p 337-362.
- [100] M. Matsumoto and T. Nishimura, Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator, ACM Trans. on Modeling and Computer Simulations, 1998.
- [101] P. Jensen, J. Blasé, Phys. Rev. B. 70, 165042, 2004.
- [102] B. Yoon, W. D. Luedtke, J. Gao, U. Landman, J. Phys. Chem. B10 , p. 5882-5891, 2003.

- [103] S. Arcidiacono, N.R. Bieri, D. Poulikakos, C.P. Grigoropoulos, *International Journal of Multiphase Flow* 30, p.979–994, 2004.
- [104] R. Ferrando, G. Tréglià, *Phys. Rev. B* 50, Issue 16, p. 12104–12117, 1994.
- [105] C.H. Claassens, [inac@mintek.co.za](mailto:inac@mintek.co.za).
- [106] A. P. Sutton, J. Chen, *Philos. Mag. Lett.* 61, p 139, 1990.
- [107] J.P.K. Doye, D.J. Wales, *New J. Chem.*, p 733-744, 1998.
- [108] W.D. Luedtke, U. Landman, *Phys. Rev. Letters* 82, p 3835, 1999.
- [109] D. Shrivastava, C. Wei, K. Cho, *Applied Mechanics Review* 56, Issue 2, p 215-230, 2003.

## **BIOGRAPHY**

Ms. Ina Claassens was born in Welkom on 11 February 1980. She matriculated at the Welkom Gimnasium High School in 1997. She obtained the B.Sc. degree in Physics and Computer Information Systems in 2001, B.Sc. Hons in Physics in 2002, M.Sc. in Physics in 2003, all with distinction at the University of the Free-State. During her M.Sc. and Ph.D. studies, she was employed by the University of the Free-State as part-time lecturer in the physics department and in July 2005 she joined the Council for Minerals and Technology (Mintek) as a scientist.

## PUBLICATIONS & CONFERENCES

- Claassens, C.H., Hoffman, M.J.H., Terblans, J.J., Swart, H.C., Kinetic Monte Carlo Simulation of the Growth of Gold Nanostructures on Graphite. *Journal of Physics, Conference Series* 29, p 124-128, 2006.
- Claassens, C.H., Hoffman, M.J.H., Terblans, J.J., Swart, H.C., Kinetic Monte Carlo Simulation of Monolayer Gold Film Growth on a Graphite Substrate., *Surface and Interface Analysis, Volume 37, Issue 11*, p. 1021-1026, 2005.
- Claassens, C.H., Hoffman, M.J.H., Terblans, J.J., Swart, H.C., Kinetic Monte Carlo Simulation of Multilayer Gold Film Growth on a Graphite Substrate. . Accepted for special proceedings of Workshop on Modelling Electron Transport for Applications in Electron and X-ray Analysis and Metrology, NIST, Washington DC, 2004.
- Claassens, C.H., Hoffman, M.J.H., Terblans, J.J., Swart, H.C., Kinetic Monte Carlo Simulation of the Growth of Gold Nanostructures on Graphite. ACCMS-3 Conference for Molecular Modelling, Chinese Academy of Sciences, Beijing, China, 2005.
- Claassens, C.H., Hoffman, M.J.H., Terblans, J.J., Swart, H.C., Kinetic Monte Carlo Simulation of Multi-Layer Gold Film Growth on a Graphite Substrate. Workshop on Modelling Electron Transport for Applications in Electron and X-ray Analysis and Metrology, NIST, Washington DC, 2004.



- Claassens, C.H., Hoffman, M.J.H., Terblans, J.J., Swart, H.C., Kinetic Monte Carlo Simulation of MonoLayer Gold Film Growth on a Graphite Substrate. Workshop on Modelling Electron Transport for Applications in Electron and X-ray Analysis and Metrology, NIST, Washington DC, 2004.

- 
- 1
  - 2
  - 3
  - 4
  - 5
  - 6
  - 7
  - 8
  - 9
  - 10

---

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

---

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

---

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

---

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

---

98

99

100

101

102

103

104

105

106

107

108

109