Christoph Gleichweit, BSc

Kinetic Monte Carlo Simulations of Nucleation and Thin Film Growth of Hexaphenyl on Amorphous Mica

MASTER THESIS

For obtaining the academic degree
Diplom-Ingenieur

Master Programme of
Technical Physics

Graz University of Technology

Supervisor:

Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Adolf Winkler

Institute of Solid State Physics

Graz, May 2012

TU Graz
Graz University of Technology
Senat

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Graz, am …………………………… ………………………………………………..
(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

…………………………… ………………………………………………..
date (signature)

Acknowledgements

I want to thank Univ. Prof. Dr. tech. Adolf Winkler for his outstanding mentoring and dedicated support. Discussions on the topic as well as off-topic conversations made the time on the institute a remarkably fruitful experience.

Furthermore, special thanks go to Dr. Karin Zojer from the Institute of Theoretical and Computational Physics for support concerning the appeared theoretical and computational challenges of the simulations.

Moreover, I would like to thank all members of the surface science group for the pleasant and cooperative working atmosphere, and my family for the support over the years of my studies.

This research was supported by a grant from the FWF.

Abstract

Ultra-thin films of para-hexaphenyl on sputter amorphized mica(001) exhibit the surprising behavior of forming bimodal island size distributions. Small islands were observed besides bigger ones. It is believed that as long as the sample is in ultra-high vacuum, there exists a 2D-gas phase of monomers between the bigger islands, which vanishes by nucleating a lot of small islands when exposed to air. Probably due to the adsorption of water on the surface the energetics of the monomers on the surface change and islands form. This behavior is named adsorbate induced subsequent nucleation.

Using KMC (Kinetic Monte Carlo) simulations this effect was tried to be reproduced with the objective of extracting information about the critical cluster size and the island size distributions. For this reason an algorithm for edge hopping that respects also the critical island size was developed. The system initializes by starting with an initial concentration of monomers on the surface. These monomers suddenly start to diffuse and hence nucleate islands with the specified diffusion barriers and critical nucleus sizes. Particles that are part of stable clusters are not anymore allowed to detach from the islands. The simulation stops as soon as all molecules are incorporated into stable islands. The resulting island densities were compared to the experimental results. It turned out that a critical island size of $i = 7 \pm 1$ matches the experiments. Moreover, the island size distributions were compared to the ones obtained from the AFM (Atomic Force Microscopy) images and were found to agree qualitatively.

Furthermore, a computational technique called Quasicontinuum Monte Carlo (QCMC) method was adjusted to incorporate an effect called attachment limitation to study repulsive adsorbate interactions at the island rim, which is believed to be the reason for formation of the 2D-gas phase in vacuum.

Zusammenfassung

Ultra-dünne Filme von para-Hexaphenyl auf sputter-amorphisiertem Glimmer(001) zeigen das überaschende Verhalten der Bildung von bimodalen Inselgrößenverteilungen. Kleine Inseln konnten neben größeren beobachtet werden. Man geht davon aus, dass, solange sich die Probe im Ultrahochvakuum befindet, eine 2D-Gasphase von Monomeren existiert, die durch Nukleation von kleinen Inseln verschwindet sobald sie Luft ausgesetzt wird. Wahrscheinlich ändern sich die Energetik der Monomere an der Oberfläche durch Adsorption von Wasser. Dieses Verhalten wird Adsorbat-induzierte nachträgliche Nukleation genannt.

Es wurde mittels KMC (Kinetic Monte Carlo) Simulationen versucht diesen Effekt mit dem Ziel zu reproduzieren, Informationen über die kritische Clustergröße und die Inselgrößenverteilungen zu bekommen. Zu diesem Zweck wurde ein Algorithmus entwickelt, der die Bewegung von Teilchen auf den Rand von Inseln begrenzt, sowie die Angabe einer kritische Clustergröße unterstützt. Das System initialisiert mit einer gewissen Menge an Monomeren an der Oberfläche. Diese Monomere beginnen plötzlich zu diffundieren und folglich zu nukleieren mit den angegebenen Diffusionsbarrieren und kritischen Keimgrößen. Moleküle die bereits Teil eines stabilen Clusters sind können diesen nicht mehr verlassen. Die Simulation stoppt sobald alle Teilchen teil einer stabilen Insel sind. Die resultierenden Inseldichten wurden mit den experimentellen Ergebnissen verglichen. Es stellte sich heraus, dass eine kritische Keimgröße von $i = 7 \pm 1$ sich mit den Experimenten deckt. Weiters wurden die Inselgrößenverteilungen mit den aus den Bildern des AFM (Atomkraftmikroskop) erhaltenen verglichen und eine qualitative Übereinstimmung festgestellt.

Zudem wurde eine computergestützte Simulationstechnik namens Quasicontinuum Monte Carlo (QCMC) angewandt, um den Effekt der Abstoßung von Adsorbaten an den Inselrändern berücksichtigen zu können, welcher als Grund für die Ausbildung der 2D-Gasphase im Vakuum gesehen wird.

Contents

List of Figures

1 Introduction

Organic thin films as active layer in organic electronic devices like organic lasers, light emitting diodes, thin film transistors, and organic solar cells have been extensively studied in recent years [1, 2].

Especially semiconducting oligomers such as pentacene as example of an oligomeric acene show promising behavior and are already used to build devices [3]. The advantage of molecules that exhibit a rod-like shape is that they can be arranged normal or parallel to the surface, which is beneficial for building devices since the optical behavior and charge carrier properties depend on the orientation of the molecules [4].

Another rod-like molecule studied frequently is para-hexaphenyl (6P), which is an oligomeric phenylene and an interesting material to study conjugated organic semiconductors [5]. Different devices have already been realized using this molecule. It exhibits blue electroluminescence [6, 7], and is therefore used to build blue light emitting diodes [8]. Moreover, solar cells [9] and organic field effect transistors [10] using that molecule have been studied.

However, in order to build such devices one has to have an understanding of the film formation and therefore of the growth parameters. These parameters are on the one hand the chemical as well as the structural composition of the surface, and on the other hand the molecular flux to the surface and the interface temperature. Moreover, the initial stages of thin film growth are of special importance, since growth behavior of thicker layers are drastically influenced by the interaction between the first layer of the film and the substrate [11]. In fact, there is still a considerable lack of understanding of the processes involved in forming ultra-thin films when it comes to organic molecules [12].

In this context the question arises whether the well developed theories of inorganic crystal growth can be applied also to the growth of large organic molecules. In order to get a better understanding of these systems, the example of para-hexaphenyl (6P) on amorphous muscovite mica(001) has been frequently studied by several groups [11,13–15]. As shown quite recently [16], the system exhibits behavior that cannot be described any more simply by diffusion limited aggregation theory (DLA), but shows effects that can be credited to the special shape of the molecule.

Moreover, there is the quite peculiar behavior of the molecules on the mica surface of forming a pronounced bimodal island size distribution, which will be the main topic of this thesis. As it can be seen clearly in the paper of Potocar et al. [11], for particular growth parameters besides the big islands a lot of small islands exists, which is a quite puzzling behavior.

The aim of this work is to use the widely applied technique of Kinetic Monte Carlo (KMC) simulations to provide further insight into the effects and mechanism causing this behavior. Continuous exchange of information between experiment and simulation shall be carried out with the objective of getting the most efficient outcome.

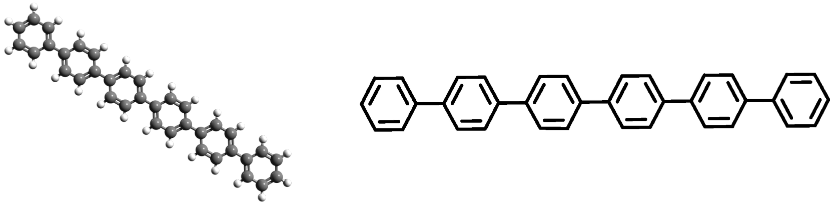
The next section describes the experimental investigations in detail. Furthermore, it tells the story that started with initial ideas of what could cause the effect of bimodal island size distributions, which ended with the final revelation of the physical reasons. Moreover, the problem definition of what the simulations shall do in detail are given.

In sec. 3 the foundation for understanding the theoretical models of nucleation and growth as well as of the computational simulation techniques is given. After that sec. 4 describes the implementation of the software that was written for this work and provides some test results to provide the reader with further information using parameter studies. The results of the application of the simulations to the system 6P on mica and a detailed discussion are given in sec. 5, after which the work is concluded.

2 Bimodal Island Size Distribution: Ideas and Experimental Investigations

The following section uses the paper of Tumbek et al. [17] as a basis because it contains crucial information to understand the issues studied in this thesis. However, concerning some of the aspects more details will be given. For theoretical descriptions the author may be referred to the section on theory of this work (sec. 3).

2.1 Details on Materials



(a) (b)

Figure 1: Single 6P molecule in planar configuration (a) and the corresponding structural formula (b)

To begin with, some details on the used materials are provided. Para-hexaphenyl (p6P, “sexiphenyl”, $C_{36}H_{26}$ [18]) is composed of six phenyl rings and has a rod-like shape with Van der Waals dimensions of 2.85 x 0.35 x 0.67 nm^3 [19] (see fig. 1). The bulk crystal structure is monoclinic and shows the pattern of the so-called “herringbone structure” [5, 20]. The band gap is $E_g \approx 3.1$ eV [21], which leads to emission of blue light. From now on, the abbreviation “6P” will be used for the molecule.

As for the substrate muscovite mica ($KAl_2(AlSi_3O_{10})(OH)_2$) was used, which has a perfect cleaving plane (001) [22]. In fact, this makes it an appropriate substrate material because of the reproducibility of this perfect crystal plane. The growth of 6P on this surface has been studied extensively (see the work of Frank et al. [13] and references therein).

However, the freshly cleaved surface is highly anisotropic, which leads to different properties compared to a surface where this anisotropy is broken. One of the ways to perform that modification is to use ion bombardment that destructs the ordered surface and causes amorphous structures [13].

2.2 Experimental Procedure

Even though this thesis is of theoretical nature, it may be of interest how the surroundings were set up by the experimentalists. The investigations were performed using an ultra-high vacuum (UHV) chamber in which ultra-thin films of para-hexaphenyl were deposited on mica(001) samples by physical vapor deposition (PVD) from a Knudsen cell made of glass. The mica samples had a dimension of 10 x 10 x ~ 0.01 mm^3 and were produced by cleaving with an adhesive tape, after which they were put into the chamber immediately. The temperature of the sample could be controlled using a NiCr-Nr thermocouple at the back of a resistively heated steel plate that was connected to the sample by tantalum wires. The film thickness was controlled by a quartz microbalance positioned next to the sample. Moreover, ion bombardment to modify the mica surface was done for about 10 min by Ar^+ sputtering.

Measurement methods include thermal desorption spectroscopy (TDS), Auger electron spectroscopy (AES), and x-ray photoelectron spectroscopy (XPS) for in-situ investigations. Moreover, atomic force microscopy (AFM) measurements could be performed ex-situ as imaging technique.

However, as the experimental challenges themselves are not subject to this thesis, detailed information on the ways to verify the results of the measurement techniques has to be gathered elsewhere (see e.g. [13]).

2.3 Difference Between Growth on Freshly Cleaved and Amorphous Mica

There are AFM measurements shown in fig. 2, which point out in a drastic way the difference between the two growth behaviors for the sputtered and freshly cleaved mica(001) surfaces. The parameters at which the deposition has been done are given in the figure.

Nonetheless, in both cases there are smaller islands visible besides bigger ones. If there are two sizes at which clusters are stable, it would mean also that islands of a size in between those two sizes are either for some reason unstable, or that the two growth events happen independently from each other. In established nucleation theory there is nothing that would point out such a behavior as long as homogeneous nucleation can be expected. Sec. 3.1 describes the analytical theory of submonolayer growth, which does not offer any parameter that could trigger this bimodal size distribution.

Anyway, one of our first ideas what could cause this peculiar fashion was that maybe the movement speeds of 6P on the surface vary when arriving at the surface depending on the orientation of the molecule with respect to the surface plane. Some computational studies were made where two types of particles were assumed to diffuse at different speeds. Furthermore, a program that studies the results of ballistic movement on surfaces was tested, where the movement was superposed with a random walk motion that accounts for statistical perturbation by thermal noise. However, all these studies did not give any

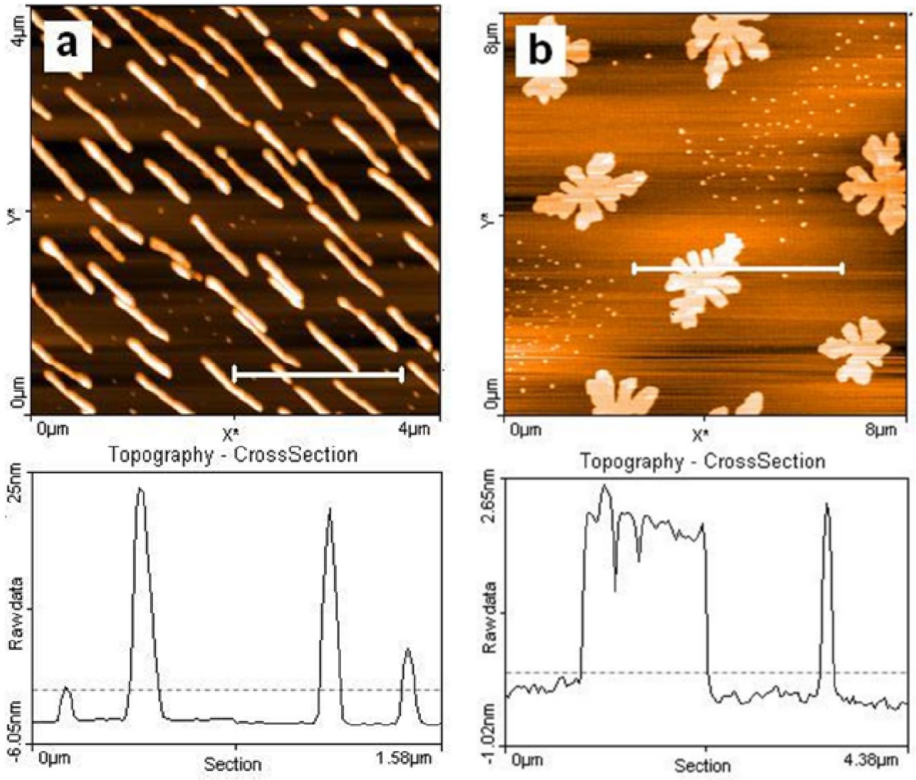


Figure 2: Comparison of the submonolayer growth of 6P between the case of (a) freshly cleaved mica substrate and (b) sputtered mica substrate. There is a huge difference in growth behavior, but in both cases exist smaller islands between the bigger ones. The locations of the cross sections below each picture are indicated in the images. See sec. 2.3 for further details. [17]

hint towards a bimodal island size distribution and are therefore omitted here.

The key to understand the behavior finally turned out to be of experimental nature. Although this work focuses on the sputtered case, it is useful to point out the freshly cleaved system's behavior in order to understand the role of the wetting layer.

2.3.1 Freshly Cleaved Case

When looking at the morphology in fig. 2(a), one sees that there is needle growth in the case of the non-sputtered substrate, which was interpreted by a strain-induced agglomeration of smaller crystallites in an earlier work on this system [15]. Further investigations pointed out that the needles are composed of flat lying molecules [13].

Anyhow, to get further information about the energetic properties of the thin film, TDS measurements have been made. If there is a wetting layer, one expects desorption of the particles belonging to that layer at a higher temperature compared to the ones of the islands, because the particles are then more strongly bound to the surface than to each other. In contrast, molecules that are part of the islands are then expected to be more weakly bound and therefore to desorb earlier during the increase of temperature. This can be seen in measurement curve (a) of fig. 3. The big peak corresponds to the particles desorbing from the needles, the smaller one at higher temperature can be interpreted as the signal from the wetting layer. The area below the signal is actually proportional to the amount of material that has been desorbed, which is in agreement with the believe that the bigger peak comes from the needled. The cross section in fig. 2(a) shows a height of up to approximately 30 nm for the big islands in needle shape, whereas the smaller ones exhibit less height.

The existence of a wetting layer leads to the conclusion that the bonding of the 6P molecules to the substrate is quite high, which is also supported by the fact that they are lying flat on the surface in contrast to the case of standing molecules as later mentioned in the case of the sputtered mica substrate.

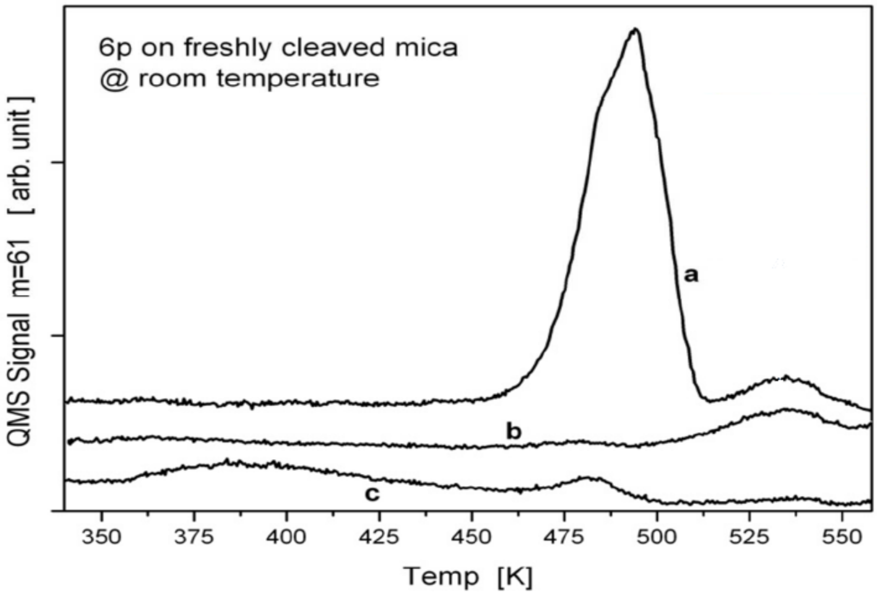


Figure 3: Thermal desorption spectra of 6P on freshly cleaved mica(001). The first curve (a) shows the spectrum for an amount leading to 1.6 nm mean thickness. For both curves (b) and (c) the deposited amount leaded to 0.32 nm mean thickness, which corresponds roughly to one monolayer of lying particles. The difference, however, is that measurement (b) was done in-situ directly after deposition and the sample used in measurement (c) was exposed to air after which it was reinstalled into the chamber. One can clearly see that the peak at approximately 535 K which points to the existence of a wetting layer is gone only for case (c). [17]

Furthermore, curve (b) of fig. 3 shows the signal which results from less coverage, where the amount is not enough to form needles, but the signal from the wetting layer persists. Moreover, fig. 4 shows the corresponding AFM picture which was made ex-situ.

Surprisingly, in contrast to the expectation that there should be more or less a uniform film since a wetting layer should be flat, one can see that there are islands. The size of the islands on this picture is similar to the small ones in fig. 2(a). Note that the AFM picture dimensions are different (4 μm in fig. 2(a) versus 1 μm in fig. 4). It turned out that the sum of molecules that are incorporated into all of the islands agrees with the amount of material a wetting layer would be composed of. Consequently the wetting layer is gone, which agrees again with curve (c) of fig. 3.

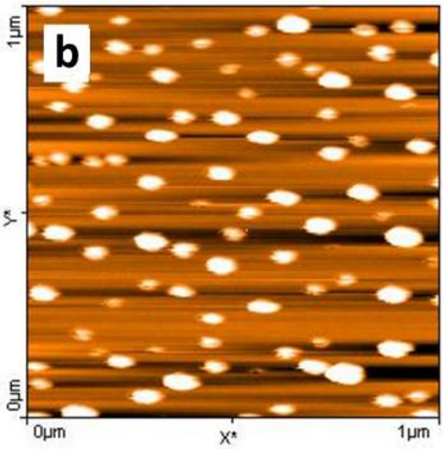


Figure 4: AFM measurement of 6P on freshly cleaved mica for an amount deposited corresponding to one monolayer lying molecules at room temperature. [17]

From this one can conclude that venting the chamber changes the energetics and the resulting morphology drastically. At the time, we believe water to be the substance that leads to dewetting and hence to the formation of these islands. For some reason, the bond energy between the 6P molecules and the substrate becomes weakened.

2.3.2 Sputtered Case

In this case the morphology is clearly different, since no needles form at all and big dendritic islands besides smaller quite compact ones exist. The dendritic shape is probably caused by the isotropic surface due to amorphization by sputtering, because in this case one can expect these islands to form according to diffusion limited aggregation (DLA). All islands consist of standing molecules as verified by the cross section below the picture (fig. 2(b)), which shows a height of about the Van der Waals dimensions of the longest axis of 6P (see sec. 2.1). Therefore, the interaction with the substrate seems to be weaker.

Moreover, it should be stated that the appearance of smaller clusters together with the bigger ones only occurs at certain coverage ranges:

No big islands: Below a particular coverage ($\theta < 0.04$ ML) no big islands exist, but small ones can be found already in great number, depending on the deposited amount. Snapshots for this regime can be seen in fig. 5a-c.

Nucleation regime of big islands: The large island number continuously increases between θ =0.04 ML and θ =0.15 ML, where both small and big islands exist. Probably in this range nucleation of the big islands occurs and therefore the number increases drastically. Furthermore, the size of the small islands basically stays the same, but they cannot be observed near to the rim of the big ones, forming so-called "denuded zones". This also leads to a smaller number density of small islands on the surface.

Aggregation regime for big islands: In the range between 0.1 ML $< \theta <$ 0.6 ML the bigger islands grow, which can be attributed to the aggregation regime for these islands in the manner of DLA.

Coalescence regime for big islands: Above a coverage of approximately 0.7 ML coalescence of the big islands starts, and small ones cannot be found any more.

Note that the theory concerning the different regimes of nucleation and growth can be found in section (sec. 3.1.4).

In fact, one cannot easily use the knowledge about the freshly cleaved case to explain the bimodal size distribution on the sputter modified mica(001) surface as well, because the existence of a wetting layer in the same manner could not be experimentally detected. Rather there is a certain coverage regime in which bigger islands do not form yet, which we interpret as being a 2D-gas phase of particles that move around on the surface and for some reason do not form islands yet. In the same way as for the freshly cleaved case, be believe that water changes the physics of the processes happening on the surface so that this 2D-gas phase vanishes due to nucleation of islands. The coverage of this 2D-gas, however, is smaller compared to the wetting layer (1 ML) in the freshly cleaved case.

Consequently, after putting this information together, the following conclusion can be drawn about the origin of the bimodal island size distribution on sputtered mica:

After exposing the samples to air, adsorbate induced subsequent nucleation of the remaining monomers on the surface happens due to influence of water as adsorbate on the surface.

In order to trigger this subsequent nucleation behavior the adsorbate has to either lower the diffusion energy or to lower the activation barrier for nucleation. Moreover, because there are still enough monomers to nucleate the smaller clusters in the case when bigger islands already did nucleate, there must be some barrier for the particles to attach to these existing islands. In fact, there is plenty of time for the (as we believe) fast diffusing molecules to find existing islands to attach to between stopping the particle flux and the moment of venting the chamber.

The reason for this 2D-gas phase to persist may be a barrier for attachment to existing islands, which was recently shown to be the case for this system [16]. In that

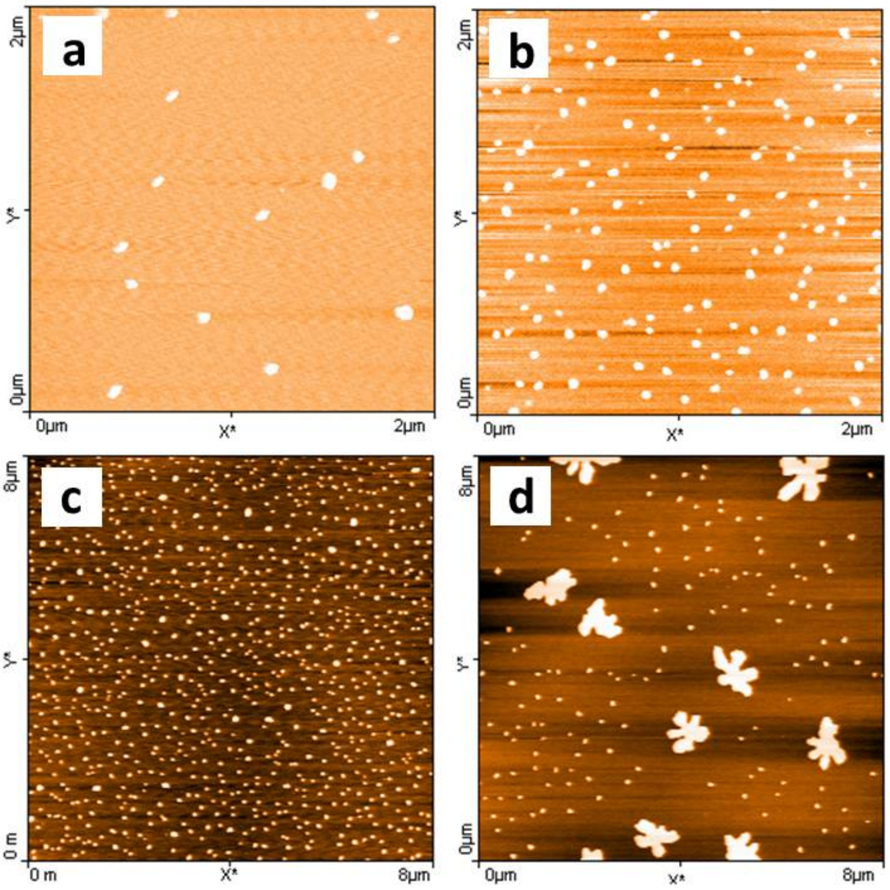


Figure 5: Example of different coverages for 6P on sputtered mica(001) substrates: (a) 0.014 ML, (b) 0.03 ML, (c) 0.036 ML, (d) 0.04 ML. Deposition rate (flux): 0.08 ML/min, substrate temperature: 400 K. The first bigger islands can be observed at a coverage of (d) 0.04 ML [17]

paper the evolution of the big islands' sizes has been followed and compared to the Venables equation (eq. 16), which describes the number density of islands for diffusion limited aggregation (DLA), and has been found not to fit to this theory. Rather an equation for attachment limited aggregation [23, 24] was found to fit the data and gave a critical island size of $i = 7 \pm 2$.

Because particles are hindered by a barrier to attach to existing islands, we believe that a quasi-equilibrium between the 2D-gas phase and the islands develops. For reasons of thermal excitations in principle there could be a non-zero probability that particles can overcome the energy of the bond to their surrounding neighbors and therefore escape from the island, which leads to a vapor pressure. Hence, there is a continuous exchange between the particles of the 2D-gas phase and the ones of the islands.

Furthermore, the attachment barrier can also be seen as the reason why nucleation does not start until a relatively high coverage is reached, since the barrier also influences the capture number of monomers and subcritical clusters [24].

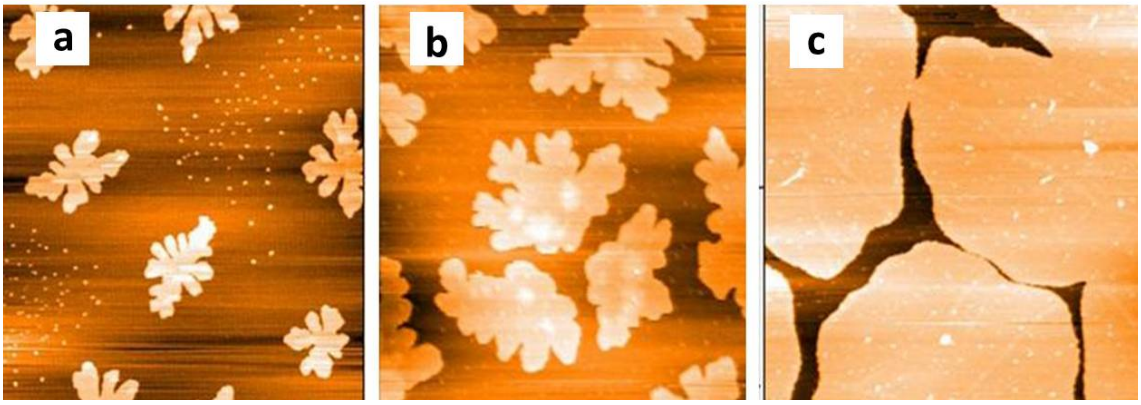


Figure 6: Example of different coverages for 6P on sputtered mica(001) substrates: (a) 0.18 ML, (b) 0.47 ML, (c) 0.94 ML. Deposition rate (flux): 0.1 ML/min, substrate temperature: 400 K. The denuded zones around the bigger islands can be seen clearly in case (a), where for case (b) and (c) no small particles can be observed any more. [17]

Finally, as another proof of concept the existence of denuded zones around the bigger islands can be understood as well by the idea of the 2D-gas phase. When the particles suddenly start moving faster because of the adsorbate, they will need some time to nucleate, because the critical island size can be expected to have a quite high value due to the low number density of the small clusters. Even though we call these islands "small", they consist of thousands of molecules, which cannot result from low critical island sizes (see sec. 5 for justification). Furthermore, nucleation takes time. During this time, particles that are near to existing islands are much more likely to be incorporated into the islands instead of nucleating new ones. Only for particles with sufficient distance to the existing clusters nucleation can be expected.

2.4 Problem Definition for the Simulations

As one can see on the AFM images, the size of the big islands is on the order of 1 x 1μm^2, which means that even single islands contain about 10^6 molecules. Doing simulations with that amount of particles is far beyond the current computer capacity, as one usually wants to study cases where even much bigger areas are needed to have space for more islands.

However, the size of the small islands is much less, as one can check for example in fig. 5(a): An island with dimension of about 100x100 nm^2 contains about 10000 particles. Among established simulation techniques, there are only Kinetic Monte Carlo (KMC) simulations that can possibly meet these sizes nowadays (see sec. 3.2). Moreover, when settling for doing simulations of the formation of the smaller islands, we expect to extract interesting physical properties even though only a small part of the whole surroundings on the surface can be taken into account. Therefore the objective of the desired simulations can be defined:

Simulate nucleation and growth of the adsorbate induced subsequent nucleation process of 6P on amorphous mica(001) by starting with an initial coverage of particles on the surface and letting the system evolve until all monomer particles are incorporated into islands

Indeed the system has to be analyzed to make reasonable assumptions to carry out those simulations with regard to the possibilities the technique offers us and to the knowledge we have about the system.

Concerning the anisotropic nature of the particles, we currently only know about one group that performs KMC simulations with taking into account the rod like shape of such molecules [25]. Nearly all studies treat particles as point like entities, position them on a lattice, and let them hop with certain probabilities to neighboring places.

Here the 6P molecules will be treated as point-like as well, because we do not expect to see a direct influence on statistics for the following reasons:

- As we know from the AFM measurements, the molecules are oriented upright in the islands and therefore occupy single adsorption sites, building up a lattice in the 6P(001) plane. Hence, when looking "from top" onto the islands the particles do not appear to be rod-like.

- The distance between the different islands is on the order of micrometers, separating them so far that we cannot expect the elongated shape to have a great influence, since the size of the long axis is on the order of nanometers.

- On the sputtered substrate the island distribution is isotropic and the shape of the big islands is still of dendritic nature as for the case of point-like entities in the theory of metal homoepitaxy.

If one considers the system in more detail, considerably more about all the energy barriers has to be known, making it necessary to do accurate calculations in advance. Anyway, using this information for large-scale KMC simulations would most probably lead to severe bottlenecks in the calculations without being for sure of physical significance.

There is a paper by Bales and Chrzan [26], where they did simulations of submonolayer growth of point-like entities on a square lattice using an algorithm that suppresses dissolution of already existing clusters. This approach is used here and extended by including the critical cluster size i as additional parameter, similar to the ideas of Li and Evans [27]. Sec. 4 deals with the implementation of this approach.

A square lattice is used and every molecule can occupy one site on this lattice. Hopping into a second layer is not allowed, since we do not regard molecules hopping on to of each other. In contrast to Bales and Chrzan, we do not use the so-called solid-on-solid model (see sec. 3.3.8). In this model, one takes into account also the binding energy to the surface besides the one to the lateral neighbors when calculating the hopping rates to neighboring places. For sure this is crucial for single atoms diffusing along an island rim, since they will always be influenced by the surface as well, even though the lateral binding energy may be higher. When 6P molecules are in their upright orientation located at the rim of an island, they will most probably be very weakly influenced by the surface because only one H-atom is in contact to the surface species. In contrast, a lot of C-atoms will make a much bigger contribution to the bonding between parallel oriented 6P molecules. In fact, we do not know exactly how diffusion along the rim of an island happens in this case, but at least studying the energetic stability of islands is possible without having this information.

Therefore, according to transition state theory (sec. 3.1.1), the diffusion rate of monomers can be described by

$$h_1 = \nu \exp(-Q/kT) \tag{1}$$

where Q is the energy barrier for surface diffusion, ν is the hopping frequency, k is the Boltzmann constant, and T is the substrate temperature.

As long as an island has a size smaller than or equal to the critical nucleus size i, particles are still treated as monomers. We believe this is justified by the fact that the island density on the surface is very small, because if these subcritical-nuclei would have already a binding energy E_i significantly above the thermal energy kT, one would expect a much higher number of islands.

At the time when a cluster reaches a size higher than the critical size, aggregation is irreversible and only edge hopping is allowed, suppressing detachment of particles form the island. The rate of movement along the rim is given by

$$h_n = \nu \exp(-nE/kT) \tag{2}$$

where n is the number of in-plane nearest neighbors and E is the in-plane nearest neighbor binding energy.

The adsorbate induced subsequent nucleation is not influenced by a flux of impinging particles, as it is the case in most computational studies. Nevertheless, the program supports it since it was written at the time when the origin of the bimodal island size distribution was not yet revealed. Desorption of particles is omitted, since there is no evidence that this happens at the given temperatures.

To sum up, the program shall perform the following steps:

1. Deposit an amount of particles on the surface that corresponds to a reasonable coverage that has been determined experimentally.

2. Start simulation, which means that mainly monomer diffusion processes are performed until stable clusters form and grow.

3. Stop the simulation when no monomers are left, meaning that the final number of clusters would not change any more except for unlikely coalescence.

4. Calculate statistics (island number density, island size distribution).

To obtain reasonable statistical data, a lot of runs probably have to be performed. This statistical data can then be used to compare with the experiments, and hopefully provides information about the critical island size involved, when one compares the island densities of simulations and AFM measurements.

The next sections describe the underlying theory and the implementation of the simulation program. For the results of the simulations applied to the problems of this section, see sec. 5.

3 Theoretical Foundation

In this section, the theoretical background for this work is provided. After a short introduction, the classical theory describing the process of nucleation and crystal growth during molecular beam epitaxy will be given, which is the necessary basis for data interpretation both for experimental and simulated results. Next, a brief overview of various simulation techniques is given, providing the principles of the computational treatment of such systems. Finally, the necessary basis for understanding in detail the simulation principle of Kinetic Monte Carlo simulations is provided.

Using molecular beam epitaxy (MBE), thin films of various materials can be grown on a substrate surface. This is typically done in ultrahigh vacuum, while maintaining the substrate at an elevated temperature. Mechanical shutters in front of the beam sources provide control over the beam flux. The crystallization of such films happens under the influence of reactions between the impinging species and the substrate, where the relative rates of arrival and surface migration of impinging particles greatly influence the morphology of the created epilayer [28].

Important to state is that most available literature in this field deals with growth of point-like entities, which is justified at least for atoms. There is no theory taking into account the anisotropy of molecules. There is no evidence that this anisotropy leads to significant changes in the physics, so that the established theory would turn out to be wrong. Consequently, in this work, 'classical' nucleation and growth theory is used. Expected deviations from that theory for our case of organic molecules will be stated if necessary.

3.1 Analytical Treatment of Nucleation and Growth of Submonolayer Films

To begin with, the basic physical parameters are defined in order to specify degrees of freedom of the system.

Coverage θ: There is a maximum number of particles that can form one single layer at the surface. Therefore the coverage can be given in units of monolayers (ML).

Flux F: Particles arrive at a rate F in units of monolayers per unit time (ML/s) at the surface from the incident beam. Alternatively units of particles per site per unit time may be used. When using the symbol R, usually the rate is related to the area L^2, giving the dimension $[L^{-2}t^{-1}]$.

Temperature T: The substrate is held at constant temperature, excluding any temperature gradient. This makes sure that diffusion is isotropic. Occasionally, the inverse product with the Boltzmann constant $\beta = 1/k_BT$ will be used.

Particle binding energy E_x: Particles that come near each other or to already existing clusters may aggregate, resulting in a preferred energy state of the system.

This energy is the barrier the particle has to overcome in order to move to another site. The index x indicates the various possible kinds of barriers there may be, including also the diffusion barrier of monomers, since there is also an interaction with the substrate.

Cluster binding energy E: The binding energy of whole clusters is the sum of all binding energies between the individual particles, plus the binding energy between the particles and the substrate.

Vibrational frequency ν: Depending on the binding energy, the particle can move due to thermal vibrations. Consequently, diffusive movement of particles is influenced both by this frequency and the barriers (see sec. 3.1.1 below), leading to the diffusion constant D_x in dimensions of area per unit time. x indicates again the various types of movements, for which the diffusion constant differs depending on the barrier.

Fig. 7 shows a schematic representation of possible processes happening during homoepitaxy on a flat surface. The rate at which the particles move on the surface depends on their surroundings. Different barriers for movement are indicated.

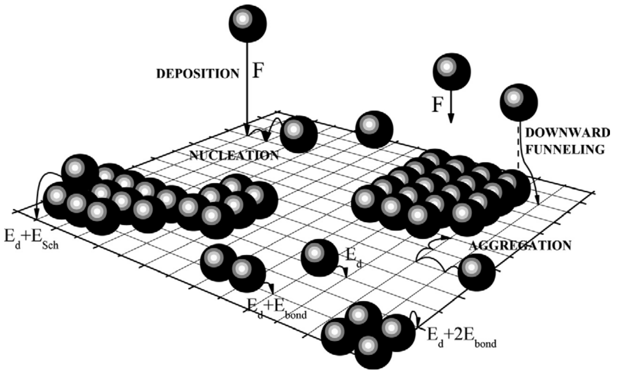


Figure 7: The sketch shows the basic processes of crystal growth on a square lattice, which corresponds to the case of metal(100) homoepitaxy [29]. Various activation barriers for diffusion are indicated, where E_d is the barrier of adatom diffusion on a terrace; E_{bond} is the barrier for overcoming in-plane nearest neighbor bonds, and E_{Sch} is the Ehrlich-Schwöbel barrier [30, 31].

3.1.1 Transition State Theory (TST)

Imagine a potential energy surface a particle is exposed to on a surface, originating from the interactions with the individual surface species. In order to find out the microscopic rates at which different processes happen, (harmonic) transition state theory (TST) is widely used [32, 33]. A “process“ here could be a particle moving from one surface site to another. This approximation says that the rate Γ for a process to happen is simply the attempt frequency ν, which corresponds to the lateral vibrations of the particle in its state, times the Boltzmann probability of being at the saddle point of the potential energy surface between two (bound) surface states:

$$\Gamma = \nu \exp\left(-\frac{E_d}{k_B T}\right) \qquad [s^{-1}] \tag{3}$$

where E_d is the activation energy barrier for diffusion, k_B is the Boltzmann constant and T is the temperature. Note that all irrelevant vibrations are neglected, giving an expression simple to use, since the parameters ν and E_d are usually possible to determine.

Let a particle move along a certain direction on the 2D-surface. Fig. 8 shows a one-dimensional sketch of what the cross-section of such a potential energy surface might look like. According to eq. (3), the rate does not depend on the energy level of the final state. Consequently, the probability that a particle that is near the step, drops into the lower area on the right by moving one step to the right, is the same as moving left, since the barrier is in both cases E_0. On the contrary, the energy barrier for e.g. particles trapped in the lower region for escaping is quite high compared to the smaller barriers for movement in this region. This could be the case for an island, where moving along the rim (E_1) may be much more probable than detaching from the island (E_{det}). Using these ideas, the rates for diffusive transport for all the possible paths on a surface can be determined. This can be done by either using ab initio calculations, or numerically by performing DFT (density-functional theory) calculations or MD (molecular dynamics) simulations (see sec. 3.2) [34].

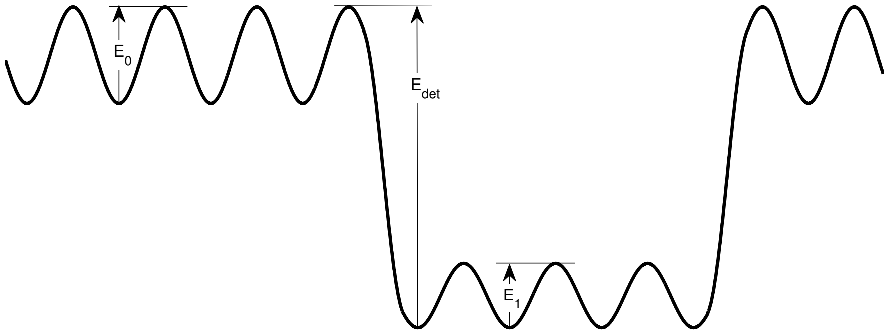


Figure 8: One-dimensional example of potential energy distribution on a surface. In TST, the rate of a particle trapped in one of the cavities for leaving it only changes with the barrier height and therefore does not depend on the final cavity depth. E_0, E_{det}, and E_1 are barriers of particles for transitions to other surface sites (see sec. 3.1.1)

3.1.2 Thermodynamic Considerations

When looking at eq. (3), one finds two special cases for the value of the temperature:

- $T \to 0 \Rightarrow \Gamma = 0$: For very low temperatures, the system will freeze out and no process will happen.
- $T \to \infty \Rightarrow \Gamma = \nu$: For very high temperatures, all processes will happen with the same rate in opposite directions, leading to thermal noise and consequently to thermodynamic equilibrium. However, at some critical value for T, there will be phase transitions to the liquid or gas state, making these considerations senseless.

Crystal Growth

When weakening the first condition so that certain processes happen occasionally, but others still do not, the system forms non-equilibrium structures. This is the case for crystal growth, which is therefore a non-equilibrium kinetic process. If the temperature changes, the probability for individual processes to happen changes as well. Consequently, the system will take different paths of processes until the final state is reached [35]. However, the final state of the system is an arbitrarily chosen point in time, since this state is not necessarily the stable one. Kinetics determine this state, which means that when waiting for a long time, the system may reach the state of lowest energy, but this timescale may be orders of magnitude longer than the one of interest. In fact, the intermediate states between $t = 0$ and $t = t_{max}$ depend strongly on the rates of the various processes, even if the final state may be the same. One can categorize the types of processes in the following way:

Kinetically forbidden processes: There are processes that may be possible in principle, but they will in fact never happen. This can include dissociation of molecules, or desorption if the system has the property that the barriers involved are too high to induce the process in considerable time. For this reason, they may be neglected both in theory and simulation.

Processes in local thermodynamic equilibrium: For certain parts of the physical system, some aspects of equilibrium situations may be locally valid. An example is the adsorption-desorption cycle. Of special interest is the continuous growth and decay of unstable clusters.

Kinetically limiting processes: Processes that happen occasionally are in fact the ones of interest, because they shape the morphology of a surface.

The latter case is of special interest because these processes turn out to be time-consuming to simulate (see sec. 3.2) and play an important role in nucleation theory.

3.1.3 Concept of Critical Nucleus Size

Consider a perfect terrace on the surface, where only homogeneous nucleation can occur. The rate-limiting process here is the formation of stable clusters, which has to be analyzed using energetic considerations.

Let particles be denoted as A growing on a substrate B. The energetic stability of a flat (2D) cluster containing j particles, governed by the surface-to-volume ratio, can be described by the change of the Gibbs free energy $\Delta G(j)$ [35]. On the one hand, when particles aggregate, energy is gained when the system is supersaturated (chemical potential change $\Delta\mu > 0$). On the other hand, forming a cluster also costs energy because of surface and edge formation:

$$\Delta G(j) = \underbrace{-j\Delta\mu}_{\text{volume energy gain}} + \underbrace{j(\gamma_A + \gamma^* - \gamma_B)\Omega^{2/3}}_{\text{surface energy change}} + \underbrace{j^{1/2}\sum_e C_e\gamma_e}_{\text{edge energy}}$$

where

γ_A, γ_B ... surface energies per unit area of particles (A) and substrate (B)
γ^* ... interface energy per unit area between particles and substrate
Ω ... volume of particles A
γ_e ... edge energies per unit length
C_e ... edge lengths

Note that when a cluster grows, the interface energy γ^* substitutes the substrate surface energy γ_B at the newly occupied site, which is why γ_B contributes negatively. Both surface and edge formation energies lead to a barrier of nucleation, which has to be overcome in order to form a stable cluster. The *critical cluster size* i and the corresponding critical energy $\Delta G(i)$ can be easily obtained from $\partial(\Delta G(j))/\partial j = 0$:

$$\begin{aligned} i &= \frac{(\sum_e C_e\gamma_e)^2}{4\left(\Delta\mu - (\gamma_A + \gamma^* - \gamma_B)\Omega^{2/3}\right)^2} = \frac{X^2}{4(\Delta\mu')^2} \\ \Delta G(i) &= \frac{(\sum_e C_e\gamma_e)^2}{4\left(\Delta\mu - (\gamma_A + \gamma^* - \gamma_B)\Omega^{2/3}\right)} = \frac{X^2}{4\Delta\mu'} \end{aligned}$$

In the denominators, one can identify a difference of potentials, which was therefore renamed $\Delta\mu'$. Choosing arbitrary values for these relations, the qualitative behavior can be drawn, which was done in fig. 9. Depending on the value of $\Delta\mu'$, a formed cluster is either unstable, or may become stable when exceeding the critical size i. However, it must be stated that the concept of the critical size has to be treated more precisely because the derivative of the Gibbs free energy only gives us information about whether growth or shrinkage is more likely to occur. It could very well be that due to local fluctuations, a cluster of size $j > i$ dissolves. Moreover, clusters smaller or equal to the critical size only form, if they are able to overcome unfavored energy states, which has to happen to reach the critical size and hence leads to stable structures.

There is a case for the value of $\Delta\mu'$ for which not only a stable formation is possible, but also the critical size is so small that already two particles are stable ($i = 1$). This is called the *irreversible aggregation* mode. Following this concept, the case for which $i > 1$ is called *reversible aggregation* mode, which accounts for the fact that clusters may decay unless they reach the critical size.

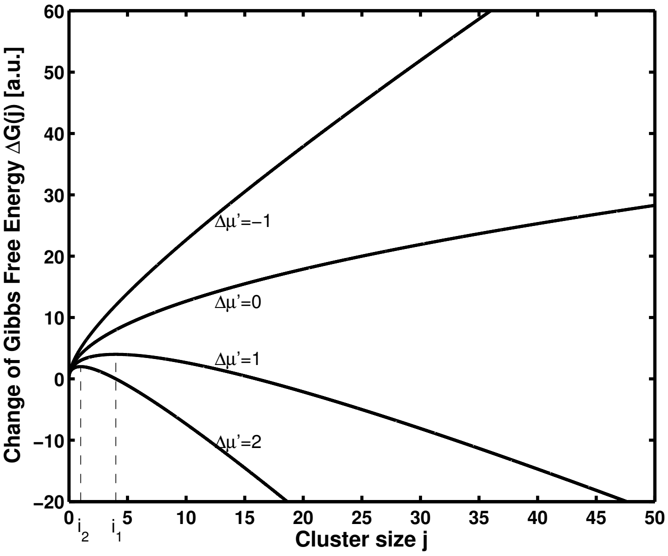


Figure 9: Sketch of the change of the Gibbs free energy $\Delta G(j)$ for nucleation of 2D clusters, where $X = 4$. The units are arbitrary for both axis. Depending on the difference between the chemical potential change and the surface energy $\Delta\mu' = \Delta\mu - (\gamma_A + \gamma^* - \gamma_B)\Omega^{2/3}$, the formation of clusters can either lead to consequent dissolution ($\Delta\mu' \leq 0$), or to stable clusters ($\Delta\mu' > 0$), for which the critical clusters sizes i_1 and i_2 are indicated. See sec. 3.1.3 for further discussion. [35]

As mentioned above in sec. 3.1.2, the continuous growth and decay of subcritical clusters is a process in local thermodynamic equilibrium. The equation describing the concentration of clusters of size $j \leq i$ is given by the so-called Walton relation [36]:

$$\frac{N_j}{N_0} = \left(\frac{N_1}{N_0}\right)^j \sum_m C_j(m) \exp\left(\frac{E_j(m)}{k_B T}\right) \tag{4}$$

where

N_j	... density of clusters of size j
N_1	... density of monomers (cluster of size 1)
N_0	... number of surface sites per unit area
m	... denotes the type of the cluster of size j
$C_j(m)$	... statistical weight of cluster type m of size j
$E_j(m)$	... binding energy of cluster type m of size j

In short, it says that the relative concentration of a cluster configuration m raises exponentially with its binding energy. When the temperature is quite low, which is a good assumption for growth conditions, the sum in eq. (4) reduces to one element, because then only the cluster with the highest energy is important.

3.1.4 Rate Equation Approach

There is a traditional analytical formulation for describing the cluster size evolution during epitaxy, originally developed in the 1960s by Zinsmeister [37, 38] and Venables [34, 35, 39]. In fact, it is a rate equation analysis based on mean field theory, treating the change of monomer density and island density as a function of time. Although proven to be slightly wrong in predicting island size distributions [40], it is a quite instructive theory, giving results that are describing the systems correctly in principle and is therefore widely used.

In addition to the physical parameters introduced at the beginning of sec. 3.1, additional variables are used, which are mainly of mathematical nature, though chosen to reflect physical behavior resulting from the nature of the system. The mentioned papers above are taken as a basis for the following, where a slightly different formalism will be used in the style of Evans [29, 40].

Hopping rate h: Monomers hop at a rate h $[s^{-1}]$ between adjacent sites on the surface, which can be determined by TST as mentioned above (sec. 3.1.1). For simplicity of the analytical treatment, only one single hopping rate is considered here, assuming the surface to be a uniform array of adsorption sites. The relation of the diffusion constant D $[m^2 s^{-1}]$ to the hopping rate is simply $D = \alpha^2 z h$ where z is the number of (free) nearest neighbor sites and α is the distance between the sites.

Critical cluster size i: The number of particles forming a cluster determines the stability of that cluster. Clusters of size greater than i are likely to grow, whereas the ones with size smaller or equal to i tend to decay more probably (see sec. 3.1.3).

Binding energy E_i: The binding energy of a critical cluster of size i is by definition positive: $E_i \geq 0$.

Monomer density N_1: The mean number of monomers per lattice site. Monomers are particles with no in-plane nearest neighbors, interacting only with the underlying substrate. Consequently, these particles are likely to have the highest diffusion constant when particle-particle interactions are stronger than particle-substrate interactions. One can look at the global values of the density by averaging, as well as at the local value e.g. around clusters.

(Stable) island density N: The mean number of islands that are stable due to exceeding the critical cluster size i. For simplicity we will consider the islands to be point-like in this section and therefore neglect their spatial extent.

Density of critical clusters N_i: The mean density of critical clusters of i atoms.

The main idea is that the monomer number on the surface increases due to flux and decreases due to nucleation and aggregation. On the other hand, the change of islands is just due to nucleation. This results in the

Rate Equations

$$\frac{dN_1}{dt} \approx F - (i+1)K_{nuc} - K_{agg} \tag{5}$$

$$\frac{dN}{dt} \approx K_{nuc} \tag{6}$$

In order to get expressions for the nucleation rate K_{nuc} and the aggregation rate K_{agg}, the quasi-equilibrium Walton relation is important, which is a simplified version of eq. (4):

$$N_i \approx c_i \exp[\beta E_i](N_1)^i \tag{7}$$

It says that the number of critical clusters of size i raises exponentially with the binding energy E_i of the cluster and with the monomer density to the power of i. The constant c_i is a parameter reflecting the number of distinct geometric configurations possible, that satisfy the condition of energetic stability of the critical cluster (see sec. 3.1.3). To form a new stable island, a critical cluster has to aggregate another particle. This yields the product of critical cluster density and monomer density N_iN_1. The rate is also proportional to the hopping rate h, which determines the principle speed of monomer diffusion and another constant called *capture number σ_i*. This number reflects the diffusional flow of particles to a critical cluster from the surrounding area. The mathematical treatment uses theory of two-dimensional random walks to deal with this problem [41]. However, at this point the capture number will be considered to be a constant in order to keep the mathematical expressions simple, leading to the following expressions:

$$K_{nuc} = \sigma_i h N_1 N_i \overset{\text{(eq. 7)}}{=} c_i \sigma_i h \exp[\beta E_i](N_1)^{i+1} \tag{8}$$

Eq. (8) contains the important information, that the nucleation rate is proportional to the monomer density to the power of $i + 1$.

For the simple case of irreversible aggregation, meaning that $i = 1$, where the binding energy is zero, $E_i = 0$, since the critical nucleus is the monomer itself, the expression gives $K_{nuc} \sim N_1^2$, the square of the monomer density.

For the aggregation rate similar ideas give

$$K_{agg} = \sigma_{agg} h N_1 N \tag{9}$$

where σ_{agg} is the mean capture number for stable islands.

Using these equations, one can now analyze the different regimes of nucleation and growth of submonolayer films.

(i) **Transient regime**

At the beginning of the process, the surface is considered to be free of adsorbates. When the first particles arrive, no islands are formed yet, because the density is too low. After the first particles nucleate, the growth of the islands can still be neglected, since the length separation of the clusters is still so big that monomers are more likely to find each other than an existing cluster. The monomer density can be approximated simply by the coverage:

$$N_1 \sim Ft = \theta$$

Using this in the expressions for island density and nucleation rate respectively, the island density can be derived:

$$N = \int dt K_{nuc} \overset{\text{(eq. 8)}}{\approx} \int dt c_i \sigma_i \exp[\beta E_i] h(\underbrace{Ft}_{\theta})^{i+1}$$

$$N = (i+2)^{-1} c_i \sigma_i \exp[\beta E_i](h/F)\theta^{i+2} \tag{10}$$

(ii) **Steady-state regime**

After some time, the monomer density will stop raising, because the particles will be either incorporated into existing stable clusters or nucleate islands, compensating for the incoming flux. If coalescence is still negligible, meaning that the coverage is still somewhat small, and nucleation is already of minor importance and therefore neglected in the rate equation, this gives the following condition:

$$dN_1/dt \approx F - K_{agg} \overset{\text{(eq. 9)}}{=} F - \sigma_{agg} h N_1 N \approx 0$$

which yields the following expression for the monomer density:

$$N_1 \approx \frac{F}{\sigma_{agg} h N} \tag{11}$$

which leads to:

$$\frac{dN}{dt} \approx c_i\sigma_i \exp[\beta E_i] h \left(\frac{F}{\sigma_{agg} h N}\right)^{i+1}$$

$$\int dN N^{i+1} = \int dt c_i \sigma_i \sigma_{agg}^{-(i+1)} \exp[\beta E_i] h \left(\frac{F}{h}\right)^{i+1}$$

$$N^{i+2} = (i+2) c_i \sigma_i \sigma_{agg}^{-(i+1)} \exp[\beta E_i] (F/h)^i \underbrace{Ft}_{\theta}$$

$$N = \left((i+2) c_i \sigma_i \sigma_{agg}^{-(i+1)}\right)^{1/(i+2)} \exp\left[\frac{\beta E_i}{i+2}\right] \left(\frac{h}{F}\right)^{-\frac{i}{(i+2)}} \theta^{1/(i+2)} \qquad (12)$$

Comparing the two equations for the island density, eq. (12) and eq. (10), one can determine the crossover coverage θ^*, where the system changes from the transient to the steady-state regime. The following relation must hold:

$$(i+2) c_i \sigma_i \sigma_{agg}^{-(i+1)} \exp\left[\frac{\beta E_i}{i+2}\right] \left(\frac{h}{F}\right)^{-\frac{i}{(i+2)}} (\theta^*)^{1/(i+2)} \overset{!}{=} (i+2)^{-1} c_i \sigma_i \exp[\beta E_i] (h/F) (\theta^*)^{i+2}$$

$$\Rightarrow \theta^* = \left((i+2)^2 \sigma_{agg}^{-(i+1)}\right)^{\frac{-(i+2)}{(i+1)(i+3)}} \exp\left[-\frac{\beta E_i}{i+3}\right] (h/F)^{-\frac{2}{i+3}}$$

By putting the last value into eq. (10), one can sum up the derived expressions for the island density:

Island Densities for the Different Growth Regimes

$$N = c_i \sigma_i (i+2)^{-1} \exp[\beta E_i] (h/F) \theta^{i+2} \qquad \text{...transient regime} \qquad (13)$$

$$N^* \sim \exp\left[\frac{\beta E_i}{i+3}\right] (h/F)^{-\frac{i+1}{i+2}} \qquad \text{...crossover} \qquad (14)$$

$$N = \eta(\theta, i) \exp\left[\frac{\beta E_i}{i+2}\right] (h/F)^{-i/(i+2)} \qquad \text{...steady state regime} \qquad (15)$$

where

$$\eta(\theta, i) = \left((i+2) c_i \sigma_i \sigma_{agg}^{-(i+1)} \theta\right)^{1/(i+2)}$$

The expression $\eta(\theta, i)$ is therefore a weak function of θ and i [35]. Fig. 10 shows a sketch of the qualitative behavior of these expressions for different values of i. At the crossover values, the curve is not smooth, which is due to the mathematical approximations. The main message here is that when going to higher values of i, the island number does not change significantly any more in the steady state regime. Consequently, for this case nucleation happens mainly in the transient regime, whereas for $i \leq 2$ the island density still exhibits considerable increase even in the steady state regime.

Admittedly, these simple considerations contain crude approximations and are just of instructional use. In the work of Amar, Family and Lam [42] two additional regimes are mentioned. Firstly, this includes the *aggregation regime*, which corresponds to the steady-state regime in the limit of higher coverage. Because of the $1/(i+2)$-exponent of θ in the expression of $\eta(\theta, i)$, the island density stays the same for some time, since only aggregation with existing clusters happens and nucleation vanishes. Secondly, there is a *coalescence and percolation regime*. Regardless of the size of i, when the coverage increases further, coalescence of existing islands happens. Therefore, one cannot assume a point-like nature of the particles for bigger coverages. Fig. 11 shows their widely-used plot of the scaling regimes.

Finally, it is worth mentioning that a rewritten version of eq. (15), also called the *Venables Equation*, is quite useful to compare with experiments [35, 39, 43]. In this case the involved energies are written together in the exponential, and the fraction h/F is rewritten using different quantities:

Venables Equation

$$\frac{N_x}{N_0} = \eta(\theta, i) \left(\frac{4R}{\nu N_0}\right)^{i/(i+2)} \exp\left(\frac{\beta(E_i + iE_d)}{i+2}\right) \tag{16}$$

where

N_x	...	saturation island density
N_0	...	number of surface sites per unit area
R	...	impingement rate per unit area
ν	...	attempt frequency for diffusion
E_i	...	binding energy of critical cluster of size i
E_d	...	activation barrier for diffusion
$\eta(\theta, i)$	...	some function of coverage and critical nucleus size

Taking the logarithm of this equation yields [11]:

$$\ln(N_x) = \ln(\eta N_0) + \frac{i}{i+2} \ln\left(\frac{4R}{\nu N_0}\right) + \frac{E_i + iE_d}{(i+2)kT} \tag{17}$$

Imaging techniques such as STM or AFM provide data which allow to count the islands, and therefore to calculate the density. For example, plotting $\ln N_x$ versus $\ln R$ for constant temperature gives one the opportunity to find out the slope and the intercept with the y axis and therefore get information about the critical nucleus size and the energies involved at constant temperature. In fact, it is important to make sure that the dependence on the coverage is weak in order to keep the first term a constant, which can be expected in the aggregation regime.

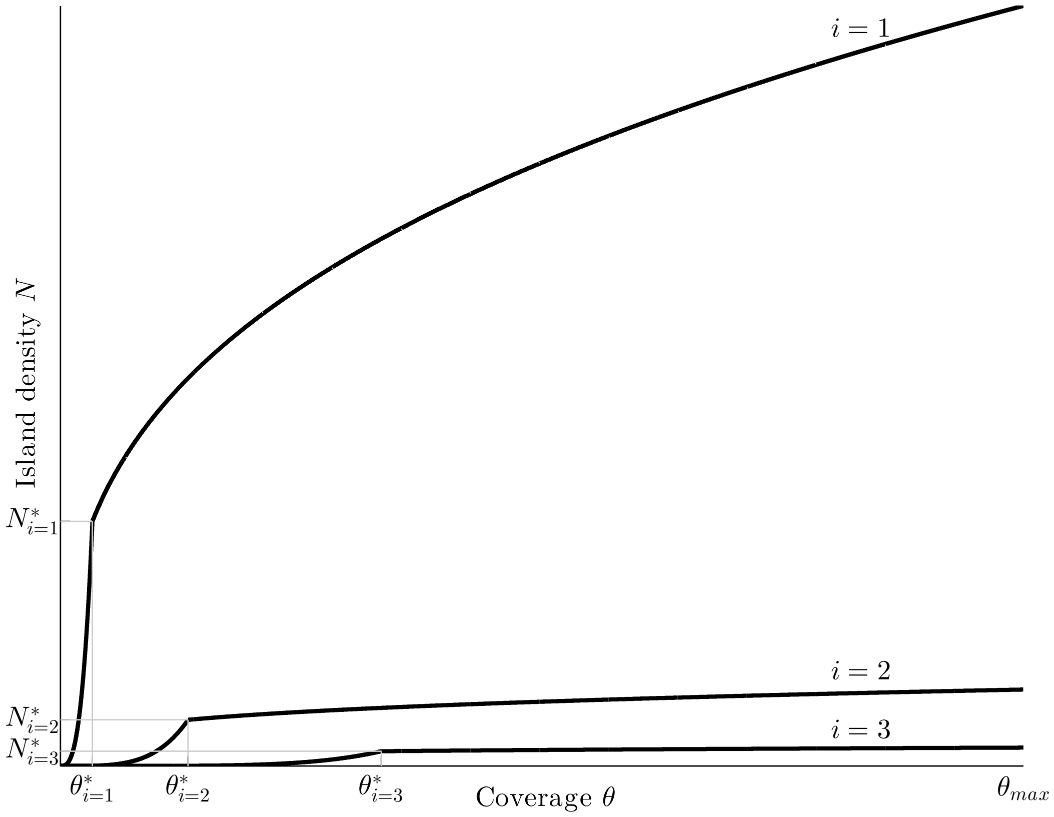


Figure 10: The different growth regimes according to eq. (13-15) for the values $i = 1, 2, 3$ are drawn qualitatively, including the indication of the crossover-values θ^* and N^*. It can be clearly seen that for higher values of i, the island density does not grow any more in the steady state regime, whereas for $i = 1$ (irreversible aggregation mode) there is still a significant increase and makes the main contribution to the density.

3.1.5 Island Size Distributions

The rate equation approach above shows that the density of stable clusters depends highly on the ratio h/F, on the energy barriers involved (E_i and E_d), and also on the

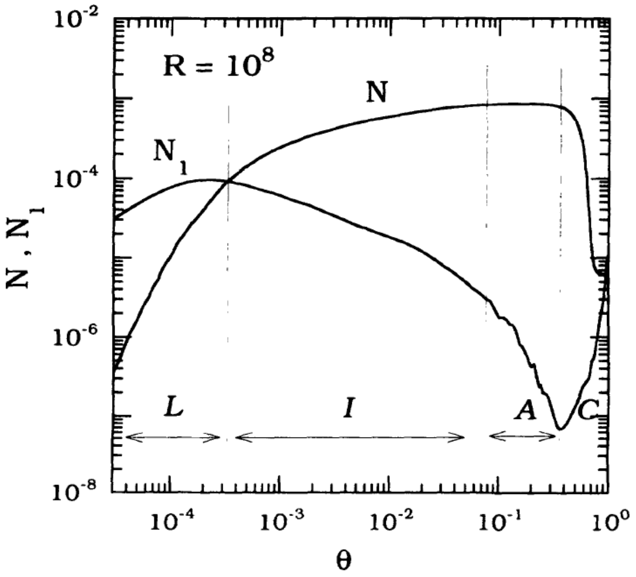


Figure 11: The different growth regimes for irreversible aggregation mode resulting from simulations in the hit-and-stick mode as Log-log plot. The regimes are: low coverage (transient) regime (L), intermediate coverage regime (I), aggregation regime (A), and coalescence and percolation regime (C). N denotes the unscaled island density, and N_1 is the monomer density. The regimes I and A both correspond to the steady-state regime (eq. (15)) mentioned in this work. (Parameter $R := D/F = 10^8$) [42]

critical nucleus size i. One can also expect a h/F dependence for the size distributions of the clusters, and they turn out to depend on i as well. Yet in this case, the dynamic scaling assumption [44] is widely used to get rid of the h/F-dependence by scaling the density of clusters containing s particles at a coverage θ to the average value S in the following way [45]:

$$N_s(\theta) = \theta \frac{f_i(s/S)}{S^2} \tag{18}$$

The average island size can be expressed as

$$S(\theta) = \frac{\sum_s sN_s(\theta)}{\sum_s N_s(\theta)}$$

One is looking for a function $f_i(s/S)$ giving distributions that agree with experiments. Island size distributions derived from the point-like model used for the rate equations above show to be simply wrong [45]. There is an approach for irreversible growth, using capture-zone distributions together with rate equations to derive expressions valid for islands that are not point-like [46]. However, since we are interested in models for reversible growth, expressions including i as parameters are needed. There is an expression which was not derived from rate equations, but by scaling considerations [45],

Table 1: Values of a_i and C_i in eq. (19) for $i = 1 \ldots 6$. [11]

i	a_i	C_i
1	0.2715	1.1091
2	0.2976	1.9678
3	0.3086	3.2385
4	0.3145	5.1214
5	0.3182	7.9036
6	0.3207	11.9963

which is of the following form:

$$f_i(x := s/S) = C_i x^i \exp(-i a_i x^{1/a_i}) \tag{19}$$

with the conditions for the constants C_i and a_i:

$$\frac{\Gamma\left[(i+2)a_i\right]}{\Gamma\left[(i+1)a_i\right]} = (ia_i)^{a_i}, \quad C_i = \frac{(ia_i)^{(i+1)a_i}}{a_i \Gamma\left[(i+1)a_i\right]}$$

One can determine the actual values of these constants by using the fact that the scaling function f_i is normalized $\int_0^\infty f_i(x)dx = 1$ and averages to one $\int_0^\infty f_i(x)xdx = 1$. These values are given in tab. 1.

3.1.6 Attachment Limitation

In contrast to the rate equation approach, which mainly describes formation of islands that is limited by diffusion (diffusion limited aggregation, DLA), there is a model where the attachment of monomers to existing islands is hindered [23, 24]. For a case of anisotropic molecules, the attachment to existing islands may be in fact hindered [16].

The capture numbers in this case are not only influenced by island size and morphology, but also by a barrier E_B that can be seen as an additional repulsive energy that has to be overcome by the monomers to attach to the islands. Fig. 12 shows a sketch of a potential energy surface where the additional barrier is indicated.

For nucleation, the capture number has to be adjusted to account for this effect using the equation [24]

$$\sigma_i^{-1} = \sigma_D^{-1} + \sigma_E^{-1}$$

where σ_D is the diffusion-limited capture number and σ_B is the attachment-barrier capture number.

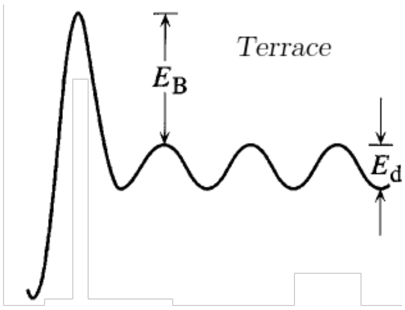


Figure 12: Sketch of the potential energy evolution from the edge of an island towards a terrace. Particles away from the barrier have to overcome a smaller barrier than the ones hopping to the island rim in one step. [24]

3.2 Overview of Computational Simulation Techniques of Crystal Growth

This section clearly does not aim at giving a review of all the available simulation techniques. For this case, the reader may be referred to the review of Clancy [47], which this overview uses as a basis. The following rather states the well-established computational methods in order to provide an idea about the field and to help the reader classify the technique subject to this work.

3.2.1 Electronic Structure Calculations

There is a class of calculations which take into account the electronic structure of the system. They are very accurate, since they are both small in time and length scales, but they tend to be also the computationally most intensive ones. Moreover, the term "ab initio“ methods is often used in this context, because they usually need little parametrization, since they calculate the electronic states of the system. Famous methods like Hartree-Fock (HF) and Density Functional Theory (DFT) are of enormous importance. However, both methods use some assumptions to approximate the exact solution to the Schrödinger equation.

In the case of traditional HF calculations, usually the solution gets approximated by a mean field method of combining molecular orbitals, neglecting dynamic electron correlations. There are ways to include these by perturbation theory methods.

In contrast, DFT uses a different approach, mainly based on the two facts, that the ground state energy of the Schrödinger equation is a functional of the electron density and that the result of minimization is a true descriptor of the density that would result from the exact ground state solution of the Schrödinger equation. The Kohn-Sham equation [48] is used to solve the Schrödinger equation for non-interacting particles, which gives the same result as for an interacting system. However, the point is that the functional is not exactly known and therefore has to be approximated. Additionally, the initial density has to be guessed, which is usually a combination of single atom orbitals.

The procedure is an iterative process of diagonalizing the Hamiltonian, selecting the resulting orbitals of the lowest energy and followed by guessing a new density, which is finished after the algorithm converges. There are plenty of choices for the functional as well as for the wave function.

The number of particles possible to simulate by these methods is roughly of the order of 10^3, because of the computational intensiveness. Furthermore, they consider also properties that are not of interest in detail when it concerns crystal growth. Rather the positions of the atoms/molecules are of interest than the electron density itself, although it determines the system's behavior.

However, when one wants to find out the reaction paths certain particles can take on a surface, such detail is necessary to provide more coarse-grained simulation techniques (such as KMC) with the energy barriers mentioned above in sec. 3.1.1 about Transition State Theory (TST).

3.2.2 Molecular Dynamics Simulations

In contrast to quantum mechanical approaches in Molecular Dynamics (MD) the classical equations of motion are solved. The system is represented by the individual atoms, whose positions are altered using continuous time steps, according to different algorithms. This time step is typically at the order of 1 fs. Convergence of these algorithms demands for small time steps, whereas one usually wants it to be large to cover reasonable time scales. Famous algorithms are the Verlet algorithm [49] and also the Beeman algorithm [50], which are quite common and well tested. Parametrization is a critical point, where one has to specify the algorithm, the potential energy functions, the time step, the boundary conditions, and the initial configuration of the system. From statistical physics, we know that systems of particles can by described by microcanonical (NVE), canonical isothermal (NVT), or isothermal-isobaric (NPT) ensembles. One has to be careful to choose the right ensemble for the initial configuration. Most importantly, the potential energy functions have to be provided. Besides intermolecular potential energy, there is also an intramolecular part containing rotational and vibrational energy parts.

The simulation times nowadays are usually limited to a few nanoseconds, of course depending on the system size. Therefore, one has to be aware of that in certain cases the global energy minimum might not be reachable in such small time periods, thus maybe providing the user with misleading local energy minima as a result.

However, this technique is well-established, hence there is plenty of software available nowadays. Introductory books are available as well, providing the reader with information in a instructional manner [51, 52].

3.2.3 Monte Carlo Simulations

The idea of solving numerical problems by using random numbers is named after the Monte Carlo Casino in Monaco [53]. In contrast to the methods mentioned above, the central point is the random experiment on the computer. Extensively applied in physics nowadays are spatial Monte Carlo methods to study the behavior of thermodynamic quantities of condensed matter [54, 55].

Traditionally, problems of equilibrium phenomena in the field of statistical physics are treated with the MC idea, for which the method of Markov chain Monte Carlo (MCMC) is famous [56, 57]. With this method, one approximates sampling from an arbitrary distribution and by creating a chain of samples, one obtains a limiting distribution that is equal to the desired one. To obtain reasonable results for equilibrium properties, certain conditions have to be met by the algorithm, such as stationarity (detailed balance). These provide for the needed reversibility of the system. One widely used algorithm is the Metropolis-Hastings algorithm [58], which, in a few words, accepts or rejects a proposed Monte Carlo step by comparing a trial random number from a uniform distribution to the instantaneously computed Metropolis probability. Therefore this algorithm belongs to the class of *null-event* algorithms, because there is a non-zero probability, that nothing happens in one of the steps. This can be an issue if the acceptance probability is low, leading to highly inefficient calculations [59].

In fact, there are plenty of applications of MC on non-equilibrium systems nowadays. As mentioned in sec. 3.1.2, crystal growth performed by molecular beam epitaxy (MBE) is typically a non-equilibrium phenomenon, since the temperature has to be low enough to sustain the solid state, which suppresses equilibration. Because reversibility is not given any more, the algorithms also do not have to include the support for it. Consequently, for these systems, different algorithms exist, extending the class of suitable algorithms to rejection free algorithms, where the system changes in every step [59]. This leads to the algorithm used in this work for Kinetic Monte Carlo presented in sec. 3.3.

The reachable time scales and system sizes of MC simulations are usually much higher than the ones of molecular dynamics simulations and obviously for electronic structure calculations. Yet MC methods do not provide much physics from an ab-initio point of view. Therefore compared to the other two methods, one has to provide MC programs with a lot of parameters and the possible processes typically have to be defined in advance. MC offers a way to treat the evolution of a physical system in a statistical manner, where consecutive individual states of a physical system are sampled and the detailed physical processes going on in between two such states are neglected.

3.3 Kinetic Monte Carlo (KMC) Simulations

A specific class of Monte Carlo (MC) algorithms provides information about the time evolution of the system, which clears the way for treating kinetically limited processes, and is therefore called Kinetic Monte Carlo (KMC). It is a spatial microscopic simulation method typically applied to atomistic simulations. The rates (in units of $[s^{-1}]$) of the different processes that may happen in a physical system, have to be known, or in other words, the pathways a particle can take have to be provided to a KMC solver. From the inverse of these rates, the time elapsed after such a process can be calculated and therefore a relation to real time can be established. This is the main difference to the widely applied Metropolis method for "normal" MC, which usually does not provide such information.

3.3.1 Delimitation of the Concept

Although methods similar to the one presented here have been applied at least since the 1970s, there is a lack of literature defining clearly the algorithm for which the term "Kinetic" is used. One reason for that may be that until recently, there used to be no widely adopted commercially or freely available KMC program [47] that supports sufficient parameters to change in order to be very flexible. Another reason may be that not everyone uses the term "kinetic", even when using an algorithm that belongs to that class (e.g. Dynamic Monte Carlo [60], Gillespie Algorithm [61]). Moreover, different names for approaches are invented when parts of the model are different, but one still uses the same algorithm.

One example is a version of Mesoscopic Monte Carlo (MMC) [62], where the model is not microscopic any more, but the algorithm contains kinetic information. Without making a claim to be complete, besides the "classical" KMC method, there are a variety of hybrid models, such as MD-KMC (Molecular Dynamics Kinetic Monte Carlo) [63] and Quasicontinuum methods [64, 65]. But there are also methods that use altered algorithms to overcome certain issues that appear in certain applications, but do not change the principle (Multiscale kinetic Monte Carlo [66], Coarse Grained Kinetic Monte Carlo [67]). However, the reader may be advised that one can get easily confused by the chaos of names and methods. All those particular methods are not subject to this work.

There exists a code of the Sandia National Laboratories (U.S. Department of Energy) called SPPARKS (Stochastic Parallel PARticle Kinetic Simulator) [68, 69], which is the only more or less easily adoptable software available and is most suitable for single atom (metal) hopping processes rather than for molecules. The usefulness compared to well-established DFT or Molecular Dynamics programs is still poor. A reason for this is that source code has to be added manually by the user for a lot of processes specific for the actual application [47]. Hence, until a code is available with a more generic nature, one has to invest quite a lot of time in programming.

For the following, the well-structured review paper of A. Chatterjee and D.G. Vlachos [59] is used as a basis.

3.3.2 Applications

Nowadays KMC simulations are extensively applied in a variety of scientific fields, thus the applications are not in the least limited to crystal growth. Important is, however, that the rates of the processes going on in the simulated physical system have to obey the Arrhenius form of eq. (3) [47].

Concerning the field of materials modeling, a few applications can be named [69]:

- surface diffusion, growth

- adsorption, desorption kinetics
- vacancy diffusion (e.g. in alloys)
- domain coarsening, grain growth/refinement

Moreover, the special case of off-lattice simulations can be applied to problems of defect mobility, amorphization/recrystallization, and atomic relaxation, but are usually a field addressed by Metropolis Monte Carlo [59, 68].

3.3.3 Stochastic Description - The Master Equation

As already mentioned in sec. 3.1.2, nonequilibrium statistical dynamics determine the bahavior of systems typically simulated by KMC. In TST those thermal vibrations without effect on the configuration of the microscopic states are neglected, treating only the "rare" events of e.g. atoms hopping from a crystal site to another. The information on momentum degrees of freedom of the treated particles as well as on the thermal vibrations is used to determine microscopic rates that are used to treat the system in a stochastic way. One can denote the current configuration of a system by a microscopic state vector $\boldsymbol{\sigma}$, which depends both on space and time variables. If one process happens, e.g. a molecule jumps from a surface site to another, the state vector $\boldsymbol{\sigma}$ changes. As certain states may have different probabilities $P(\boldsymbol{\sigma})$ of occurring than others, one can track the change of this quantity with respect to time [59]:

Master Equation

$$\frac{dP(\boldsymbol{\sigma})}{dt} = \sum_{\substack{\boldsymbol{\sigma}' \\ \boldsymbol{\sigma}' \neq \boldsymbol{\sigma}}} G(\boldsymbol{\sigma}' \to \boldsymbol{\sigma}) P(\boldsymbol{\sigma}') - \sum_{\substack{\boldsymbol{\sigma}' \\ \boldsymbol{\sigma}' \neq \boldsymbol{\sigma}}} G(\boldsymbol{\sigma} \to \boldsymbol{\sigma}') P(\boldsymbol{\sigma}) \tag{20}$$

Here $G(\boldsymbol{\sigma}' \to \boldsymbol{\sigma})$ is the coefficient for transitions from state $\boldsymbol{\sigma}'$ to $\boldsymbol{\sigma}$. The change with time of the probability that a system is in a state $\boldsymbol{\sigma}$ can be found out by summing over all rates of changes from $\boldsymbol{\sigma}'$ to the particular state $\boldsymbol{\sigma}$, minus the rate of the process in the other direction.

In other words, the rates from other states to this minus the ones from this to the others determine the likelihood of the state to change. This deterministic equation cannot be solved analytically for typical system sizes, but is a stochastic description of the time evolution of the system and therefore is the basis of KMC simulations. One can imagine that a lot of transitions are practically of probability zero, because many states can only be reached from others by a lot of intermediate ones. This can be understood when looking just at the change of the state at a particular site with index i:

$$d\boldsymbol{\sigma}_i = \sum_j \Gamma_{ij}^{+}(\boldsymbol{\sigma})dt - \sum_j \Gamma_{ij}^{-}(\boldsymbol{\sigma})dt$$

where $\Gamma_{ij}^{+}(\Gamma_{ij}^{-})$ is the rate for particle addition(deletion) at site i by a process j. A particle at a site far away from i is very unlikely to hop to site i in one step. Consequently, Γ_{ij} is a sparse matrix which illustrates why the coefficient matrix $G(\boldsymbol{\sigma}' \rightarrow \boldsymbol{\sigma})$ of the more globally written eq. (20) must also be sparse.

This information is crucial in order to develop efficient algorithms for determining the (non-zero) rates (see sec. 3.3.5).

3.3.4 KMC Algorithms

When referring to KMC, the so called BKL-algorithm [70] of Bortz, Kalos and Lebowitz, originally proposed for applying to the Ising spin system, is commonly applied. They were actually not the first to invent the algorithm [71, 72], but provided a detailed description of the method. A process that is possible to happen in a physical system described in our model, is called *event*. If all the events possible in a system are known, one particular with index m is selected with a probability proportional to its rate r_m using a random number. Then the event is executed and finally the time is increased by the time step. This time step is given by the exponential distribution with expectation value of $1/R_N$, the inverse of the overall rate, which is the sum of all rates $R_N = \sum_{k=0}^{N} r_k$. This corresponds to the probability that all other events do not happen in that step and contains fluctuations in time corresponding to the Poisson process. The procedure can be represented by the following steps:

Basic Rejection-Free Algorithm of KMC

```
1. Set time $t = 0$
2. Create list $r_m$ of all possible rates in the current configuration
3. Calculate cumulative function $R_m = \sum_{k=0}^{m} r_k$ where $m = 1 \ldots N$, and $N$ is the total number of transitions
4. Find out which event $k$ satisfies $R_{k-1} < \zeta_1 R_N < R_k$, where $\zeta_1 \in (0, 1]$ is a uniformly distributed random number
5. Execute event with index $k$
6. Advance time with step $\Delta t = -\frac{\ln(\zeta_2)}{R_N}$ where $\zeta_2 \in (0, 1]$ is a uniformly distributed random number
7. If $t < t_{max}$ return to step 2, otherwise stop
```

In many cases, there are more efficient ways to find those events to execute. Step 4 is the one which costs a lot of time, if there are a lot of events. Besides linear search methods, which are very inefficient, there are methods which use binary trees that provide quite acceptable scaling behavior.

Here we will not concentrate on these, but focus on the following procedure: The rate r, at which this event happens, is usually existent many times in the system, because the same may happen at many sites. If there is a finite number N_p of different kinds of events, every event *class* can be given a number j. The number of events of class Γ_j is denoted by n_j. Each event that belongs to the class j happens at a rate Γ_j. By choosing a random number, first an event class gets selected, and then one event of that class is executed. There is no computational bottleneck in finding the one event of the class to execute, because it is simply selected by its index, rather than searched by an algorithm. This is possible because all the events of the class have the same rate. An integer random number between 1 and n_j can be generated efficiently. This way of keeping track of the possible event classes that are existent in a system is called the n-Fold Way and is also often referred to as BKL method.

n-Fold Method: BKL-KMC Algorithm

1. Set time $t = 0$
2. Create list Γ_j of all possible classes of events in a system
3. Create (update) the list of the individual events of all classes j in the current configuration.
4. Calculate cumulative function $R_m = \sum_{k=0}^{m} n_k \Gamma_k$ where $m = 1 \ldots N_p$
5. Find out which event class k satisfies $R_{k-1} < \zeta_1 R_N < R_k$ to execute, where $\zeta_1 \in (0, 1]$
6. Execute one particular event of class k, selected by an integer random number $i \in [1, n_k]$
7. Advance time with step $\Delta t = -\frac{\ln(\zeta_3)}{R_N}$ where $\zeta_3 \in (0, 1]$
8. If $t < t_{max}$ return to step 3, otherwise stop

This way only the class to execute is searched for and step 4 in this case can be quite efficiently handled. [59]

As already mentioned, a lot of different algorithms can be found in literature, which are not part of this work. Time is then incremented by $\Delta t = -\ln(r)/\Gamma_{tot}$ where $r \in (0, 1]$ is a uniformly distributed random number, $\Gamma_{tot} = \sum_j n_j \Gamma_j$ is the total rate.

3.3.5 Event Update Algorithms

Since the rejection free algorithms depend on an accurate list of events (step 3 above), a time-saving algorithm for updating the event lists is crucial. The requirements for such an update procedure greatly depend on the kind of system that is simulated. If single processes can change the situation in the whole system, maybe the whole list has to be updated, which is termed “global update“. For this case efficient methods exist, for example using binary trees. In general, one can say that the more exactly one knows the system, the better the algorithm can be optimized.

If only the neighborhood of a particle changes when an event is executed, which is normally the case for diffusion processes, there is an extremely efficient method proposed by Schulze [73] as an example for a ”local update” method. In short, this is done by recalculating the events of the neighboring particles only, using an inverse list that keeps track of where to find the events of a specific particle in the event lists. In the section of the actual implementation in the context of this work (sec. 4) an example is given, where this approach is presented in detail.

3.3.6 Computational Scaling

It must be stated at this point that in a lot of publications, especially concerning the ones aiming at presenting accelerated algorithms, the computational intensity is either not analyzed at all, or just mentioned to e.g. “achieve a factor of ten speed up” [66] than the non-accelerated one, without giving explicit mathematical expressions of the scaling order. Detailed analysis is given whenever search and update algorithms are proposed, because these theories normally result from theoreticians developing algorithms. [59, 74]

What one can definitely state is, that the system size, the time range, as well as the rate of the events have a huge impact on the scaling of the algorithm. These quantities cannot be avoided without approximations, because they are exactly the ones which are desirable to be represented accurately. The parts which are wasteful in the sense of doing useless calculations are usually the ones responsible for search and update mechanism.

However, for the simple case of (homogeneous) epitaxial surface growth, there is a paper of Chou and Falk [74], who did an analysis of the scaling of free diffusion events on a terrace. The number of events, that are necessary to form one monolayer scales like

$$\mathcal{O}\left(\frac{R_d N}{F} + \frac{N L^2}{6}\right); \qquad R_d/D \ll 1 \tag{21}$$

where

$\mathcal{O}$ ... denotes the upper asymptotic bound of the algorithm’s execution time[1]
R_d ... detachment rate of particles from the island rim
N ... total number of sites on the surface
F ... flux rate
L ... average terrace length between the edges
D ... terrace diffusion rate

Since this work is mainly about submonolayer growth, this expression will be quite useful to estimate the behavior of CPU time, when various parameters are changed.

3.3.7 Issues & Challenges

The well-structured review paper of Chatterjee and Vlachos [59] gives a nice overview of challenges that appear when carrying out spatial KMC simulations. The key points are

[1] See “big O notation” in computational complexity theory [75]. In short, the expression says that when looking e.g. at the number of sites N the execution time grows linearly with N in the worst case, but does not grow with a higher power or even exponentially. Computers may be different in performance, therefore constants are neglected, but the complexity stays the same.

stated here briefly with giving key references, where the prevailing problem of stiffness is of special importance.

Input to Simulation

As already mentioned, the microscopic processes have to be identified and their transition probabilities have to be determined. This includes, for example, all types of diffusional processes on surfaces, like lonely-atom diffusion to nearest and next-nearest neighbor positions, but could also include exchange processes of adatoms with substrate atoms. The catalog of the processes has to be complete, because otherwise the results may be severely erroneous [59].

Efficient Search and Update Algorithms

This issue has already been addressed (sec. 3.3.4,3.3.5), but is listed here by reason of completeness.

Time Scale Separation: Stiffness

Even though TST provides us with an efficient way to avoid the treatment of unnecessary vibrations, KMC simulations often suffer of time scale separations between processes of low and high energy barriers. Since the rate given by eq.(3) depends exponentially on the ratio of energy and temperature, the difference between high and low barriers may be several orders of magnitude. This does not mean that the algorithm itself is inefficient, but rather that most of the time uninteresting processes are carried out, since the events with low rates are often the ones of interest. The famous example in epitaxial growth is the diffusion of lonely particles on a surface far away from the step edges, which on the one hand is the driving force of nucleation and growth in the Diffusion Limited Aggregation (DLA) mode. On the other hand, most computational effort has to be invested in the simulation of random walk motion, where one actually is interested in the change of island density and island morphology, which happens usually when higher barriers are overcome. The dependencies of the execution time on the different parameters are given by eq. (21). The situation further worsens if, for some reason, also the number of particles in those time consuming states is high, yielding an additional separation of event numbers. To top it all off, particles in states of favored energy often have less degrees of freedom, e.q. particles on the island edge may only have two directions left to hop, where free particles may have still four possibilities on a square lattice.

There is a variety of methods to overcome this issue for specific applications. For the case of diffusion on a surface multiscale methods are subject of recent work [40, 66, 74, 76, 77, 77]. The idea in this case is, in principle, to let lonely particles hop bigger distances. The issue is therefore also connected to length scale separation. However, these methods sometimes increase by some factor of speed, which is already quite useful, but the order of complexity does not reduce. Moreover, there are methods to treat the diffusing particles with diffusion equations like the Level-set method (BCF) [78, 79], but these are approximations and therefore sometimes do not represent fluctuations in the right way.

Length Scale Separation

In the case of spatial simulations, with large system sizes, transport over long distances is of interest, which could be another bottleneck for CPU time. To address these problems, there are methods to coarse grain the system (Coarse Grained Monte Carlo, CGMC) [67, 80, 81]. The idea is to structure the system into different sizes of cells, and to treat the diffusional flow of particles in a more global way. This can be combined with multiscale approaches and makes it easy to implement parallel algorithms [76].

Single Event Handling

Simple microscopic KMC algorithms are not capable of handling more than one event at a time. There are ways to perform parallelizations by segmenting the system, and performing the algorithm in parallel for these regions. One has to be careful at the borders between those segments and also about the time clock, which could advance at different time steps depending on the surroundings [68]. Another way is known as the τ-leap method, which fires multiple events at a time while coarse-graining the system in time [82, 83].

3.3.8 Lattice models

It is quite common to use lattice models for KMC though there are off-lattice models currently in development [84]. The main idea is that every site can be either occupied or unoccupied by a particle:

$$\sigma_i = \begin{cases} 1 & \text{occupied by particle} \\ 0 & \text{unoccupied} \end{cases}$$

With the Arrhenius law (TST) one can calculate the transition probability from a state i to a state i'. A particle can hop from a cell to a neighboring one only if the current cell is occupied and the neighboring is not:

$$\Gamma(i \to i') = \frac{1}{4}\nu\sigma_i(1-\sigma_{i'})e^{-\frac{E}{kT}}$$

where

E ... energy barrier
ν ... attempt frequency
T ... temperature
k ... Boltzmann constant

In a widely used model for metal epitaxy named solid-on-solid (SOS) model the energy barrier takes the following form:

$$E = \frac{Q + n * J}{kT}$$

where

Q ... surface diffusion barrier
n ... number of nearest neighbors
J ... nearest neighbor binding energy

It simply means that when a particle diffuses on the surface the barrier raises with the number of nearest neighbors and always the surface diffusion barrier adds to the energy term. The sketch in fig. 7 shows such a model, where other names for the barriers are used: $Q = E_d$ and $n \cdot J = E_{bond}$.

3.4 Quasicontinuum Simulations of Nucleation and Growth

As already mentioned simulating nucleation and growth becomes expensive when the monomer density is high and the KMC algorithm spends most of the time with simulating monomer hops rather than island growth. Especially for higher values of the critical cluster size i and attachment limited system one can expect the monomer density to grow severely.

A recently developed approach is called Adatom-density [64] or Quasicontinuum Monte Carlo (QCMC, [65, 85]) method which uses diffusion equations to simulate the monomer density evolution of the system, and the particles of the islands are still positioned on discrete positions.

Using the rate equations (see sec. 3.1.4) it is possible to treat the flux F and the diffusion continuously and nucleate islands with the help of

$$dN/dt = \sigma_i h N_1^{i+1} \tag{22}$$

The way to do this is, in fact, to continuously extract some part of the monomer density according to the local density ρ_1 which leads to the integral

$$N = \int dt \sigma_i h \rho_1^{i+1}$$

After exceeding an integer value of this "collected" amount of monomers for nucleation an island gets added at a place on the lattice again with probability proportional to N^{i+1}.

As for attachment of monomers to islands, one sums up the values of ρ along the rim of an island, therefore gets an idea of how much "mass" is around such an island. When this value exceeds 1, a particle is added to the island rim and the probability function gets collapsed that way. For further details on this probabilistic techniques the reader may be referred to the paper of Mandreoli et al. [64].

In the following the important treatment of the monomer density by diffusion equations is described. For simplicity the mathematical explanations are in detail presented for the 1-dimensional case and a finite difference scheme is used. In fact the explanations mentioned in Devita et al. [85] have been extended.

The 1-dimensional diffusion equation reads as follows [86]:

$$\frac{\partial \rho(x,t)}{\partial t} = D\frac{\partial^2 \rho(x,t)}{\partial x^2} \tag{23}$$

The discretization is done by letting every grid point of the pure KMC simulation be a point on the grid of the continuous diffusion treatment. Using the index n for the time steps and the index $i = 1 \dots L$ for the x- and y- values, the discrete values of ρ will be denoted as

$$p_i^n = p(x_i, t_n)$$

Using the backward difference for the time step and a central finite difference approximation of the second space derivative, we end up at the implicit recurrence equation:

$$p_i^{n+1} - p_i^n = s\left(p_{i-1}^{n+1} - 2p_i^{n+1} + p_{i+1}^{n+1}\right) \tag{24}$$

with $s := \frac{D\Delta t}{(\Delta x)^2}$.

In a random walk, particles move either to the left or to the right with equal probability 1/2. However, this is only the case if the particle is far away from the edges where the rules may be different.

In order to account for the stochastic behavior of the random walk, as well as for the cells near the island edges, some modifications are made. Depending on the current cell, flow of monomer density may be not allowed in all directions uniformly. Near to an island, the rules for mass flow from and to certain cells (e.g. edge cells) have to be modified.

Coefficients are placed in front of forward-, backward- and central term in the following way:

$$p_i^{n+1} - p_i^n = s\left(c_i^- p_{i-1}^{n+1} - 2c_i^\circ p_i^{n+1} + c_i^+ p_{i+1}^{n+1}\right) \tag{25}$$

where

c_i^- ...fraction received from the left
c_i^+ ...fraction received from the right

and the coefficient c° corresponds to the amount of flow out of the cell, which has to be the same flow the neighboring cells receive from this cell:

$$c_i^\circ = 1/2\left(c_{i-1}^+ + c_{i+1}^-\right) \tag{26}$$

The Factor 1/2 compensates for the 2 in front of the central term and means that the flow out of the cell has to be divided by 2 because there are 2 neighboring cells.

For the particle at least 2 steps away from the island edge, the coefficients can be chosen to be $c_i^- = 1/2$, $c_i^+ = 1/2$ and $c_i^\circ = 1/2$.

Rearranging the implicit scheme (25)

$$p_i^n = -s\left(c_i^- p_{i-1}^{n+1} + c_i^+ p_{i+1}^{n+1}\right) + p_i^{n+1}(1 + 2c_i^\circ s) \tag{27}$$

one can see that this is a system of L equations with L unknowns and is solved by the matrix division:

$$P^{n+1} = M \setminus P^n \tag{28}$$

where $P^n := \{p_1^n, \ldots, p_L^n\}$ and the elements of the matrix read

$$M_{i,k} = \begin{cases} 1 + 2sc_i^\circ & \text{if } i = k \\ -sc_i^+ & \text{if } i < L \wedge k = i + 1 \vee i = L \wedge k = 1 \\ -sc_i^- & \text{if } i > 0 \wedge k = i - 1 \vee i = 0 \wedge k = L \\ 0 & \text{else} \end{cases}$$

for periodic boundary conditions.

For the case as presented in fig. 13, where periodic boundary conditions are introduced ($p_8 = p_1$), this gives the following result:

$$M = \begin{pmatrix} 1+s & -1/2s & 0 & 0 & 0 & 0 & -1/2s \\ -1/2s & 1+s & -1/2s & 0 & 0 & 0 & 0 \\ 0 & -1/2s & 1 & & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1/2s & 0 \\ 0 & 0 & 0 & 0 & -1/2s & 1+s & -1/2s \\ -1/2s & 0 & 0 & 0 & 0 & -1/2s & 1+s \end{pmatrix}$$

For the two-dimensional case, the diffusion equation reads

$$\frac{\partial \rho(x,y,t)}{\partial t} = D_x \frac{\partial^2 \rho(x,y,t)}{\partial x^2} + D_y \frac{\partial^2 \rho(x,y,t)}{\partial y^2} \tag{29}$$

For the space discretization now the indices (i, j) for the x- and y- values are used.

Analogous to the 1D-case (24) the implicit recurrence equation reads:

$$p_{ij}^{n+1} - p_{ij}^n = s\left(p_{i-1,j}^{n+1} - 2p_{i,j}^{n+1} + p_{i+1,j}^{n+1}\right) + s\left(p_{i,j-1}^{n+1} - 2p_{i,j}^{n+1} + p_{i,j+1}^{n+1}\right) \tag{30}$$

with

$$s := \frac{D\Delta t}{(\Delta x)^2}$$

assuming $D = D_x = D_y$ and $\Delta x = \Delta y$.

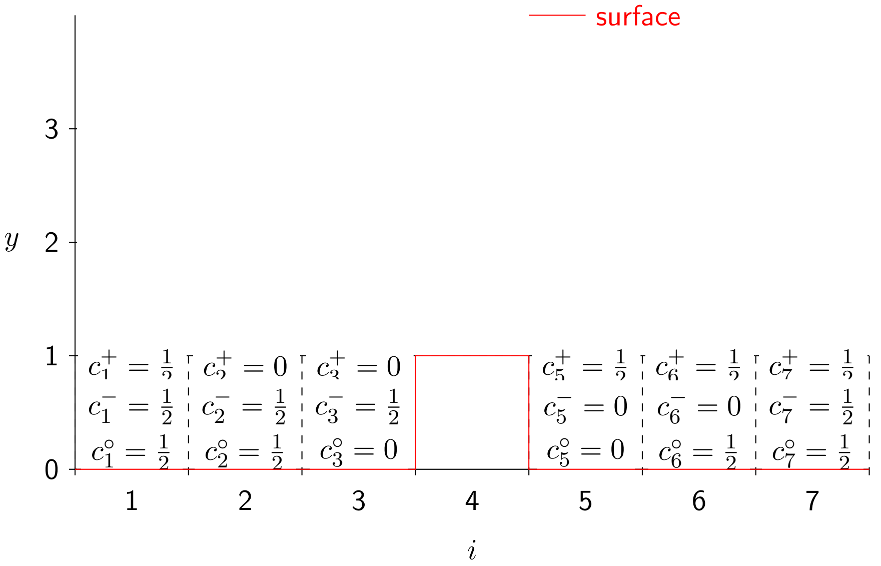


Figure 13: Sketch of the coefficients of the one-dimensional case for unhindered attachment and no detachment from the island edges as used by the Quasicontinuum Monte Carlo method (QCMC).

The coefficients are placed again

$$p_{ij}^{n+1} - p_{ij}^{n} = s\left(c_{ij}^{\leftarrow} p_{i-1,j}^{n+1} - 2c_{ij}^{\circ} p_{i,j}^{n+1} + c_{ij}^{\rightarrow} p_{i+1,j}^{n+1} + c_{ij}^{\downarrow} p_{i,j-1}^{n+1} - 2c_{ij}^{\circ} p_{i,j}^{n+1} + c_{ij}^{\uparrow} p_{i,j+1}^{n+1}\right)$$

and rearranging gives

$$p_{ij}^{n} = -s\left(c_{ij}^{\leftarrow} p_{i-1,j}^{n+1} + c_{ij}^{\rightarrow} p_{i+1,j}^{n+1} + c_{ij}^{\downarrow} p_{i,j-1}^{n+1} + c_{ij}^{\uparrow} p_{i,j+1}^{n+1}\right) + p_{ij}^{n+1}(1 + 4c_{ij}^{\circ} s) \tag{31}$$

where

$c_{ij}^{\leftarrow}$ . . . fraction received from the left
$c_{ij}^{\rightarrow}$ . . . fraction received from the right
$c_{ij}^{\uparrow}$ . . . fraction received from the top
$c_{ij}^{\downarrow}$ . . . fraction received from the bottom

and

$$c_{ij}^{\circ} = 1/4\left(c_{i-1,j}^{\leftarrow} + c_{i+1,j}^{\rightarrow} + c_{i,j+1}^{\uparrow} + c_{i,j-1}^{\downarrow}\right) \tag{32}$$

The treatment of the system by all these coefficients is very flexible when it comes to simulating effects as the attachment limitation (see sec. 3.1.6). By simply changing these coefficients so that the flow to the island is less than the one away from it, this effect can be reflected without having any problems with computational intensiveness compared to the conventional case where the coefficients have the same size. Mass conservation is always assured by eq. (32).

4 Computational Implementation

This section aims at presenting the actual program that was developed to carry out simulations of the proposed model.

4.1 History & Development

The algorithms presented in the following explanations were first developed and tested using MATLAB® in order to save time in both programming and debugging. Moreover, this software provides excellent tools for visualization and is therefore used to present the resulting data graphically. For the time consuming parts a standalone C++ command line application was developed for reasons of speed, which returns the simulation results in an appropriate format for MATLAB to plot. Consequently, all plots of simulation results were generated by MATLAB, even though the application itself is programmed in a different language.

The open-source software community nowadays provides the programmer with a considerable number of freely available, well-checked and reliable source codes for a variety of scientific applications. In fact, the particular functionality is usually encapsulated in so-called packages or libraries, which then can be used by the programmer. This makes it easy to create new programs by combining these already established program libraries, and one can concentrate on parts that are new or of special interest instead of spending time on the implementation of already well-developed auxiliary functions. For the case of MATLAB, these libraries are all part of the software since it is distributed as a stand-alone software, whereas for the more low-level language C++, the libraries have to be selected in detail by the programmer. The advantage of the latter is that further speed optimizations are possible by adjusting the configuration of the packages at compile time. For instance, checks of index overflows in arrays and other sources of memory problems can be turned off as soon as the program is considered stable, which boosts the speed considerably for memory intensive applications. Moreover, code optimizations for speed done by the compiler are often done for the actual machine the program is running on, which is not possible for commercial software since these packages are precompiled.

4.2 Pseudo Code Conventions

In order to present the algorithms in a plausible way, a version of pseudo code is used which is quite near to C++ syntax and that omits parts of the syntax and semantics which would make reading difficult. To clarify the meaning of the structure of this code, an example is given as follows regarding the reader's way of reading this thesis:

ALGORITHM: How to read through a thesis

```
procedure ReadThroughThesis(text)        ▷ Input parameters in brackets
    initialization;
    while not at end of this document do ▷ Repeat until this is false
        read current and increase knowledge;
        if understand then
            go to next section;
            current section becomes this one;
        else
            go back to the beginning of current section;
        end if
    end while
    return knowledge;                    ▷ Output of the procedure is returned
end procedure
```

The first part of the illustration specifies the procedure name and the input parameters. Next, all the steps that are to be performed when one implements the algorithm are specified. Keywords are written in **bold** face, variable names and logic expressions formatted in *italics*. Comments are specified in gray either with two slashes (//) in front when taking its own line or with a ▷ when it gives information about a line. They are not part of the actual steps, but they give details on the steps.

The indent of the different lines guide the eye to keep track of the current scope of **if**-blocks or loops like **for** or **while**. For the current example, the vertical line below the **while** keyword and the indent of the expressions until the **end while** keyword point out that these commands are performed repetitively until the condition for stopping the loop is met.

Note that steps are given in a general textual descriptive version to avoid too complicated statements that occur when one gives a detailed version close to source code. In the next sections, some steps are described in more detail, which is why this introduction is kept short.

4.3 The KMC Program in Detail

Since this work does not contain an introduction to programming languages, the programmatic procedures are presented in an easy readable way. More complex parts like the algorithms for event selection and updates are expressed in pseudo code. This strategy makes sure that the reader is spared uninteresting details concerning the implementation. Expressions that refer to names in the source code are written using an `equidistant` font to prevent misunderstandings.

4.3.1 General Structure

The program uses object-oriented design[2] and complies with a coding standard[3] for C++.

To give an overview, fig. 14 shows a graphical representation of the objects the program is structured in. Basically, there are three important objects:

Solver class `KMCSolver`: Contains the actual (rejection-free) KMC algorithm for selecting an event to execute.

Event processing class `EventProcessor`: Keeps track of the events in the system, executes the events, and performs the update after changes in the system.

Lattice class `Lattice`: Represents the grid on which the particles are positioned and also contains a list of positions for all particles. This class is abstract[4], enforcing the user to implement it as a specific lattice (e.q. square lattice, hexagonal lattice). The class `SquareLattice` is such an implementation.

This structure has the advantage that these objects can be implemented more or less independently from each other. In fact, the KMC algorithm does not make a difference between events that are happening on a square lattice and events that are happening on a system which is not described by a lattice (off-lattice methods). Therefore the `KMCSolver` just selects events according to a list that is provided by the `EventProcessor` and the corresponding probability that is dependent on the energies and temperatures involved. For this reason, the solver class has the processing class as a member included in the object. However, the `EventProcessor` class also does not care if the underlying lattice is a e.g. square lattice or a hexagonal one, because the inherited `Lattice` class has all the functionality included that makes a difference between certain lattice types. For all the three levels, there are communication functions which provide the other classes with information that is crucial for the program to work. For instance, the `EventProcessor` needs to know what neighbors a particle has, which only the `Lattice` class can know. Therefore a function exists that provides the `EventProcessor` with this information, but the way this is done is part of the `Lattice` class and is none of the event processor's business.

This way the programmer can exchange these three classes with new objects that may have extended functions without changing the other two objects, as long as these extensions do not go beyond the current scope of the desired applications.

[2] One approach to software design, which uses objects that interact with each other. This approach can be seen as alternative to the one of sequential execution of operations [87].

[3] List of rules for usage and formatting of source code. Used in lecture "Software Development Practical Exercises" on the Institute of Information Systems and Computer Media (IICM) of the TU Graz [88]

[4] In C++, *abstract* classes are objects that cannot be used right away, but have to be *derived* by a subclass, meaning that they are used to create another object (the subclass) which inherits all properties of the parent class and extends its functionality.

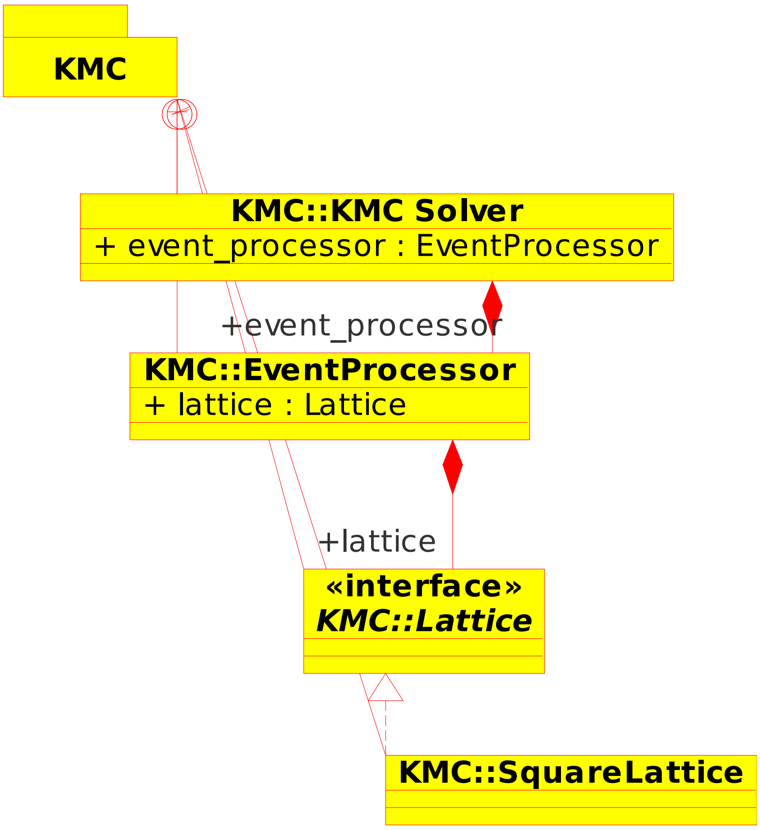


Figure 14: Class diagram of the KMC program (without member list). All classes are part of the package named *KMC*. The solver class contains the event processor and that in turn contains the lattice class. This lattice class has to be implemented to be e.g. a square lattice class, which is derived from the lattice class. See sec. 4.3 for details.

4.3.2 The Main Program

The execution of the KMC application starts with the creation of the `Lattice`, `EventProcessor` and `KMCSolver` object. The order of creation is important, since the `EventProcessor` cannot work properly without having a lattice supplied, and the `KMCSolver` needs a properly configured `EventProcessor` class in the same way. After these objects are instantiated, the actual simulation is started. Finally, when the run finishes, which can be for different reasons, the output is generated and the program stops.

Note that in the C++ implementation, the structure of the main KMC application is a little bit different than represented in the condensed procedure here. The resulting functionality is still the same, but one cannot compare the `main()` method of the C++ source code to the following procedure:

The first line *lattice* ← createSquareLattice(L) means that the object created by the

KMC main procedure

```
1: procedure KMC($L$, $F$, $t_{max}$, $J$, $Q$, $T$, $\nu$, $i$, $m$)
2:         $lattice$ ← createSquareLattice(L);
3:         $processor$ ← createEventProcessor(lattice, i, m);
4:         $solver$ ← createKMCSolver(processor);
5:         $solver$.simulate($t_{max}$,$F$,$J$,$Q$,$T$,$\nu$);
6:         return lattice.generateOutput();    ▷ matrix with value 1 at occupied positions and 0 otherwise
7: end procedure
```

function createSquareLattice() with parameter L is stored in a variable named *lattice*. The next two lines basically do the same. The fourth line *solver*.simulate(t_{max},F,J,Q,T,ν); means that a function simulate(...) of the object *solver* is called with the specified parameters. During the last step, the state of the lattice after simulation is generated by the function generateOutput() and then returned to the command line .

One can clearly see that the input parameters of the program are passed to the underlying functions exactly when they are needed. The square lattice object can be created with only the knowledge of the lattice size, because the amount memory the program will use to represent this object can already be calculated.

The event processor needs the lattice object to prepare for calculating the events and also the critical island size i and the growth mode m, since these two parameters make a huge difference in event calculations.

The solver object only needs the processor to be created. When the actual simulation starts, the other input parameters of the program are used to perform the calculations. Therefore the object's creation and existence are not directly coupled with the simulation. This way, after simulating with a specific choice of parameters, another simulation can be performed with the same object using other parameters, as long as the critical island size and the growth mode m do not change. If one of them does, also the event processor has to be created with the new parameters and provided to the solver. However, in this case one can still use the same lattice object. Consequently, whatever actions are performed on the lattice, it is easy to use the results of a simulation as input for another one.

4.3.3 The Event Processor Class

As the intermediate instance between the KMC solver and the lattice object, the event processor keeps track of the events possible in a system. Besides functions that help completing the interface between the object, which will not be mentioned here, the main functions are the ones which create, delete, execute events in the list of events. However, the core of the object is the function calcPossEvents(p_{num}) that calculates the possible events that can happen around a particle. As mentioned above, at creation time of the `EventProcessor` object, the growth mode m is known. In this program, there are two different growth modes available to choose:

(Totally) reversible mode ($m = 0$)**:** At all time it is in principle possible for a particle to move away from the edge of an island and therefore become a monomer again. Since the calculation of the probability (Transition State Theory see sec. 3.1.1) does not take into account the final state of the particle, there is no difference between movements along and away from the rim of the island. This means that all these movements happen with the same rate, even if this may be physical nonsense. Moreover, if the probability of one of these movements is quite low since temperature is already low, or the binding energy is high, these events will be found by the algorithm and saved because strictly speaking they could happen.

Only edge-hopping mode ($m = 1$) **:** As soon as the critical island size i is reached, only movements of particles along the rim of the island are allowed. That is actually easier said than done because, as seen below, the algorithm has to take into account a lot of cases, where simple conditions that come to one's mind that restrict the movements are not enough to ensure the correctness of the algorithm.

Whenever in this thesis a critical nucleus size i is specified for a simulation this implies that the mode $m = 1$ is used.

The description of the algorithms is done using a square lattice for simplicity.

Algorithmic description of (totally) reversible mode

First, the algorithm that performs the event search for the first mode is given:

Algorithm: calcPossEvents() for totally reversible case ($m = 0$)

```
procedure calcPossEvents($p_{num}$)    ▷ $p_{num}$ ...number (name) of this particle
    // check if there are neighbors
    if particle has no neighbors then
        // particle is a monomer
        $event\_type \leftarrow 0$;
        for all directions $i$ do
            if event is absent or wrong then
                // delete event if one with different type is present
                if event exists then
                    deleteEvent($p_{num}$, $i$)
                end if
                addEvent($p_{num}$, $event\_type$, $i$);
            end if
        end for
    else
        // particle is part of a cluster...
        $event\_type \leftarrow num\_neighbors$;    ▷ event type is numbered by number of neighbors
        // find free positions and add event to list
        for all directions $i$ do
            // add same event if not already present and if neighboring position is free
            if neighboring position is free and event is absent or wrong then
                // delete event if one with different type is present
                if event exists then
                    deleteEvent($p_{num}$, $i$)
                end if
                addEvent($p_{num}$, $event\_type$, $i$);
            end if
            // delete event if there is an event saved for it, but neighboring position is occupied
            if neighboring position is occupied and event exists then
                deleteEvent($p_{num}$, $i$)
            end if
        end for
    end if
end procedure
```

The part which performs the initializations is not mentioned here, and simply gathers the data from the lattice object to get knowledge about the surroundings of the particle, and gets the list of events that are currently in memory about the particle.

Fig. 15 helps to clarify the usage of the different directions on a square lattice. A

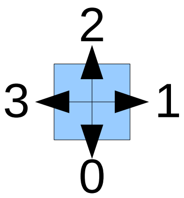


Figure 15: Different directions possible for a monomer on a square lattice, numbered from 0 to 3

particle can move in four directions if it has no neighbors at these positions.

The algorithm basically distinguishes between the two cases of whether the particle has a neighbor or is a monomer. If it is a monomer, the four directions in which it could possibly move are checked for events that may be in memory, which is done iterating over all directions. If they are of the right type already, nothing happens, since the list is correct as it is. If otherwise there is a deviation, the old one gets deleted and new ones are added.

A little bit more complicated, however, is the case of when the particle is part of a cluster. When one counts the number of different events than can occur, depending on the number of neighbors for the case $m = 0$, one finds that there can be 5 different types for a square lattice (no neighbor and 1,2,3,4 neighbors). Therefore one can also name the events from 0 to 4, which makes handling these variables easy (see line 17). Since there are occupied positions nearby, it has to be checked if there are maybe events from before that would point to these positions and hence have to be deleted. This keeps memory tidy and consistent. For an example of the working principle and of the way the memory is organized, see sec. 4.4.

For the second case, the way of adding/deleting the event is basically the same as for the first case. The reason why the two parts are separated, even though they are more or less similar except for some checks, is that small speedups for the monomer case boosts the whole efficiency of the program, since this case appears so often when there are a lot of lonely particles.

Algorithmic Description of Edge-Hopping Case ($m = 1$)

The pseudo-code description can be found in appendix A.

In fact, this algorithm is quite complicated both in the principal steps it does as well as in the computational implementation.

Firstly, one has to check if the critical nucleus size i has been reached when calculating the possible movements of a particle. If the cluster is smaller or equal to i, the particles may still leave the cluster, which enables them to hop around a corner without incorporating special treatment by the event processor, since it can do that in two consecutive steps. This is more or less similar to the case of $m = 0$ and the four directions $0 \ldots 3$ are still enough to process all events. However, if the cluster is stable, one has to ensure that when hopping around the corner, the particle does not become a monomer after the first step. This means that the algorithm has to save the two consecutive steps as a single event with another probability that can be calculated using the two single step rates. Therefore, 4 further directions are introduced ($4 \ldots 7$) to provide the support for that kind of events. Fig. 16 shows these directions together with the ones for $m = 0$. Furthermore, fig. 17 shows an example of a lattice that contains some islands, and on top somewhere there is a monomer. On the right hand side the directions that are used by the algorithm are indicated, where the additional directions for edge hopping are also present in red, but not allowed as long as the particle is not part of a stable cluster. A special cut-out is indicated and drawn again on the right hand side, where the edge particle (E) at the bottom should be able to change its position according to the green arrows and is not allowed to move in the directions indicated in red.

Moreover, when a particle changes its position inside the cluster, one has to ensure that after the step the cluster still has the same size. If the algorithm only checks if the particle still has a neighbor in its final position, but not to which island that neighbor belongs, the particle could change to a different cluster, or even if the peculiar case could happen that one part of the cluster separates from the parent one through quite a convoluted process.

To address this issues, the algorithm “depth first search” [75] used extensively in computer science is used, which on the one hand, calculates the cluster size in very high speed and, on the other hand, makes it easy to mark particles if they have already been visited. It treats the cluster as a graph, which is basically a bunch of nodes and edges: A node is a particle and the connection from the nodes to one another are the edges. Very important concerning graph theory are edges called “bridges”. A bridge is a connection between nodes that leads to disconnection of a part of the graph when removed. In this work, a bridge particle is defined as one that causes disconnection of the cluster if it moves. Fig. 17 shows a particle (B) which would disconnect the particle at the bottom if it moves anywhere. and therefore forms a bridge. The algorithm has to recognize this restriction and must not allow any such events to happen.

In short, the algorithm works as follows: If the particle is a monomer, proceed as for case of $m = 0$. If it is not, first check if the cluster is big enough to be stable. If it is not, proceed as for $m = 0$, otherwise check if the particle forms a bridge (done by lattice class). If all conditions are met, check all 8 directions for valid edge hop events and maybe add them to the list. Furthermore, deletion of old events has to be done carefully if particles happen to change from normal edge particles to bridge particles.

In fact, these checks are quite complicated concerning the implementation in the actual programming language. The pseudo code in appendix A only mentions what happens, omitting a lot of source code that is not interesting from a physicist’s point of

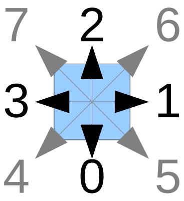


Figure 16: Different directions possible for a particle in the partially reversible edge-hopping mode. The directions in black indicate the movement directions a monomer or a particle belonging to an unstable cluster would be allowed to follow $(0\ldots3)$. The directions in gray show the additional directions $(4\ldots7)$ that are introduced to perform edge hopping on a stable cluster as described in sec. 4.3.3.

view.

Event Execution and Update Procedures

Another important function performs the execution of an event by telling the lattice which particle to move where and updates the list after execution. There is a lot of optimization done regarding these processes, because computational efficiency can drop severely if things are done multiple times without reason.

4.3.4 The Lattice Class

This object basically contains the arrays used to represent the lattice itself and the particles with the corresponding positions. The only complex algorithm part of this class is the one performing the calculation of island size (depthFirstSearch()), which will not be mentioned here. In fact, memory allocation and data structure representation forms the most part of the object. Moreover, as for the inherited class `SquareLattice`, information like neighboring particles, direction numbers and second nearest neighbor information is rapidly communicated to the event processing class, since this depends on the actual lattice type.

4.3.5 The KMC Solver Class

Finally, the class which contains the actual BKL-KMC algorithm is presented. It is quite slim, since it contains just the algorithm itself and the logic for deposition of particles on the surface. The latter one will be omitted here, leaving only the implementation of the KMC algorithm as described theoretically in sec. 3.3.4:

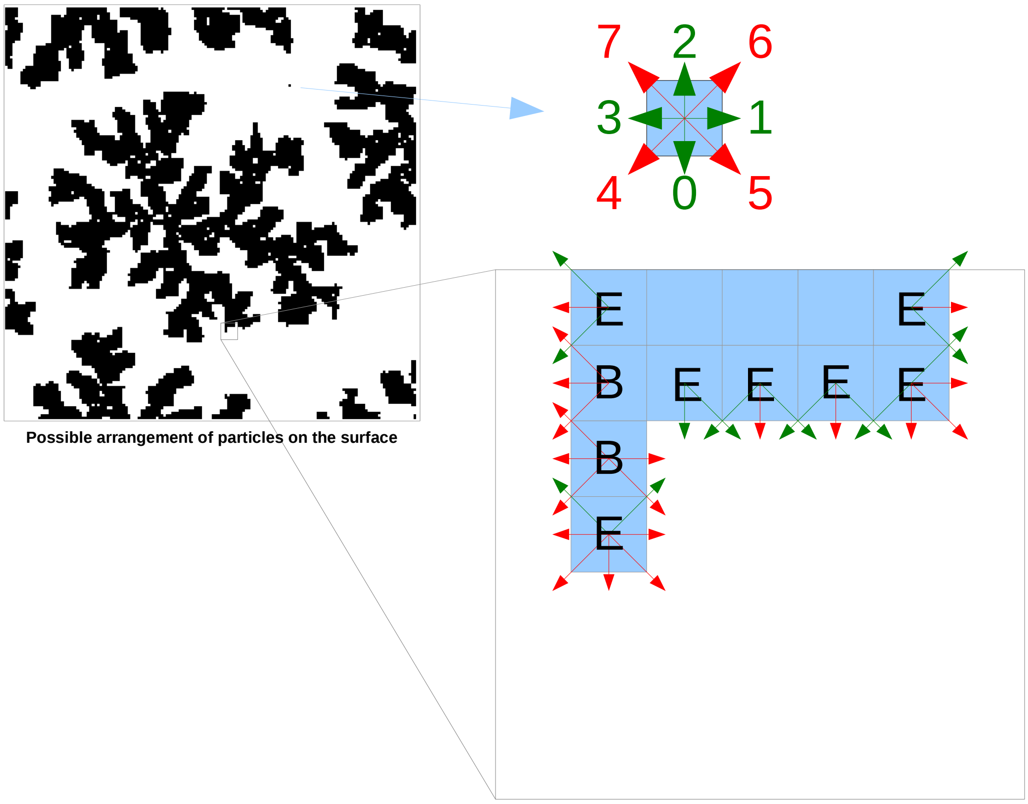


Figure 17: Illustration of the algorithmic circumstances in the case of the edge hopping algorithm ($m = 1$). At the top, all possible directions of movements of a monomer are indicated in green. The cut-out shows edge-particles (E) for which the algorithm allows certain events to happen. If the bridge particles (B) have moved, the edge particle at the bottom would be disconnected from the rest of the cluster. Therefore all events moving bridge particles are prohibited.

BKL-KMC Algorithm Implementation: simulate()

```
procedure SIMULATE($F$, $t_{max}$, $J$, $Q$, $T$,$\nu$)
    // $\Gamma$ is a vector containing the rates for all kinds of events
    $\Gamma[0] \leftarrow \nu \exp(-Q/kT)$;                    ▷ monomer diffusion rate
    // events for reversible approach $m = 0$
    for all nearest neighbor events $i \leftarrow 1$ to 4 do
        $\Gamma[i] = \nu \exp(-iJ/kT)$;
    end for
    // additional events for edge hopping approach $m = 1$
    for all next nearest neighbor events $j$ do
        $\Gamma[i_{max} + j]$=rate for next nearest neighbor event;
    end for
    // perform the loop
    while $t < t_{max}$ do
        // get the number of existent events in the system for every event type
        $n[] \leftarrow EventProcessor$.getEventDistribution();
        for all events $i$ do
            $\Gamma_{tot} \leftarrow \Gamma_{tot} + \Gamma[i] * n[i]$;
        end for
        // select process using random number
        $\rho \leftarrow$ random number $\in [0, 1)$;
        $j \leftarrow -1$;
        $R \leftarrow 0$;
        while $\rho\Gamma_{tot} \geq R$ do
            $j \leftarrow j + 1$;
            $R \leftarrow R + n[j] * \Gamma[j]$;
        end while
        $\rho \leftarrow$ integer random number $\in [1, n[j]]$;
        $EventProcessor$.executeEvent($j$, $m - 1$);
        $\Delta_t \leftarrow 1/(\Gamma_{tot} + FL^2)$;
        if $F > 0$ then
            $\rho \leftarrow$ random number $\in [0, 1)$;
            if $\rho < FL^2\Delta_t$ then
                deposition();
            end if
        end if
        $t \leftarrow t + \Delta_t$;
    end while
end procedure
```

The first part basically just calculates the rates that will not change any more until the simulation is complete. After that, the big loop works until the desired time limit has been reached. Important function calls have been highlighted.

Before the selection of the events during every cycle, the new event distribution has to be provided to the algorithm in order to select from a list of events that is up to date. The total rate can be calculated from it and in the next step one of the event types is selected. One of these events is then executed by the `EventProcessor` class and the time step is calculated. If there is a flux F specified, also deposition takes place during one of these steps according to the probability that a deposition event happens in that interval. Finally, the time is incremented by the time step.

4.4 Example: The Algorithm in Action

In addition to the described procedures above, a short example is given here to clarify further the working principle of the whole program.

Consider the situation as presented in fig. 18(a), where a total number of 5 particles is positioned somewhere on a lattice. Two of them are monomers (particle #2 and #3), two have one neighbor (particle #4 and #5), and one particle (particle #1) has two neighbors.

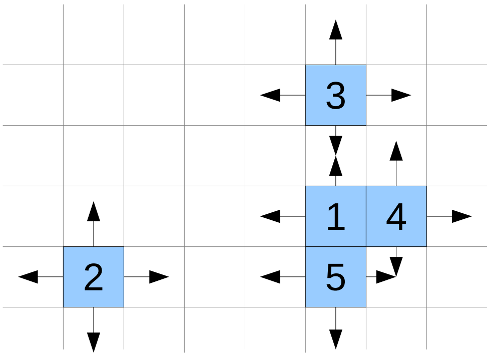
The event list on the right hand side lists the correct event information for this situation, as it is done by the `EventProcessor` class. Note that the list is not necessarily in an ordered state, since ordering would cost additional time. Particle #2 and #3 can move in all four directions. Therefore the list for event type 0 with rate Γ_0 is filled up with 8 events. In contrast to that, particle #1 can only move in direction 3 or 2, with a rate Γ_2, since it has two neighbors.

This information is acquired by the solver class and then used to select an event to execute. In fact, the selection depends on the parameters provided to the program and are chosen for this example as follows:

Parameter	Chosen Value
Temperature ...	$T = 300$ K
Monomer diffusion barrier ...	$Q = 0.1$ eV
Nearest neighbor binding energy ...	$J = 0.4$ eV
Attempt frequency ...	$\nu = 10^{14}$ /s

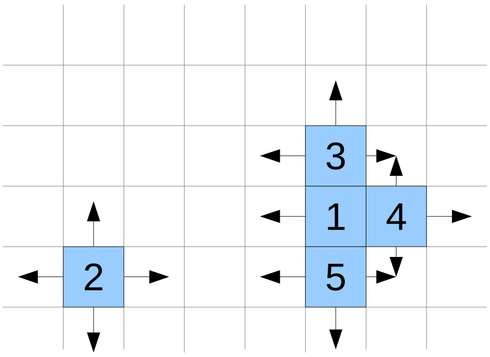
Using these values, the different rates can be calculated:

$$\Gamma_0 = \nu \exp(-Q/kT) \approx 2.0897 \cdot 10^{12}/s$$



Event Type	Index in List	1	2	3	4	5	6	7	8
event 0 (Γ_0)	particle	3	3	3	2	2	3	2	2
	direction	3	1	2	0	1	0	2	3
event 1 (Γ_1)	particle	4	4	4	5	5	5		
	direction	0	1	2	0	1	3		
event 2 (Γ_2)	particle	1	1						
	direction	3	2						
event 3 (Γ_3)	particle								
	direction								

(a) Situation before event execution



Event Type	Index in List	1	2	3	4	5	6	7	8	9
event 0 (Γ_0)	particle	2	2	2	2					
	direction	1	3	2	0					
event 1 (Γ_1)	particle	4	4	4	5	5	5	3	3	3
	direction	0	1	2	0	1	3	1	2	3
event 2 (Γ_2)	particle									
	direction									
event 3 (Γ_3)	particle	1								
	direction	3								

(b) Situation after event execution

Figure 18: Graphical representations of the particle positions and event lists for the example mentioned in sec. 4.4.

$$\Gamma_1 = \nu \exp(-J/kT) \approx 1.9068 \cdot 10^7/s$$

$$\Gamma_2 = \nu \exp(-2 \cdot J/kT) \approx 3.6357 \cdot 10^0/s$$

The total rates of the individual event types are as follows:

$$\Gamma_{0,tot} = 8 \cdot \Gamma_0 = 1.6717 \cdot 10^{13}/s$$

$$\Gamma_{1,tot} = 6 \cdot \Gamma_2 = 1.1441 \cdot 10^8/s$$

$$\Gamma_{2,tot} = 2 \cdot \Gamma_3 = 7.2714 \cdot 10^0/s$$

Finally, the overall rate results in:

$$\Gamma_{tot} = 1.6717 \cdot 10^{13}/s$$

Next, an event is selected using the following steps, as described by the n-Fold Method in sec. 3.3.4:

- Draw random number $u \in (0, 1]$ and scale to $\Gamma_{tot} = 1.6717 \cdot 10^{13} \Rightarrow$ e.g. $u = 0.5$ $\Rightarrow$ scaled to $u' = 8.3587 \cdot 10^{12}$
- Find out which event type to execute: $\Gamma_{0,tot} = 1.67 \cdot 10^{13} > u' \Rightarrow$ event type 0 is selected
- Find out which particular event of that type to execute: draw integer random number v between 1 and 8: e.g. $v = 6$
- Execute event with that number: as in the event list of fig. 18(a), particle #3 gets moved one step in direction 0
- Update events of moved particle and of all previous and new nearest neighbors: only particles #3 and #1 have to be updated

Fig. 18(b) shows the situation after the described cycle. Particle #1 now has three neighbors, therefore it can only move in direction 3 now. Only one monomer is left and there are now three particles with one neighbor. The order of the events of the monomer is a result of the deletion process: Every time an event is deleted, the list gets shorter by one and the last entry changes place with the one that gets deleted. Deletion is ordered by the directions, meaning that first, an event with direction 0 is tried to be deleted and after, in direction 1,2,3. The individual steps in this case are listed in table 2.

Table 2: Steps of deleting events of particle #3 from the list of events with rate Γ_0. Events indicated in red are deleted in the next line and the place is filled with the event at the end of the list. Events in green have just been moved in the current step.

Step	Index in List	1	2	3	4	5	6	7	8
0	particle	3	3	3	2	2	3	2	2
	direction	3	1	2	0	1	0	2	3
1	particle	3	3	3	2	2	2	2	
	direction	3	1	2	0	1	3	2	
2	particle	3	2	3	2	2	2		
	direction	3	2	2	0	1	3		
3	particle	3	2	2	2	2			
	direction	3	2	3	0	1			
4	particle	2	2	2	2				
	direction	1	3	2	0				

4.5 Parameter Studies - Program Verification

In spite of the exercised care when writing the program, it is self-evident to compare the output of the program with results published in literature. However, since in most publications space is limited and therefore usually algorithmic details are mentioned in a sketchy way rather than in every detail, one has to be careful when doing a comparison.

First, the reader is introduced to the way of presenting the results of simulations with the help of pictures of the particles on the lattice. Occupied positions are denoted by black pixels and free ones with white pixels. After that, a parameter study is given that looks at the change of island densities when changing the ratio between D and F.

4.5.1 Variation of Temperature

To begin with, a small study of the morphological results for the case of low and high temperature values is given. Fig. 19 shows an example for $L = 400$ and for two values of the temperature. In case (a) the temperature is very low ($T \rightarrow 0$) and the time-scale separation between processes with a higher and a lower barrier is several orders of magnitude. Consequently only the fastest processes are executed, which are the monomer diffusion events. The barrier for diffusion around the island rim is higher and therefore these events will be very rarely executed (hit-and-stick mode). This leads to a fractal island shapes as expected from Diffusion Limited Aggregation (DLA).

On the contrary, case (b) is an example of $T \rightarrow \infty$ which means that edge hopping processes are almost as likely to happen as monomer diffusion processes. On a square lattice, the energetically lowest state of the system is an island in the shape of a square. Then there are just 4 particles in the corner that have only two neighbors. In the figure one can see that the islands are not perfectly square-shaped. This is because fluctuations are very strong. On the one hand, fluctuations are necessary because without them the

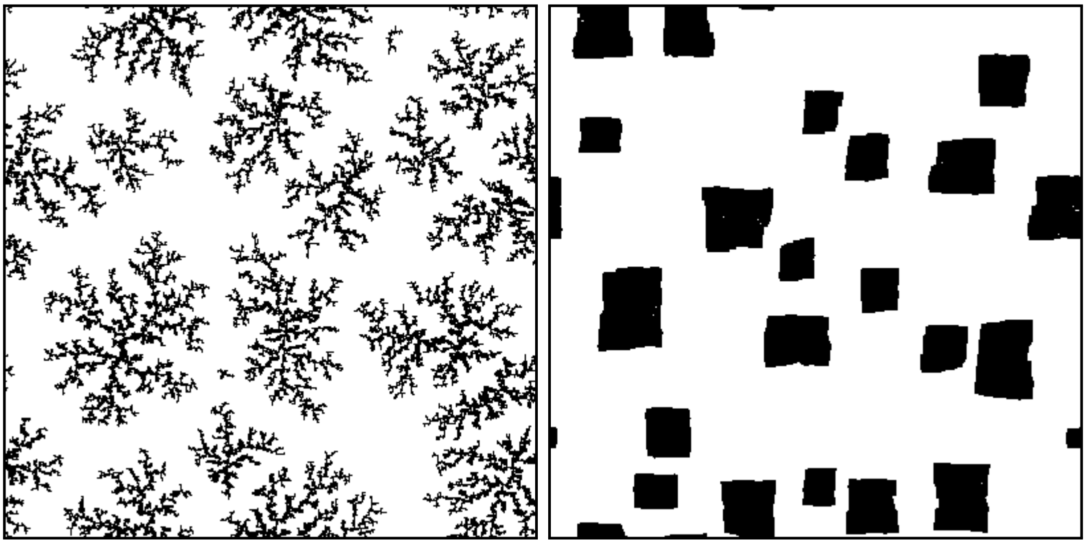


Figure 19: Comparison between island shapes of (a) low temperatures ($T \to 0$) and (b) high temperatures ($T \to \infty$). Parameters: $L = 400$, $kT = J/50$, $\theta_{max} = 0.3$ ML and $h_1/F = 10^8$

system would never be able to overcome certain energy states to reach the energetically lowest state. Dendritic islands have to undergo a lot of edge-hopping processes so that they get more compact. On the other hand, fluctuations also lead to deviations from the state of lowest energy.

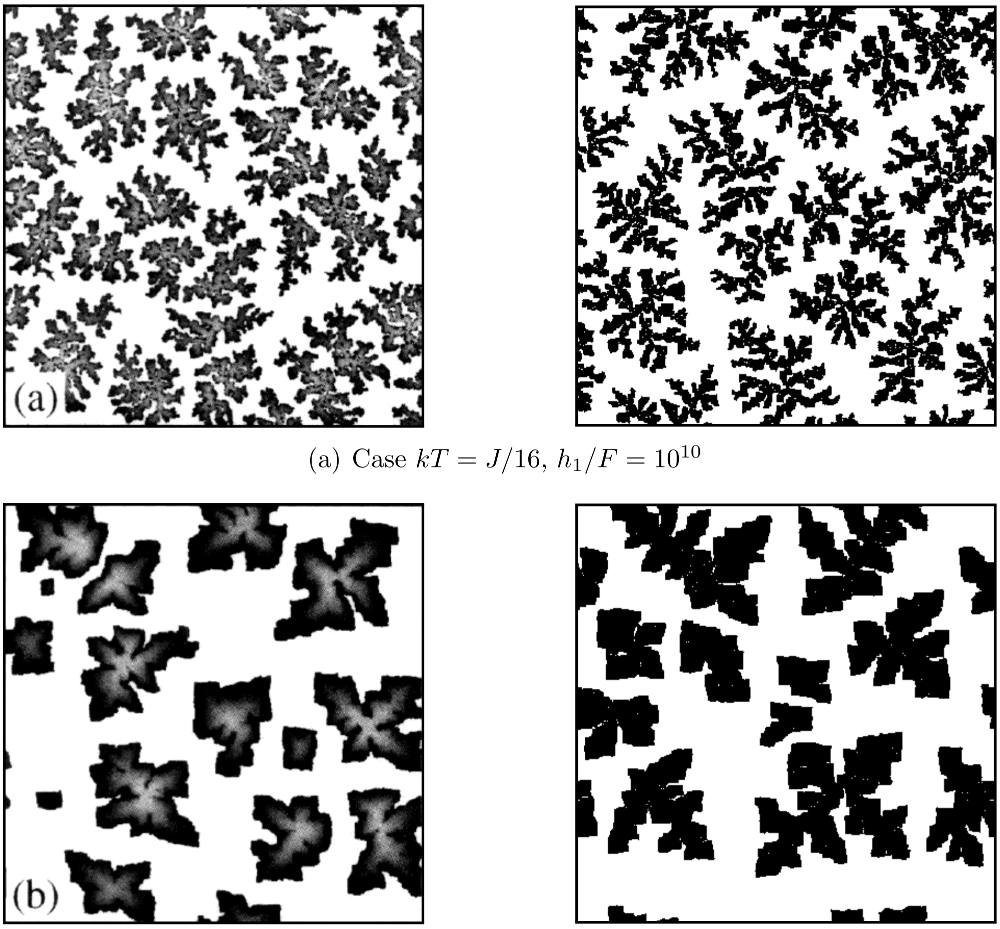
In fact, it takes the KMC program a very long time to simulate the latter case, since fluctuations are simulated that may be uninteresting as soon as a low energy state has been reached, even the islands are still growing. Moreover, with the totally reversible aggregation mode ($m = 0$) it would not be possible to reach such a state, since particles will leave the islands continuously and the system will suffer from severe stiffness issues.

However, it shall be states that in nature such processes may be likely to undergo such states of thermal noise, but with simple KMC algorithms it is not possible to simulate such cases close to thermal equilibrium.

4.5.2 Morphological Comparison with Literature

As already mentioned in the section about the problem definition (sec. 2.4), the ideas of Bales and Chrzan [26] were used in a modified way as a basis of the BKL KMC program. Since they did not have the critical nucleus size incorporated in their model as a parameter, here only the case of $i = 1$ can be compared to their work. Fig. 20 shows their published figures on the left hand side, and the results of the KMC program of this thesis.

When one looks in detail at the appearance of the islands edges, one can see that the morphological appearance is quite similar. However, the total number of islands is a little bit higher in their case, which can be due to statistical fluctuations, and also may be an effect of slightly different working principles of the algorithms. They only state



(a) Case $kT = J/16$, $h_1/F = 10^{10}$

(b) Case $kT = J/14$, $h_1/F = 8 \cdot 10^{10}$

Figure 20: Comparison between the published data by Bales and Chrzan [26] (left column) and the KMC program of this work (right column) for the parameters lattice size $L = 400$, coverage $\theta = 0.4$, and for different values of the ratio h_1/F and $\beta = kT$ as specified in the caption of the individual cases.

that detachment of monomers from a cluster is not allowed, but they do not give details if they suppress separation of clusters as done by the approach in this work (sec. 4.3.3), which would lead to an increased total number of islands.

4.5.3 Justification of the Venables Equation - Variation of Critical Nucleus Size

More importantly than the morphological behavior, however, is that the statistics show the right behavior according to the theory. A nice check offers the Venables equation eq. (16), since the important parameters flux F as rate R, $D \sim h_1$ as attempt frequency ν, and the critical cluster size i are reflected in the formula. When again using the solid-on-solid model (sec. 3.3.8), the final island density of a simulation should scale according to

$$N = C_i \cdot \left(\frac{h_1}{F}\right)^{-i/(i+2)} \qquad \text{[ML]}$$

where C_i is the constant reflecting the other dependencies of the Venables equation, which were not altered in this study except for i, which is why there is the index i.

In order to verify the behavior of the simulations with respect to this formula, simulations for the following ranges of the different parameters have been done:

$$\begin{aligned} h_1/F &= 10^5 - 10^7 \\ i &= 1 - 4 \end{aligned}$$

The other parameters have been chosen to $Q = 0.01$ eV, $J = 0.1$ eV, $T = 300$ K, $\theta_{max} = 0.1$, and $L = 1000$.

The case $i = 1$ corresponds to totally irreversible aggregation because even a dimer will never dissolve. It is the computationally least expensive one and has been carried out for the whole range of h_1/F specified above. Fig. 21 shows the results of the simulations and also the fit with the resulting parameter $C_1 = 5.946 \cdot 10^5$. The other three cases for $i = 2, 3, 4$ are compiled in one plot (fig. 22) since the values for h_1/F in these cases are all the same. One can see that when increasing i the island number is severely reduced. The fit shows the changing exponent $i/(i + 2)$ of the Venables equation clearly since the curves get flatter with increasing i. As the value of h_1/F approaches higher values, the number of islands gets reduced until there are not enough islands on the lattice left and boundary effects appear. This is the reason why the last point at $h_1/F = 3.2 \cdot 10^6$ for $i = 4$ could not be determined, besides the fact that the computational time increases dramatically. The values for the fit curves[5] are $C_2 = 5.946 \cdot 10^5$, $C_3 = 8.654 \cdot 10^5$, and $C_4 = 1.163 \cdot 10^6$.

[5] Note that the curves all have an offset which is also attributed to boundary effects. However, these deviations are within statistical error and therefore the offset is omitted here.

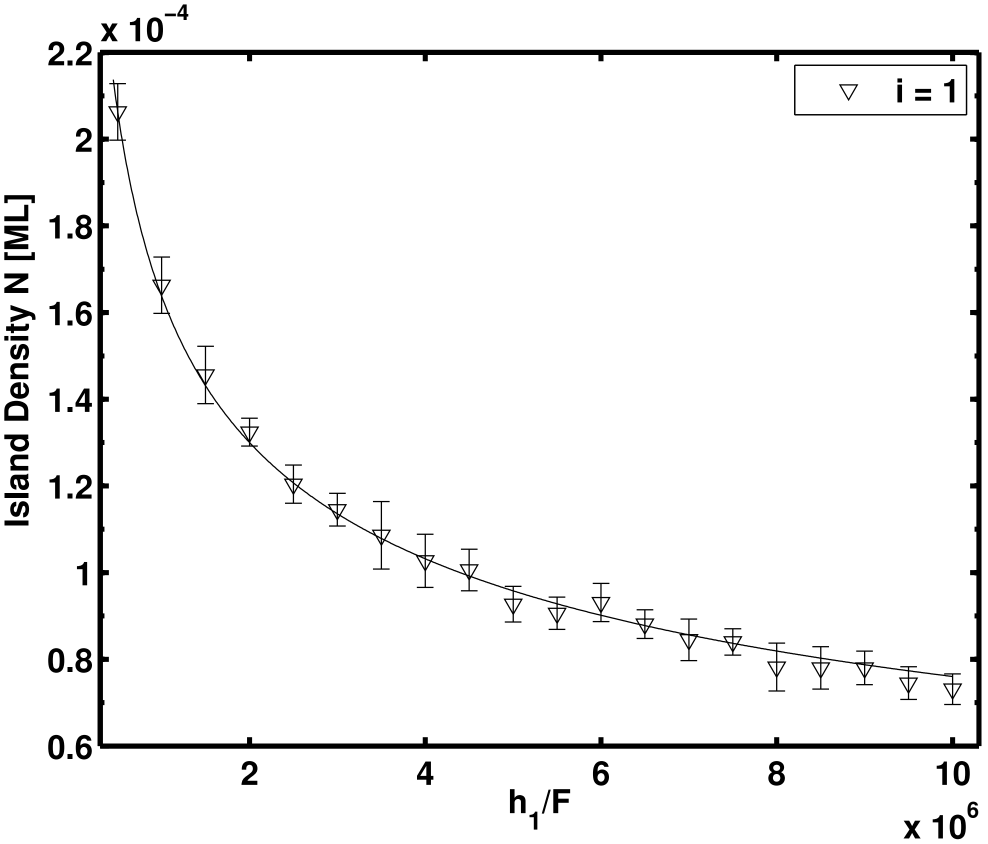


Figure 21: Island densities N as a function of h_1/F for a critical island size $i = 1$ resulting from KMC simulations using the solid-on-solid (SOS) model. The solid line is the result of the fit as specified in sec. 4.5.3.

4.5.4 Evolution of Monomer- and Island Densities

Finally, a comparison of the simulations with the monomer- and island density evolutions as depicted in fig. 10 and fig. 11 is done. The program was justified to track the island and monomer densities during the simulation. Due to limited computer power this expensive calculations have been done only for the three cases $i = 1, 2, 3$ and with the parameters $h_1/F = 10^5$, $\theta_{max} = 0.3$ ML, $L = 400$ and $kT = J/18$. Note that this choice is not aimed in any way at reflecting real physical situations but is a compromise between computer time and system size. Too small parameters for h_1/F or L would lead to situation where there is only noise in the curves because there are too few particles on the surface at the same time.

Fig. 23 shows the double-logarithmic plot of the time evolution of all three cases $i = 1, 2, 3$ for both the island and monomer densities. Due to reasons of fluctuations and the discrete nature of KMC simulations, the same plot is also shown in fig. 24 with linear scaling.

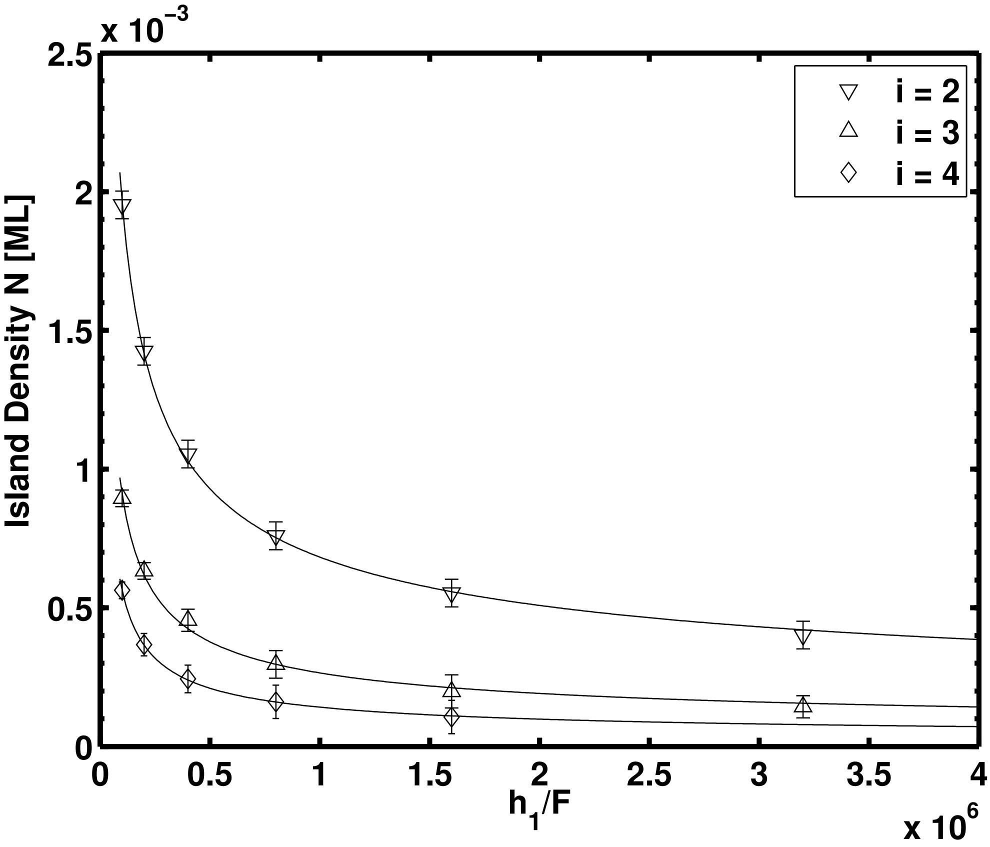


Figure 22: Island densities N as a function of h_1/F for different values of the critical island size i that are indicated in the figure resulting from KMC simulations using the solid-on-solid (SOS) model. The solid lines are the results of the fits as specified in sec. 4.5.3.

Interesting is that for higher values of i the monomer density rises to higher values without having a lot of islands nucleated until it drops even drastically when the final number of islands is reached. For all three cases one can see that already some coalescence of islands happens at the end of the simulation, since the island number decreases. For $i = 1$ the monomer density still rises a bit when there are already some islands on the surface. For matter of completeness and to satisfy the reader the three lattices are plotted in fig. 25.

4.6 Quasicontinuum Monte Carlo Method

The current implementation done for this thesis is still in an experimental state and therefore only the main working principle is presented.

In fact the program contains essentially the same parts as the KMC program described in sec. 4.3, and additionally contains a solving engine for the diffusion equations. Besides the lattice that contains the discrete positioned particles (`Lattice` class of KMC program)

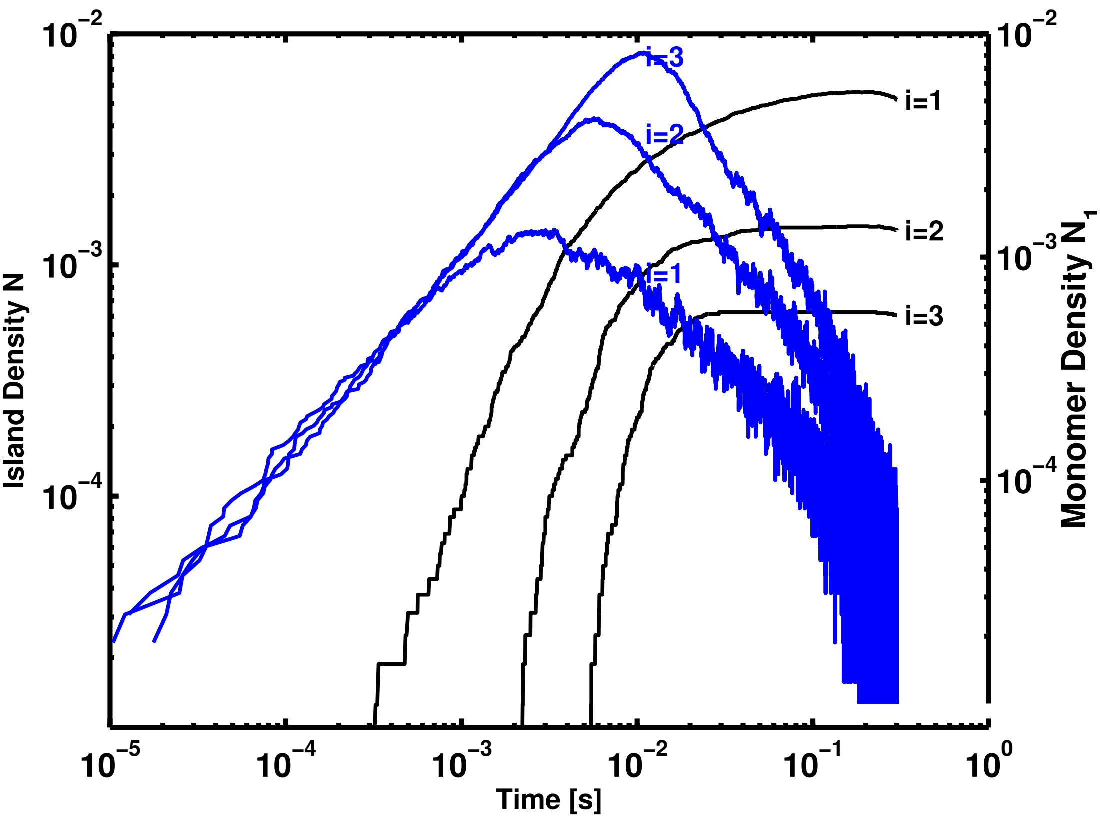


Figure 23: Log-plot of monomer and island density time evolution during a KMC simulation. The noise on the right-hand side is due to big time steps and therefore big fluctuations of monomers during these steps. Parameters: $h_1/F = 10^5$, $\theta_{max} = 0.3$ ML, $L = 400$ and $kT = J/18$

another matrix representing the lattice that contains the monomer density values is kept in memory. The event processing class is used to keep track of events going on exclusively on the islands, but the monomers are not any more reflected in the list. This saves a lot of computer time.

For every time step Δt a fraction of the flux F becomes added to the whole monomer density lattice and then the monomer concentration around the islands is calculated. After that either the next time step is taken, or a particle gets added to the island rim.

Unfortunately, the nucleation behavior using the model mentioned in sec. 3.4 currently is unreliable, since the final island density observed after a simulation turns out to be dependent on the time step. Yet it is unclear if this is an effect of systematic errors concerning the theory or if there is a problem in the implementation. Therefore the nucleation behavior is not further discussed here.

Moreover because the program currently uses an implicit finite difference scheme, lattice sizes are limited to about $L = 120$ at the time, because the memory needed for

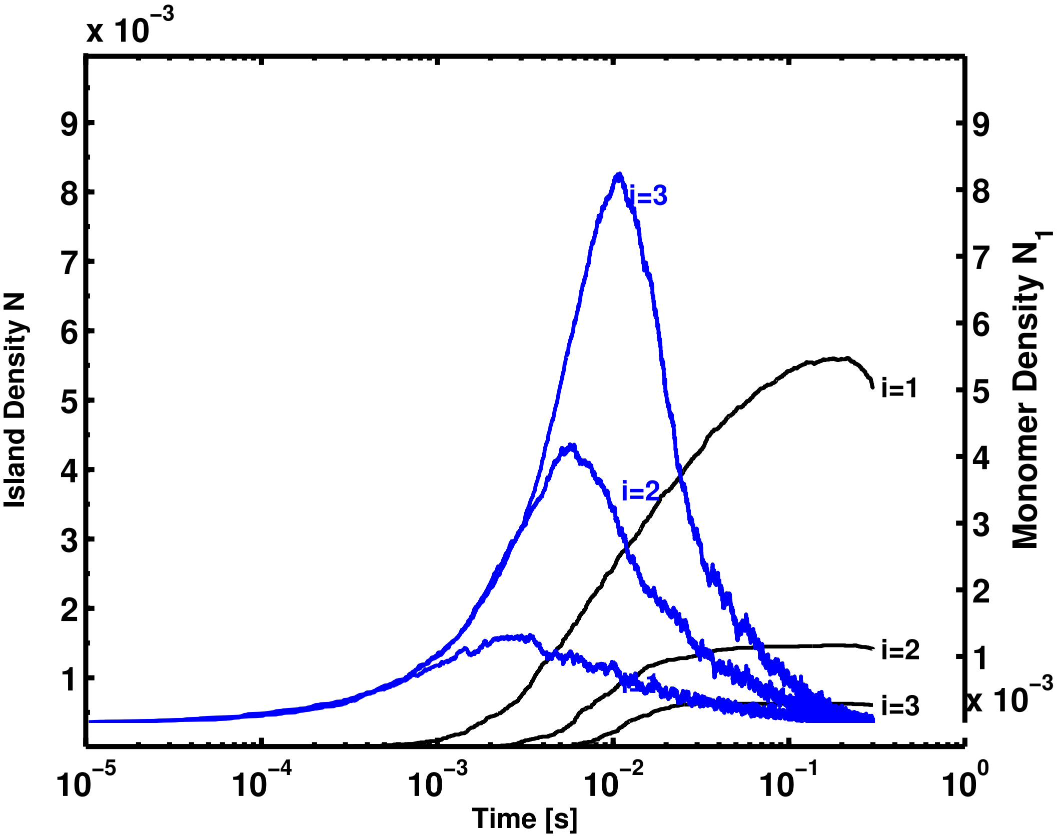


Figure 24: Plot of monomer and island density time evolution during a KMC simulation. Parameters: $h_1/F = 10^5$, $\theta_{max} = 0.3$ ML, $L = 400$ and $kT = J/18$

the matrix M which is used to perform the solving step scales like L^4. There are better methods to solve these problems with extremely reduced memory requirements because the matrix M is sparse, but they have not been implemented yet.

Regardless of these limitations a small example which shows qualitatively the promising behavior of the approach when it comes to attachment limited systems is shown here.

Consider a single island growing in hit-and-stick mode resulting in a fractal shape. The resulting island and the final monomer density around the island can be seen in fig. 26(a). Once a particle is attached it cannot move any more. This corresponds to 100 % attachment probability.

The probability for attachment is different for the other two cases 26(b) with 10 % and 26(c) with 1 %. What can be clearly seen is that the islands become more and more compact even though no edge diffusion is allowed. This is because the monomers now flow along the rim with a small distance and can attach with a smaller probability, and therefore are more likely to reach the area between the branches of the islands.

Moreover, when specifying a small attachment probability the overall density of

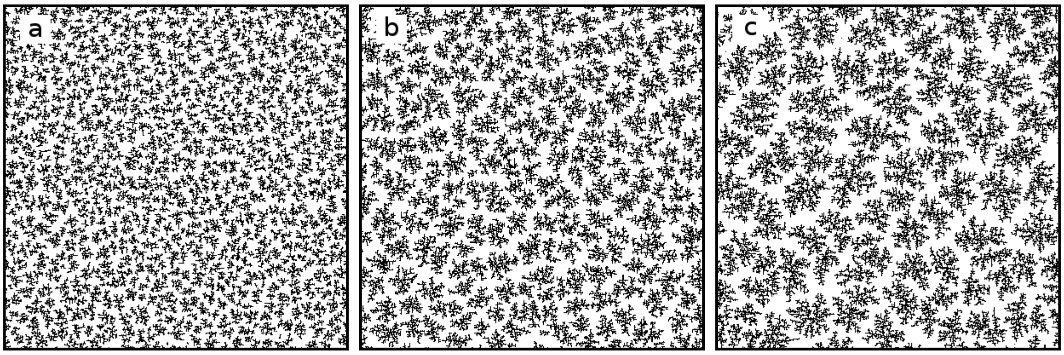


Figure 25: Lattice images of the three cases of fig. 23 and 24. (a) $i = 1$, (b) $i = 2$, and (c) $i = 3$. Parameters: $h_1/F = 10^5$, $\theta_{max} = 0.3$ ML, $L = 400$ and $kT = J/18$

monomers around an island grows and stays very high until the flux F is turned off.

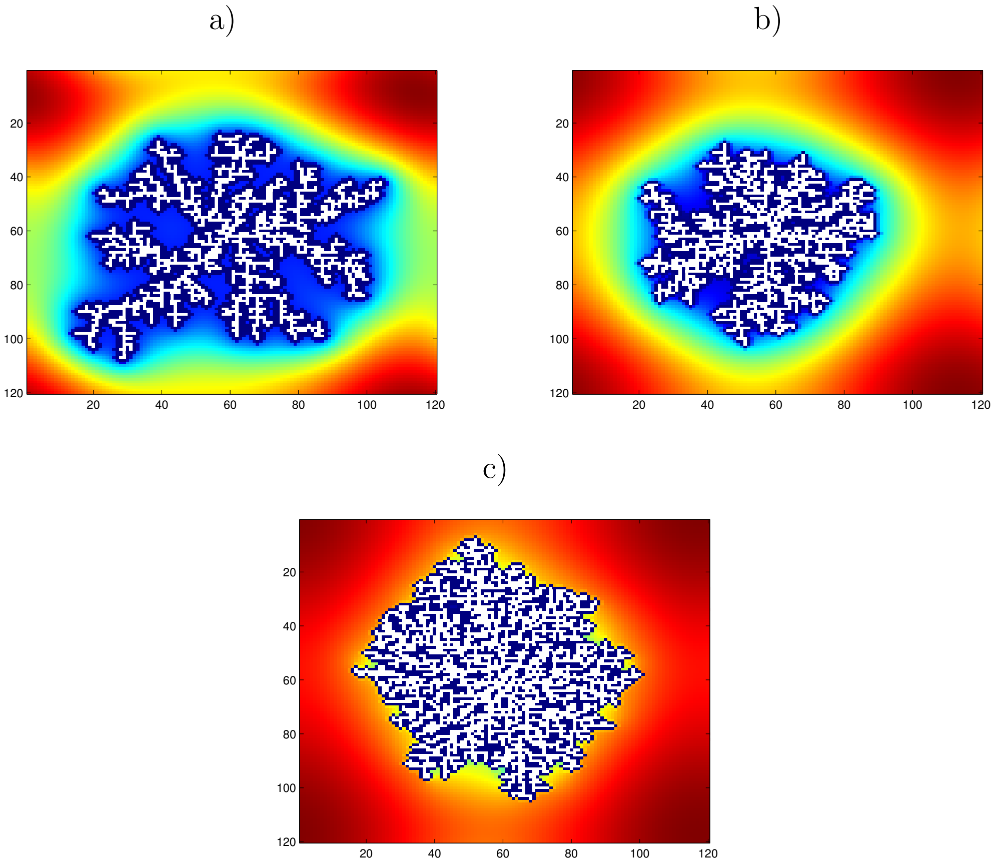


Figure 26: Island growth in hit-and-stick mode for different rates of diffusion to the edge simulated by Quasicontinuum Monte Carlo (QCMC) method. The white pixels denote fixed particle positions. The colored area around is the monomer density where red is the maximum and blue the minimum. In case (a) there is no attachment barrier, which corresponds to the classical DLA case. In (b) the rate of attachment is 10% and in (c) 1% of the terrace monomer diffusion rate. Lattice size $L = 120$

5 Application of KMC Simulations to the Case of Adsorbate Induced Subsequent Nucleation of 6P

This section now focuses on the results that have been gathered by using the program to perform the simulations whose objectives where defined in 2.4.

To begin with, the program was tested for performance in order to find out the maximum lattice size that could be reached while still getting reasonable statistical data in acceptable time. It turned out that the limits for the parameters are as follows:

- Critical cluster sizes up to a maximum of 8 particles could be performed only on lattice sizes not smaller than $L = 1000$ because of boundary effects. When the simulation finishes, there are very few islands on the surface and the periodic boundary conditions (PBC) lead to wrong island densities.

- Lattice sizes $L > 1000$ and/or critical cluster sizes $i > 8$ are computationally too expensive and therefore too time consuming.

Consequently, these limits have to be taken into account. If one particle needs approximately a space of 1 nm^2 when standing in an island, a lattice length of $L = 1000$ corresponds to an equivalent distance of about 1 μm. This is comparable to the AFM images (see fig. 5(a)).

The parameters for the hopping rates (eq. (1)-(2)) were chosen as follows: From the AFM measurements, because of certain issues when evaluating the island sizes, the real coverage cannot be estimated more precicely than being in the range of 0.02 ML $< \theta <$ 0.05 ML. The attempt frequency $\nu = 10^{13}\ s^{-1}$ is assumed to be the same for both monomer diffusion and edge diffusion. We know that the monomer diffusion barrier is about the size of $Q = 0.05$ eV, as we know from a MD calculation [89].

When choosing an appropriate value for the edge hopping barrier E, we can only guess that it will have a value close to the one of Q. Very important for the case of subsequent nucleation, this value will drastically influence the island shape. For values of $E > Q$, the islands will turn out to be quite fractal. In fact, it turns out that islands become totally compact if the values are about the same $E = Q$ or if E is smaller, because nucleation is much faster than compacting of the islands.

Consequently, one can argue that one can expect a difference of island density for the following cases:

1. Islands become compact after the island density saturates ($Q > E$), meaning that compacting happens when nucleation has already stopped.

2. Islands become compact shortly after nucleation $E \lesssim Q$, meaning that other islands may still form while others are compact already.

Fractal islands have a slightly bigger capture zone, since they cover a bigger area in which monomers can be caught. Hence, the first case should lead to a reduced island number, since the existing fractal ones will reduce the monomer number more severely than if they were compact already.

However, comparison with a lot of simulations show that the number density only increases at most for about 10% when edge hopping play a role shortly after nucleation. Fig. 27 shows examples for fractal and compact islands, where only the energy barriers for edge diffusion are different: $E >>$ for fig. 27(a) and $E = Q = 0.05$ eV for fig. 27(b).

Moreover, in the work by Bales and Chrzan [26] the same weak dependence of the island shape on the island density has been found for their case of non-zero flux F, and there are studies made by other groups [42, 90] revealing also a weak influence on the island size distributions.

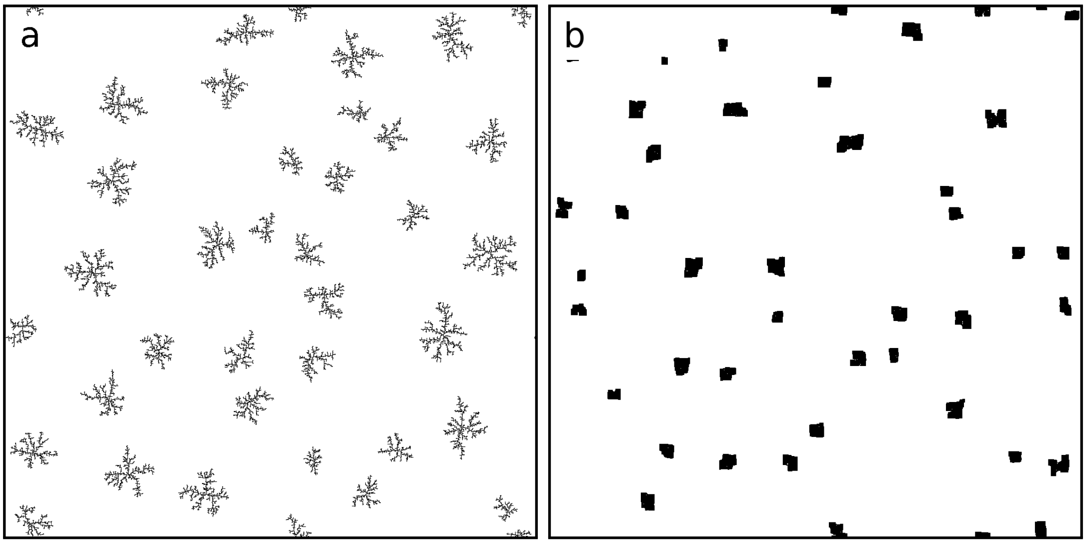


Figure 27: Comparison between (a) fractal islands without edge hopping and (b) compact islands with $Q = E = 0.05$ eV for the KMC simulations mentioned in sec. 5. In case (b) compacting happened already when the nucleation of islands was not yet finished. Parameters: lattice size $L = 1000$, coverage $\theta = 0.03$ ML, critical cluster size $i = 7$

Thus, in order to safe calculation time all the simulations for statistical evaluation have been performed without edge hopping. Therefore, only nucleation remains as expensive process to simulate, since after that the simple hit-and stick mechanism leads to quick elimination of the remaining monomers and no time is wasted for compacting.

5.1 Graphical Description of Time Evolution

In order to give an idea of how the simulation process proceeds, the time evolution of the case which is presented in fig. 27b can be seen in fig. 28a-f. Starting with monomers distributed all over the surface, which looks more or less like thermal noise (fig. 28a), stable nuclei start to form (fig. 28b). Until the final island density is reached, nucleation happens continuously (fig. 28c-d), but this decreases the monomer density around these islands. In fact, as we know from the Walton relation (eq. (7)) local nucleation probability scales like:

$$\frac{dN_{isl}}{dt} \sim N_1^{i+1}$$

Consequently, formation of a denuded zone around an island will severely reduce the probability that a new stable island forms. Thus, when a certain number if islands is reached, nucleation stops, as it did already in fig. 28e. Finally, simply attachment to existing islands happens until the monomers are all gone (fig. 28f).

Additionally, the development of island and monomer density have been tracked for exemplary cases. The results are plotted in fig. 29. One can clearly see that the final island density is strongly dependent on the value of the critical island size i. Nucleation starts more or less at the beginning of the simulation, but is the strongest after certain time, depending again on the value of i. Correspondingly, the monomer densities drop later with higher i.

This behavior explains the appearance of denuded zones as mentioned in sec. 2.3.2, since the time scale for diffusion events is on the order of the attempt frequency ν, whereas the nucleation needs more than 4 orders of magnitude additional time to finish. If there are islands on the surface from the beginning, this effect should be clearly visible. Unfortunately, the system size would have to be much bigger than $L = 1000$ and therefore one has to settle for the results of this section.

5.2 Dependence of the Island Density on the Parameters

As one would expect from theory, the island density will depend strongly on the critical island size i as well as on the initial coverage θ.

However, since changing the diffusion barrier changes the speed at which the system evolves in real time, those processes are the only strong influence on the nucleation process. The final result would not change if one changed the value for Q. From this we conclude that knowing that parameter is not important for our case, since the final island density does not change. The same is true for the attempt frequency[6] ν.

To give an overview of the statistically derived data, fig. 30 shows the dependence of the island density N on the value of the critical island size i. The scale is logarithmic for the ordinate, because then for constant coverage, one sees that the dependence on i is linear.

Furthermore, when one keeps i constant the results are plotted in fig. 31. In this case, the linearity shows up when plotting in a double-logarithmic way.

From this information one can deduce the following law the island density probably obeys:

[6] The attempt frequency ν is more important in cases where there is a flux $F > 0$ and therefore determines the value for D/F, as shown in sec. 4.5.2

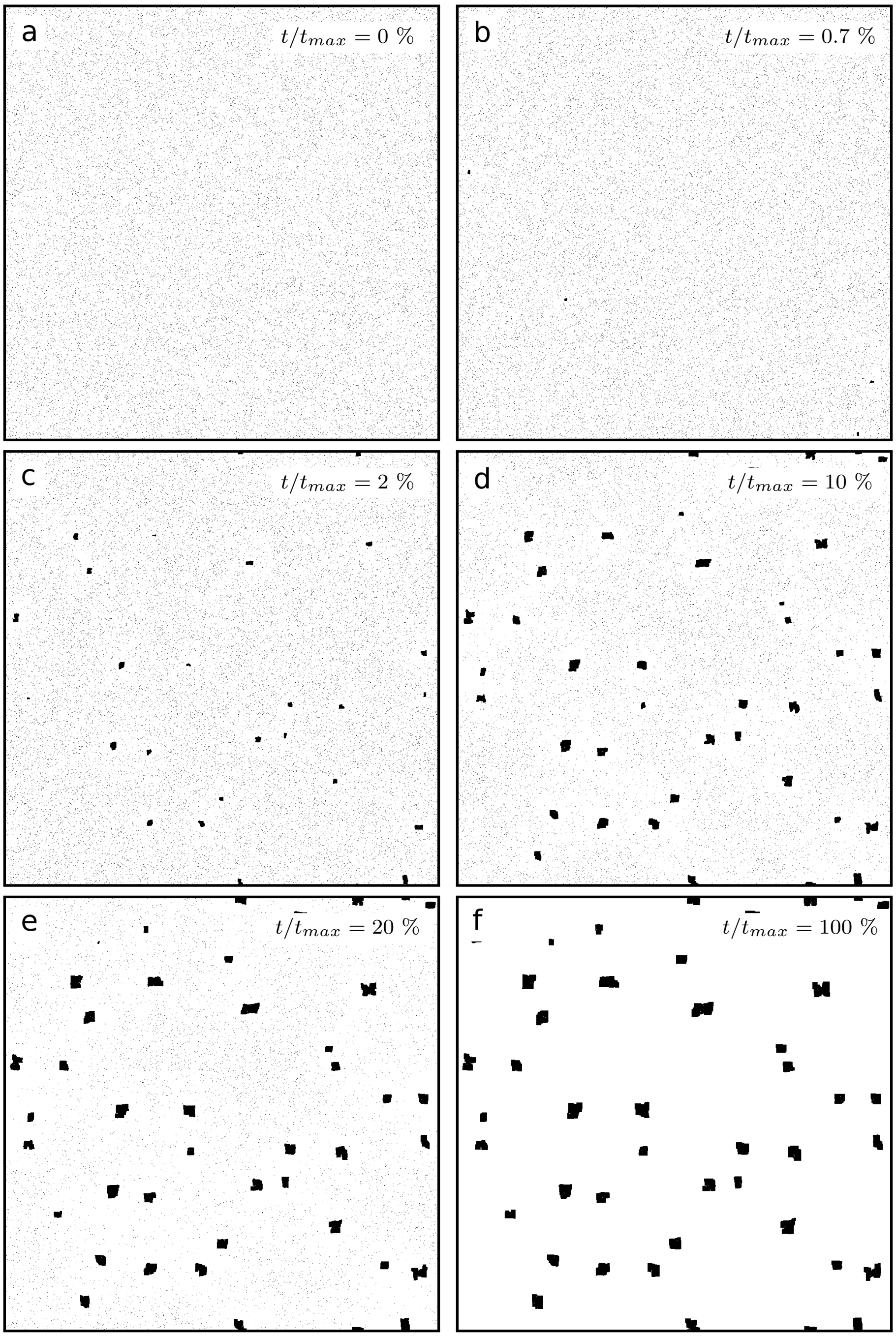


Figure 28: Time evolution of the KMC simulations of adsorbate induced subsequent nucleation. In image (b) nucleation starts and in images (c) and (d), still nucleation happens. As denuded zones form after some time, the monomer density becomes lower and therefore nucleation becomes unlikely and mainly aggregation happens as in image (e) until no monomers are left as in image (f). Parameters: monomer diffusion rate $Q = E$, lattice size $L = 1000$, coverage $\theta = 0.03$ ML, critical cluster size $i = 7$

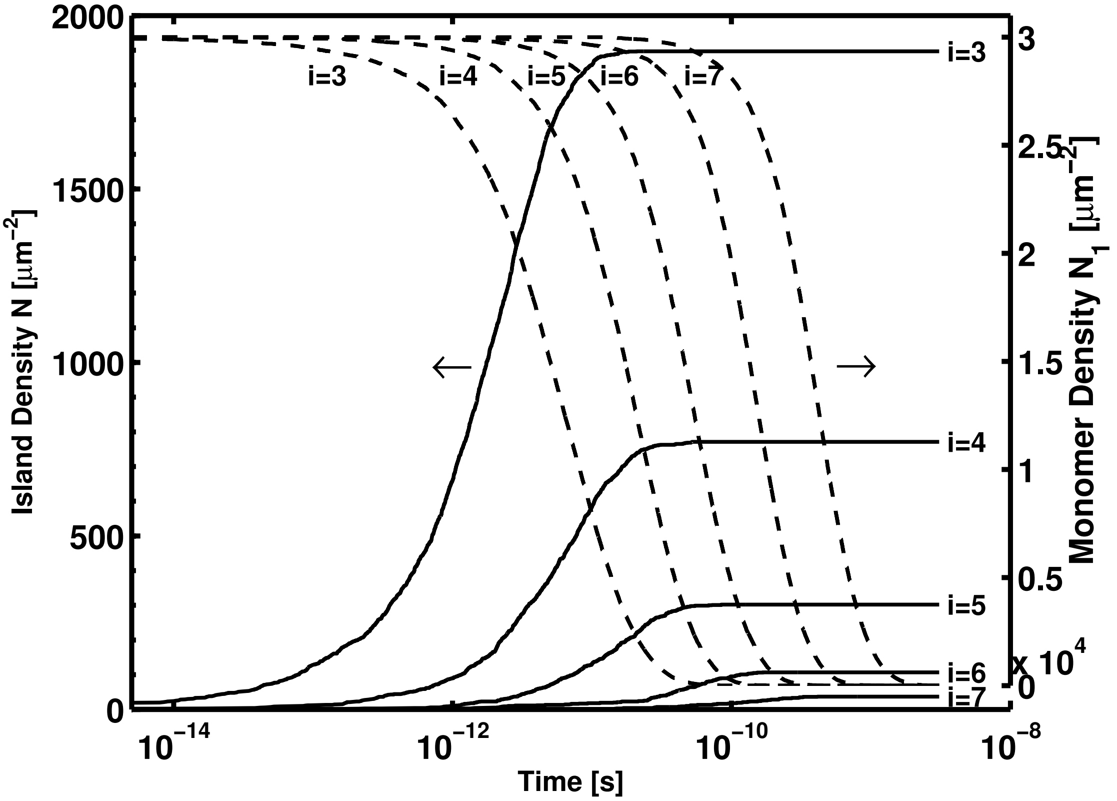


Figure 29: Monomer density (dashed lines) and island density (solid lines) for adsorbate induced subsequent nucleation as a function of time. The curves are for different critical nuclei sizes i as indicated in the image. Coverage $\theta = 0.03$ ML, attempt frequency $\nu = 10^{13}$ s^{-1}, diffusion barrier $Q = 0.05$ eV

$$N(i,\theta) = c \cdot \theta^{\alpha(i)} e^{\beta(i)} \tag{33}$$

where c is a constant, and $\alpha(i), \beta(i)$ are linear functions of i.

When one fits this function to the data provided in the plots of fig. 30 and fig. 31, the results $c = 4 \cdot 10^4$, $\alpha(i) = 0.436i$ and $\beta(i) = 0.538i$ are obtained. The corresponding fit curves are drawn in the figures as solid lines.

5.3 Comparison with the AFM Images - Island Density

The data shown in the plots of the previous sections did not include the uncertainties of the obtained values. Tab. 3 shows all the results containing this additional information.

Eq. 33 shows that the island density $N(i,\theta)$ raises exponentially with i. Although it is difficult to obtain good estimations of the coverage from the AFM pictures, this makes a quantitative comparison to the simulated data possible, because raising the

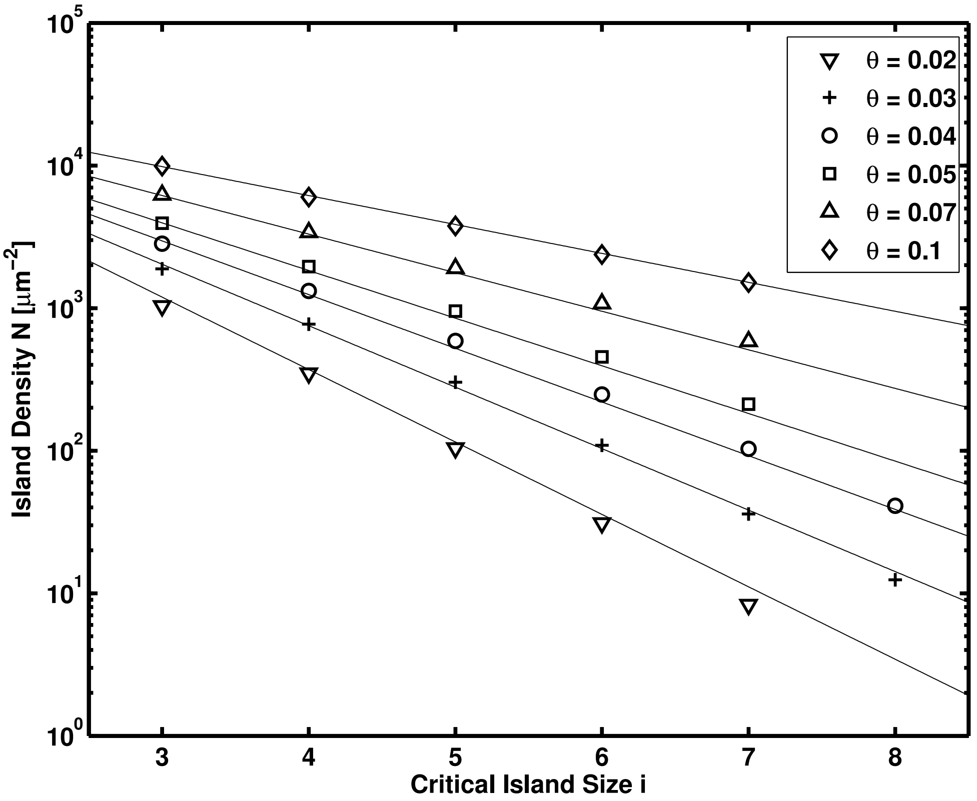


Figure 30: KMC simulations of adsorbate induced subsequent nucleation: island density N as a function of critical island size i for different values of the coverage θ that are indicated in the figure. The solid lines are the result of the fit using eq. (33). No edge diffusion was allowed.

critical nucleus size by one changes the island density by orders of magnitude. Fig. 32 shows a comparison for $i = 6$ and $i = 7$ to the AFM picture that was made for the case of $\theta = 0.03$ ML and covers an area of 1 x 1 μm^2.

One can clearly see that the number of islands approximately fits for $i = 7$ (highlighted in tab. 3), since from the AFM images a density of about $N_{AFM} = 35 \pm 5$ μm^{-2} is obtained. This value seems a little bit too high for $i = 6$ and too low for $i = 8$, but nevertheless from the data we estimate the following value for the critical island size:

$$i = 7 \pm 1$$

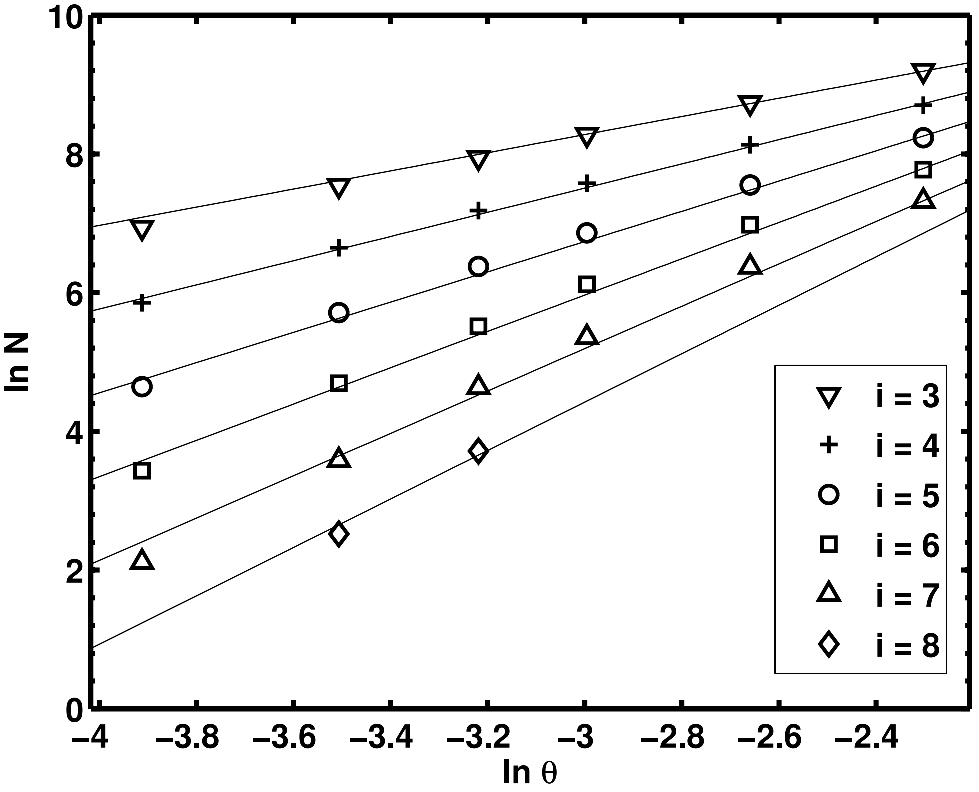


Figure 31: KMC simulations of adsorbate induced subsequent nucleation: Double logarithmic plot of the island density N [μm^{-2}] as a function of the coverage θ [ML] for different values of the critical island size i that are indicated in the figure. The solid lines are the result of the fit using eq. (33). No edge diffusion was allowed.

5.4 Comparison with the AFM Images - Island Size Distributions

Another comparison that can be done is using island size distributions (ISD) (see sec. 3.1.5). For two different values of the critical island size $i = 5$ and $i = 7$ such distributions have been made and afterwards a fit using eq. (19) was made to determine the parameter i for these cases.

However, it turns out that the process of adsorbate induced subsequent nucleation does not produce island size distributions that can be compared to the ones obtained when there is a non-zero flux $F > 0$, because even for $i = 7$, the ISD does not look in any way similar to the curve that one gets for eq. (19) with $i = 7$. Fig. 33(a) and 33(b) show these plots together with the fit curves. Because one cannot assume that the critical cluster size is reflected in the formula for the fit anymore by the variable i,

Table 3: Resulting island densities from KMC simulations of adsorbate induced subsequent nucleation. N depends strongly both on the coverage θ and the critical cluster size i. No edge diffusion was allowed.

N [μm^{-2}]		Critical Cluster Size i					
		3	4	5	6	7	8
	0.02	1030±30	349±5	104±5	31±3	8±2	
	0.03	1890±40	772±8	302±7	110±10	36±4	12±2
Coverage	0.04	2830±50	1320±10	590±10	250±20	100±10	41±4
θ [ML]	0.05	3940±20	1960±30	950±20	460±10	210±10	
	0.07	6230±40	3410±20	1910±30	1080±10	590±10	
	0.10	9910±40	6010±40	3770±30	2380±30	1510±20	

it was renamed to p:

$$f_p(x := s/S) = C_p x^p \exp(-p a_p x^{1/a_p}) \tag{34}$$

The data for the constants a_p and C_p can still be found in tab. 1 when one substitutes i by p.

When the results of the simulations are compared to the ones that are obtained by analyzing the AFM pictures, one gets the same picture of the distributions. For the small islands as depicted in fig. 34(b) the curve for a value $p \approx 1$ fits best.

Moreover, another distribution was made for the big islands (fig. 34(a)) and shows a distribution that fits best for $p = 2$.

If one however distrust the simulations, another comparison to literature has been done for the usual case of non-zeros flux $F > 0$. Fig. 35 shows the attempt to reproduce the results of Li and Evans [40] for $i = 3$, a ratio between diffusion and deposition of $h_1/F = 10^6$, and a final coverage of $\theta_{max} = 0.1$ ML. The agreement is striking for the value of $p = 3$. Consequently, the theory of Amar and Family [45] for island size distributions can be only applied for systems with a non-zero flux and that our case of adsorbate induced subsequent nucleation of 6P shows clearly a different behavior both in simulations and experiment.

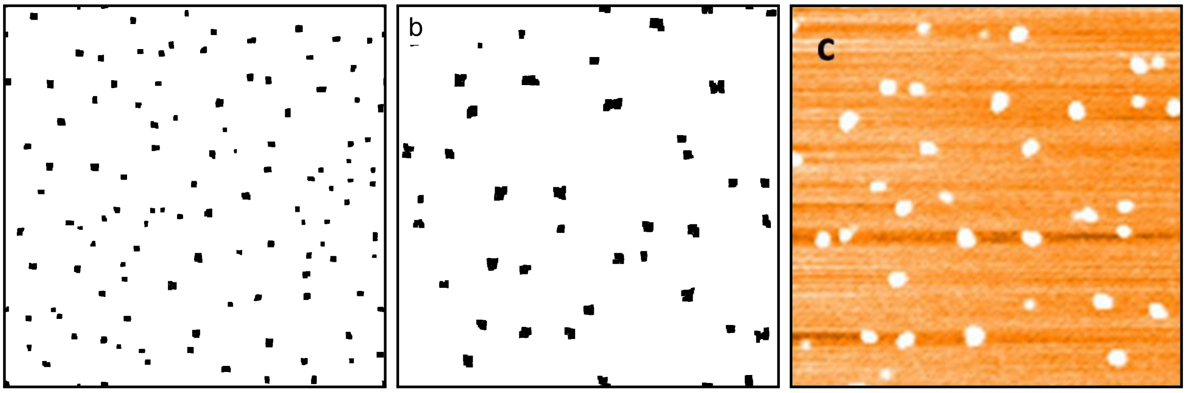


Figure 32: Comparison between AFM and KMC pictures showing islands that have their origin in adsorbate induced subsequent nucleation at a coverage $\theta = 0.03$ ML. The first two pictures are the results from KMC simulations done at (a) $i = 6$ and (b) $i = 7$. Image (c) shows the corresponding AFM picture. For the KMC simulations, no edge diffusion was allowed.

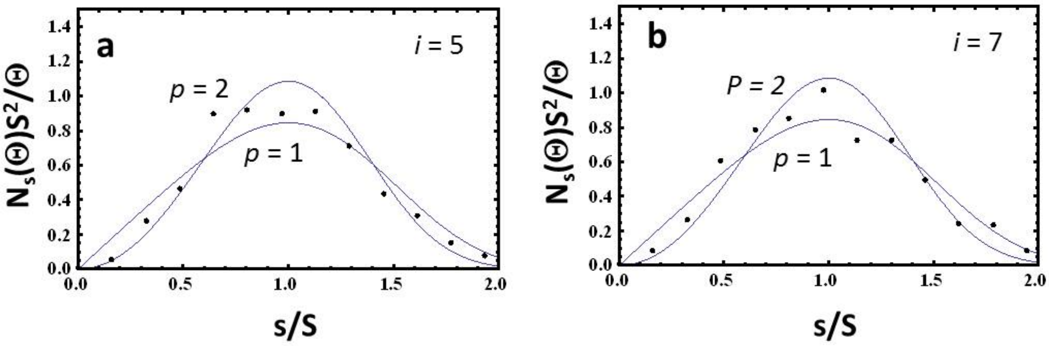


Figure 33: Island size distributions for the cases (a) $i = 5$ and (b) $i = 7$ resulting from KMC simulations of adsorbate induced subsequent nucleation with a coverage of $\theta = 0.03$ ML. The fits are done using eq. (34).

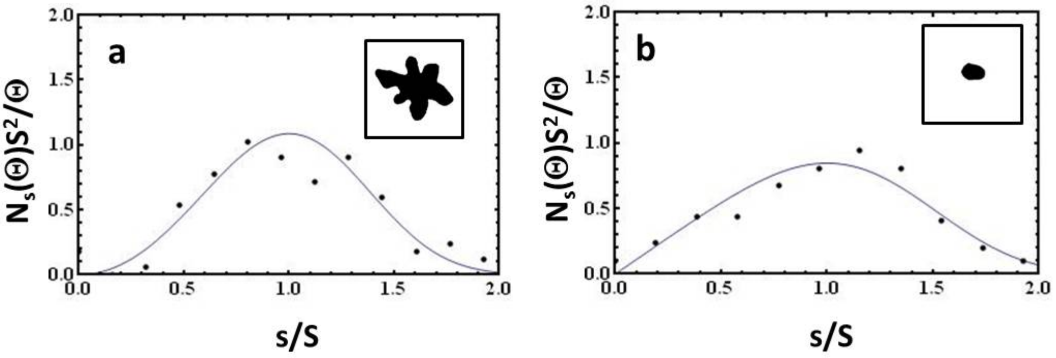


Figure 34: Island size distributions for the cases of (a) big islands at a coverage $\theta = 0.147$ ML at $T = 400K$, and (b) small islands at a coverage $\theta = 0.03$ ML at $T = 400K$, resulting from AFM measurements of 6P on sputtered mica(001). The fits are done using eq. (34). [17]

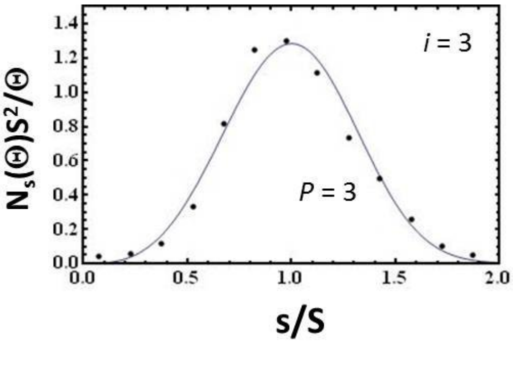


Figure 35: Island size distributions for the case $i = 3$ of KMC simulations with a non-zero flux $F > 0$. This corresponds to growth during deposition. The fits are done using eq. (34), and agree with expected results in literature [45]. Parameters: $i = 3$, $h_1/F = 10^6$, $\theta_{max} = 0.1$ ML

6 Conclusion

The peculiar behavior of para-hexaphenyl (6P) of forming bimodal island size distributions when growing ultra-thin films on sputter amorphized mica(001) was described by showing experimental results of thermal desorption spectroscopy (TDS) as well as AFM images. One could frequently observe smaller islands in the area around bigger ones, where these islands also formed a denuded zone. The proposed reason for this behavior was presented as being an effect of water adsorbed on the surface that changes the behavior of 6P on the mica surface when venting the UHV chamber. Before venting a 2D-gas phase exists on the surface, which after venting vanishes by nucleating clusters due to interaction with water.

This experimentally revealed behavior could be satisfactorily simulated by the KMC simulations done for this thesis. Though computer power on a single core is limited to smaller systems than the ones needed for simulating the whole growth process, it was possible to study the process of adsorbate induced subsequent nucleation in great detail. In order to perform the simulations an algorithm for edge hopping that respects the critical island size was developed and compared to data from existing literature and to the theory of epitaxial growth which is also given in the thesis. The procedure of simulating the adsorbate induced subsequent nucleation by KMC simulations was to start with an initial concentration of monomers on the surface and to let them start moving suddenly and hence to nucleate islands with the critical nucleus size as parameter. When all monomers were incorporated into islands, the simulation stopped and counted the island number.

Depending on the parameters of initial coverage and critical cluster size the resulting island densities and island size distributions could be compared the experimental results gathered from AFM images.

The critical island size that matches the island densities from the AFM images was found to be $i = 7$ for a coverage of $\theta = 0.03$ ML. When respecting the experimental uncertainties of AFM measurements, the value for the coverage could deviate from the real one, which leads to the assumption that also the value for i can deviate. This leads to the final result:

$$i = 7 \pm 1$$

The island size distributions showed to be clearly different from the conventional case when growth happens under the influence of a flux $F > 0$ of particles. Both in experiments and simulations the distribution showed to be broader, which is a good agreement.

Moreover the time evolution of monomer density and island density was tracked during the simulations. It turned out that for higher values of i the time for nucleation differs by several orders of magnitude from the one for monomer diffusion. This may be the reason for the denuded zone of the smaller islands distributed around the bigger ones.

Furthermore, a computational technique called Quasicontinuum Monte Carlo

(QCMC) was examined to develop a method to incorporate an effect called attachment limitation to study repulsive adsorbate interactions at the island rim. Particles approaching an island during their random walk motion may be reflected from the island rim due to a barrier. The hybrid model of KMC and the use of diffusion equations was adjusted to incorporate such a barrier.

A Update Algorithm for Edge-Hopping Events

The following pseudocode shows the steps for finding edge-hopping events in the `EventProcessor` class. The description can be found in sec. 4.3.3.

Algorithm: calcPossEvents() for partially reversible edge-hopping case ($m = 1$)

```
procedure calcPossEvents($p_{num}$)                    ▷ $p_{num}$ ... number (name) of this particle
    // check if there are neighbors
    if particle has no neighbors then
        // particle is a monomer
        $event\_type \leftarrow 0$;
        for all directions $i$ do
            if event is absent or wrong then
                // delete event if one with different type is present
                if event exists then
                    deleteEvent($p_{num}$, $i$);
                end if
                addEvent($p_{num}$, $event\_type$, $i$);
            end if
        end for
    else
        // particle is part of a cluster...
        $event\_type \leftarrow num\_neighbors$;       ▷ event type is numbered by number of neighbors
        // check for critical nucleus
        $stable \leftarrow true$;
        if $critical\_i > 1$ then
            if island size not yet known then
                calculate island size;
                execute function calcPossEvents for all those particles;
            end if
            // particle can move freely if critical nucleus has not yet been reached
            if $island\_size \leq critical\_i$ then
                $event\_type \leftarrow 0$;
                $stable \leftarrow false$;
            end if
        end if
        if $event\_type > 0$ then
            $bridge\_found \leftarrow false$;
            if $island\_size > 2$ then
                if particle forms a bridge then
                    $bridge\_found \leftarrow true$;
                end if
            else
                $bridge\_found \leftarrow true$;
            end if
            // if particle is no bridge, next check if neighboring particles all have more than one neighbor
            if $bridge\_found$ is false then
                // now follow all free directions and check for neighbor numbers
                for all directions $i$ do
                    if final position has neighbor of the same island then
                        mark direction $i$ as found possible;
                    end if
                end for
                // delete all possible existing edge hop events that have not been found here
                for all directions $j$ do
                    if direction $j$ has not been found possible then
                        deleteEvent($p_{num}$, $j$);
                    end if
                end for
            else
                // delete all possible existing events
                for all directions do
                    if event exists then
                        deleteEvent($p_{num}$, $i$);
                    end if
                end for
            end if
            unmark particle as visited;
        else
            // find free positions and add event to list
            for all directions $i$ do
                // add same event if not already present and if neighboring position is free
                if neighboring position is free and event is absent or wrong then
                    // delete event if one with different type is present
                    if event exists then
                        deleteEvent($p_{num}$, $i$)
                    end if
                    addEvent($p_{num}$, $event\_type$, $i$);
                end if
                // delete event if there is an event saved for it, but neighboring position is occupied
                if neighboring position is occupied and event exists then
                    deleteEvent($p_{num}$, $i$)
                end if
            end for
            delete all edge hopping events;
        end if
        delete all events pointing to occupied directions;
    end if
end procedure
```

References

[1] H. E. Katz and J. Huang, Thin-Film Organic Electronic Devices. *Annual Review of Materials Research* **39**, 71 (2009).

[2] S. R. Forrest and M. E. Thompson, Introduction: Organic Electronics and Optoelectronics. *Chemical Reviews* **107**, 923 (2007).

[3] T. W. Kelley, P. F. Baude, C. Gerlach, D. E. Ender, D. Muyres, M. A. Haase, D. E. Vogel, and S. D. Theiss, Recent Progress in Organic Electronics: Materials, Devices, and Processes. *Chemistry of Materials* **16**, 4413 (2004).

[4] M. Kastler, W. Pisula, F. Laquai, A. Kumar, R. J. Davies, S. Baluschev, M. C. Garcia-Gutiérrez, D. Wasserfallen, H. J. Butt, C. Riekel, G. Wegner, and K. Müllen, Organization of Charge-Carrier Pathways for Organic Electronics. *Advanced Materials* **18**, 2255 (2006).

[5] P. Puschnig and C. Ambrosch-Draxl, Density-functional study for the oligomers of poly(*para*-phenylene): Band structures and dielectric tensors. *Phys. Rev. B* **60**, 7891 (1999).

[6] G. Leising, S. Tasch, F. Meghdadi, L. Athouel, G. Froyer, and U. Scherf, Blue electroluminescence with ladder-type poly(para-phenylene) and para-hexaphenyl. *Synthetic Metals* **81**, 185 (1996).

[7] H. Yanagi and S. Okamoto, Orientation-controlled organic electroluminescence of p-sexiphenyl films. *Applied Physics Letters* **71**, 2563 (1997).

[8] G. Leising, S. Tasch, and W. Graupner. Fundamentals of Electroluminescence in Paraphenylene-type Conjugated Polymers and Oligomers. In *Handbook of Conducting Polymers, 2nd edition*, 847. T. Skotheim, R. Elsenbaumer, J. Reynolds, Marcel Dekker, New York, 1997.

[9] C. Winder, A. Andreev, H. Sitter, G. Matt, N. S. Sariciftci, and D. Meissner, Optoelectronic devices based on para-sexiphenyl films grown by Hot Wall Epitaxy. *Synthetic Metals* **139**, 573 (2003).

[10] D. J. Gundlach, Y.-Y. Lin, T. N. Jackson, and D. G. Schlom, Oligophenyl-based organic thin film transistors. *Applied Physics Letters* **71**, 3853 (1997).

[11] T. Potocar, S. Lorbek, D. Nabok, Q. Shen, L. Tumbek, G. Hlawacek, P. Puschnig, C. Ambrosch-Draxl, C. Teichert, and A. Winkler, Initial stages of a para-hexaphenyl film growth on amorphous mica. *Physical Review B* **83**, 075423 (2011).

[12] R. Ruiz, D. Choudhary, B. Nickel, T. Toccoli, K.-C. Chang, A. C. Mayer, P. Clancy, J. M. Blakely, R. L. Headrick, S. Iannotta, and G. G. Malliaras, Pentacene Thin Film Growth. *Chemistry of Materials* **16**, 4497 (2004).

[13] P. Frank, G. Hlawacek, O. Lengyel, A. Satka, C. Teichert, R. Resel, and A. Winkler, Influence of surface temperature and surface modifications on the initial layer growth of para-hexaphenyl on mica (001). *Surface Science* **601**, 2152 (2007).

[14] L. Kankate, F. Balzer, H. Niehus, and H.-G. Rubahn. *The Journal of Chemical Physics* **128**, 084709 (2008).

[15] C. Teichert, G. Hlawacek, A. Andreev, H. Sitter, P. Frank, A. Winkler, and N. Sariciftci, Spontaneous rearrangement of para-sexiphenyl crystallites into nanofibers. *Applied Physics A: Materials Science & Processing* **82**, 665–669 (2006).

[16] L. Tumbek and A. Winkler, Attachment limited versus diffusion limited nucleation of organic molecules: Hexaphenyl on sputter-modified mica. *Surface Science* (2012). DOI: 10.1016/j.susc.2012.03.018.

[17] L. Tumbek, A. Winkler, C. Gleichweit, and K. Zojer, Origin of the bimodal island size distribution in ultra-thin films of para-hexaphenyl on mica. *Phys. Rev. B* (submitted 2012).

[18] NIST Standard reference database Number 69 (Chemistry), 2012. `http://webbook.nist.gov/chemistry/ (29.05.2012).`

[19] P. Frank. Thin film growth of rod-like and disc-shaped organic molecules on insulator and noble metal surfaces. Ph.D. thesis, Graz University of Technology, 2009.

[20] K. Seki, U. O. Karlsson, R. Engelhardt, E. E. Koch, and W. Schmidt, Intramolecular band mapping of poly (p-phenylene) via UV photoelectron spectroscopy of finite polyphenyls. *Chemical Physics* **91**, 459 (1984).

[21] A. Niko, F. Meghdadi, C. Ambrosch-Draxl, P. Vogl, and G. Leising, Optical absorbance of oriented thin films. *Synthetic Metals* **76**, 177 (1996).

[22] G. Hlawacek. Molecular growth mechanisms in para-sexiphenyl thin film deposition. Ph.D. thesis, MU Leoben, 2007.

[23] D. Kandel, Initial Stages of Thin Film Growth in the Presence of Island-Edge Barriers. *Phys. Rev. Lett.* **78**, 499 (1997).

[24] J. A. Venables and H. Brune, Capture numbers in the presence of repulsive adsorbate interactions. *Phys. Rev. B* **66**, 195404 (2002).

[25] D. Choudhary, P. Clancy, R. Shetty, and F. Escobedo, A Computational Study of the Sub-monolayer Growth of Pentacene. *Advanced Functional Materials* **16**, 1768 (2006).

[26] G. S. Bales and D. C. Chrzan, Dynamics of irreversible island growth during submonolayer epitaxy. *Physical Review B* **50**, 6057 (1994).

[27] M. Li and J. W. Evans, Geometry-based simulation (GBS) algorithms for island nucleation and growth during sub-monolayer deposition. *Surface Science* **546**, 127 (2003).

[28] M. A. Herman and H. Sitter. *Molecular Beam Epitaxy: Fundamentals and Current Status.* Springer-Verlag Berlin, 1989.

[29] J. W. Evans, P. A. Thiel, and M. C. Bartelt, Morphological evolution during epitaxial thin film growth: Formation of 2D islands and 3D mounds. *Surface Science Reports* **61**, 1 (2006).

[30] G. Ehrlich and F. G. Hudda, Atomic View of Surface Self-Diffusion: Tungsten on Tungsten. *The Journal of Chemical Physics* **44**, 1039 (1966).

[31] R. L. Schwoebel and E. J. Shipsey, Step Motion on Crystal Surfaces. *Journal of Applied Physics* **37**, 3682 (1966).

[32] A. F. Voter, Introduction to the Kinetic Monte Carlo Method. *Radiation Effects in Solids* **235**, 1 (2007).

[33] G. H. Vineyard, Frequency factors and isotope effects in solid state rate processes. *Journal of Physics and Chemistry of Solids* **3**, 121 (1957).

[34] C. Ratsch and J. A. Venables, Nucleation theory and the early stages of thin film growth. *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films* **21**, S96 (2003).

[35] J. A. Venables, G. D. T. Spiller, and M. Hanbücken, Nucleation and growth of thin films. *Reports on Progress in Physics* **47**, 399 (1984).

[36] D. Walton, Nucleation of Vapor Deposits. *Journal of Chemical Physics* **37**, 2182 (1962).

[37] G. Zinsmeister, Theory of thin film condensation Part B: Solution of the simplified condensation equation. *Thin Solid Films* **2**, 497 (1968).

[38] G. Zinsmeister, Theory of thin film condensation Part D: Influence of a variable collision factor. *Thin Solid Films* **7**, 51 (1971).

[39] J. A. Venables, Rate equation approaches to thin film nucleation kinetics. *Philosophical Magazine* **27**, 697 (1973).

[40] M. Li and J. W. Evans, Modeling of Island Formation During Submonolayer Deposition: A Stochastic Geometry-Based Simulation Approach. *Multiscale Modeling & Simulation* **3**, 629 (2005).

[41] L. H. Tang, Island formation in submonolayer epitaxy. *Journal de Physique I* **3**, 935 (1993).

[42] J. G. Amar, F. Family, and P.-M. Lam, Dynamic scaling of the island-size distribution and percolation in a model of submonolayer molecular-beam epitaxy. *Phys. Rev. B* **50**, 8781 (1994).

[43] M. J. Stowell, The dependence of saturation nucleus density on deposition rate and substrate temperature in the case of complete condensation. *Philosophical Magazine* **21**, 125 (1970).

[44] T. Vicsek and F. Family, Dynamic Scaling for Aggregation of Clusters. *Physical Review Letters* **52**, 1669 (1984).

[45] J. G. Amar and F. Family, Critical Cluster Size: Island Morphology and Size Distribution in Submonolayer Epitaxial Growth. *Physical Review Letters* **74**, 2066 (1995).

[46] J. G. Amar, M. N. Popescu, and F. Family, Rate-Equation Approach to Island Capture Zones and Size Distributions in Epitaxial Growth. *Physical Review Letters* **86**, 3092 (2001).

[47] P. Clancy, Application of Molecular Simulation Techniques to the Study of Factors Affecting the Thin-Film Morphology of Small-Molecule Organic Semiconductors. *Chemistry of Materials* **23**, 522 (2011).

[48] W. Khon and L. J. Sham, Self-Consistent Equations Including Exchange and Correlation Effects. *Phys. Rev* **140**, A1133 (1965).

[49] L. Verlet, Computer ”Experiments” on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. *Physical Review* **159**, 98 (1967).

[50] D. Beeman, Some multistep methods for use in molecular dynamics calculations. *Journal of Computational Physics* **20**, 130 (1976).

[51] D. Frenkel and B. Smit. *Understanding Molecular Simulation: From Algorithms to Applications.* Academic Press, Inc., 1996.

[52] D. C. Rapaport. *The Art of Molecular Dynamics Simulation.* Cambridge Univ. Press, 2004.

[53] N. Metropolis, The Beginning of the Monte Carlo Method. *Los Alamos Science* **15**, 125 (1987).

[54] N. Metropolis and S. Ulam, The Monte Carlo Method. *Journal of the American Statistical Association* **44**, 335 (1949).

[55] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics* **21**, 1087 (1953).

[56] K. Binder, Applications of Monte Carlo methods to statistical physics. *Reports on Progress in Physics* **60**, 487 (1997).

[57] D. P. Kroese, T. Taimre, and Z. I. Botev. *Handbook of Monte Carlo Methods*, volume 706. Wiley, 2011.

[58] W. K. Hastings, Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* **57**, 97 (1970).

[59] A. Chatterjee and D. G. Vlachos, An overview of spatial microscopic and accelerated kinetic Monte Carlo methods. *Journal of Computer-Aided Materials Design* **14**, 253 (2007).

[60] K. A. Fichthorn and W. H. Weinberg, Theoretical foundations of dynamical Monte Carlo simulations. *Journal of Chemical Physics* **95**, 1090 (1991).

[61] D. T. Gillespie, Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry* **81**, 2340 (1977).

[62] C. Groves, R. G. E. Kimber, and A. B. Walker, Simulation of loss mechanisms in organic solar cells: A description of the mesoscopic Monte Carlo technique and an evaluation of the first reaction method. *Journal of Chemical Physics* **133**, 144110 (2010).

[63] P. Zoontjens, T. P. Schulze, and S. C. Hendy, Hybrid method for modeling epitaxial growth: Kinetic Monte Carlo plus molecular dynamics. *Physical Review B* **76**, 245418 (2007).

[64] L. Mandreoli, J. Neugebauer, R. Kunert, and E. Schöll, Adatom density kinetic Monte Carlo: a hybrid approach to perform epitaxial growth simulations. *Physical Review B* **68**, 155429 (2003).

[65] G. Russo, L. M. Sander, and P. Smereka, Quasicontinuum Monte Carlo: A method for surface growth simulations. *Physical Review B* **69**, 121406 (2004).

[66] J. P. DeVita, L. M. Sander, and P. Smereka, Multiscale kinetic Monte Carlo algorithm for simulating epitaxial growth. *Physical Review B* **72**, 205421 (2005).

[67] M. A. Katsoulakis, A. J. Majda, and D. G. Vlachos, Coarse-grained stochastic processes and Monte Carlo simulations in lattice systems. *Journal of Computational Physics* **186**, 250 (2003).

[68] S. Plimpton, C. Battaile, M. Chandross, L. Holm, A. Thompson, V. Tikare, G. Wagner, X. Webb, C. G. Cardona, and A. Slepoy, Crossing the Mesoscale No-Man's Land via Parallel Kinetic Monte Carlo. *Sandia report* (2009).

[69] C. C. Battaile, The kinetic Monte Carlo method: Foundation, implementation, and application. *Computer Methods in Applied Mechanics and Engineering* **197**, 3386 (2008).

[70] A. B. Bortz, M. H. Kalos, and J. L. Lebowitz, A new algorithm for Monte Carlo simulation of Ising spin systems. *Journal of Computational Physics* **17**, 10 (1975).

[71] D. R. Cox and H. D. Miller. *The theory of stochastic processes, Methuen & Co.* Wiley, New York, 1965.

[72] W. M. Young and E. W. Elcock, Monte Carlo studies of vacancy migration in binary ordered alloys: I. *Proceedings of the Physical Society* **89**, 735 (1966).

[73] T. P. Schulze, Kinetic Monte Carlo simulations with minimal searching. *Physical Review E* **65**, 036704 (2002).

[74] C. C. Chou and M. L. Falk, Multiscale diffusion Monte Carlo simulation of epitaxial growth. *Journal of Computational Physics* **217**, 519 (2006).

[75] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Second Edition*, chapter 3, p. 44. MIT Press, 2001.

[76] A. Chatterjee and D. G. Vlachos, Multiscale spatial Monte Carlo simulations: Multigriding, computational singular perturbation, and hierarchical stochastic closures. *Journal of Chemical Physics* **124**, 064110 (2006).

[77] A. Samant and D. G. Vlachos, Overcoming stiffness in stochastic simulation stemming from partial equilibrium: A multiscale Monte Carlo algorithm. *Journal of Chemical Physics* **123**, 144114 (2005).

[78] W. K. Burton, N. Cabrera, and F. C. Frank, The Growth of Crystals and the Equilibrium Structure of their Surfaces. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences* **243**, 299 (1951).

[79] C. Ratsch, M. F. Gyure, R. E. Caflisch, F. Gibou, M. Petersen, M. Kang, J. Garcia, and D. D. Vvedensky, Level-set method for island dynamics in epitaxial growth. *Physical Review B* **65**, 195403 (2002).

[80] A. Chatterjee, D. G. Vlachos, and M. A. Katsoulakis, Spatially adaptive lattice coarse-grained Monte Carlo simulations for diffusion of interacting molecules. *Journal of Chemical Physics* **121**, 11420 (2004).

[81] M. A. Katsoulakis and D. G. Vlachos, Coarse-grained stochastic processes and kinetic Monte Carlo simulators for the diffusion of interacting particles. *Journal of Chemical Physics* **119**, 9412 (2003).

[82] A. Chatterjee, D. G. Vlachos, and M. A. Katsoulakis, Binomial distribution based τ-leap accelerated stochastic simulation. *Journal of Chemical Physics* **122**, 024112 (2005).

[83] A. Chatterjee and D. G. Vlachos, Temporal acceleration of spatially distributed kinetic Monte Carlo simulations. *Journal of Computational Physics* **211**, 596 (2006).

[84] F. El-Mellouhi, N. Mousseau, and L. J. Lewis, Kinetic activation-relaxation technique: An off-lattice self-learning kinetic Monte Carlo algorithm. *Phys. Rev. B* **78**, 153202 (2008).

[85] J. DeVita, L. Sander, and P. Smereka. Quasicontinuum Monte Carlo Simulation of Multilayer Surface Growth. In *Multiscale Modeling in Epitaxial Growth*, volume 149 of *International Series of Numerical Mathematics*, 57. Birkhäuser Basel, 2005.

[86] R. J. Leveque. *Finite Volume Methods for Hyperbolic Problems.* Cambridge texts in applied mathematics. Cambridge Univ. Press, 1 edition, 2002.

[87] S. Ramnath and B. Dathan. *Object-Oriented Analysis and Design.* Undergraduate Topics in Computer Science. Springer, 2011.

[88] TU Graz Lecture: Software Development Practical Exercises (706.007) - Coding Standard, 2011. `https://palme.iicm.tugraz.at/wiki/SEP/CS (30.04.2012).`

[89] G. Hlawacek, P. Puschnig, P. Frank, A. Winkler, C. Ambrosch-Draxl, and C. Teichert, Characterization of Step-Edge Barriers in Organic Thin-Film Growth. *Science* **321**, 108 (2008).

[90] P. A. Mulheran and J. A. Blackman, Capture zones and scaling in homogeneous thin-film growth. *Phys. Rev. B* **53**, 10261 (1996).