KMCInterative: an interactive molecular beam epitaxy kinetic Monte Carlo simulator for education

Michael Grundmann

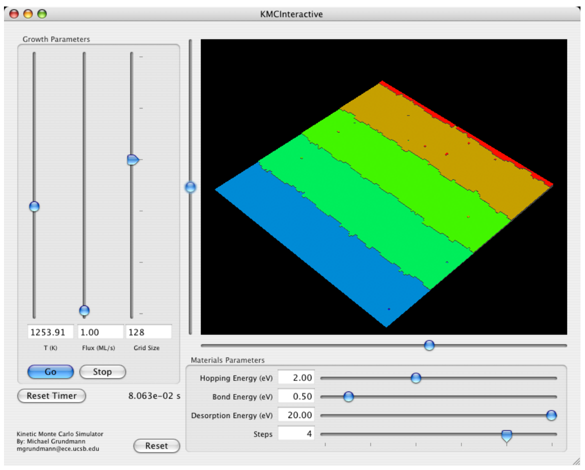
November 20, 2006

Abstract

A fast Monte Carlo simulator with a graphical user interface is an ideal education tool for students learning about molecular beam epitaxy (MBE). This paper discusses the implementation of such a simulator, as well as simulation techniques that may be used to tailor surfaces. Simulations carried out on the learning tool show how MBE can recover step-flow growth, in addition to the temperature dependence of the surface morphology.

1 Introduction

KMCInteractive (figure 1) is a solid–on–solid kinetic Monte Carlo (KMC) simulation program that is designed to be fast and have an easy to use user interface for education purposes. Monte Carlo simulations have long been performed to explain experimental results[2, 7, 6], but recent personal computer hardware is now sufficient to solve reasonably complex problems in a short amount of time. In 1987, Vvedensky *et al.* used a Cray X-MP/48 supercomputer to perform KMC calculations for MBE on a 1,000 site grid[2]. These computers cost roughly $16 million dollars and performed roughly 220 MFLOPS (million floating point operations per second)[1]. By



T (K): 1253.91
Flux (ML/s): 1.00
Grid Size: 128
8.063e-02 s
Hopping Energy (eV): 2.00
Bond Energy (eV): 0.50
Desorption Energy (eV): 20.00
Steps: 4

Kinetic Monte Carlo Simulator
By: Michael Grundmann
mgrundmann@ece.ucsb.edu

Figure 1: Screenshot of KMCInteractive

comparison, an Apple Power Mac G5 currently costs less than \$2,000 and can perform up to 8 GFLOPS per processor on vector optimized code[4]. x86 processors have similar performance. Though FLOPS may not be an ideal indicator of performance, these number nonetheless show that modern desktop computers now have the computing power that was previously reserved for supercomputers. Therefore, many simulations that previously needed supercomputers may now be done faster on a desktop computer.

In addition to the increases in modern processor performance, display technology has been exceeding Moore's law for several years and has made the display of complex scientific data relatively simple, with the additional benefit of offloading the main computer processor for simulation tasks. KMCInteractive takes advantage of these improvements by displaying the simulation as it executes in a 3-dimentional plot.

2 Implementation

Monte Carlo simulation is a very compute-time intensive task. In order to create a simulation that can be seen in a reasonable amount of time and view the results in a responsive graphical user interface, a quad tree data structure stores the rate information, and the simulation takes place in a separate thread from the user interface. A lockable object is used to pass information between the view and simulation threads.

2.1 Quad tree

Each simulation iteration, an event is selected from an event stack. In general, a random number R is chosen such that $0 \leq R \leq \sum r_i$ where r_i is the rate for event i. After choosing the number, the program finds the event j such that $\sum_0^j r_i \leq R < \sum_0^{j+1} r_i$. The naive approach is to construct a list containing all of the events and iterate through the list until the above conditions are satisfied. However, if the list has a length n this takes $O(n)$ operations, and performing local updates to the rate equations also takes $O(n)$ operations. A common approach in Monte Carlo algorithms is to use a binary tree[5] to store the rate information. Finding the appropriate event j takes $O(\log n)$ operations using a binary tree. This technique speeds up searches considerably. For the sake of clarity, operations are defined loosely here as the number of sites each algorithm must visit; it does not take into account additional overhead. For instance, to find an event in a 128 x 128 grid would take roughly 4000 operations on average using the list approach, while it would take about

14 operations using a binary tree. A binary tree has the additional benefit that its operation count to find a node is always the same, while the list in the above example has a worst case search time of over 8000 operations (though a best case search time of 1 operation.) Despite the search time advantage, a binary tree does not provide a convenient framework in which to update local rates after each iteration, which could still require $O(n)$ operations.

KMCInteractive uses an extension of the binary tree algorithm mentioned above. Instead of each node in the tree having two children, each node in KMCInteractive has four children, thus it uses a quad tree[5]. In this scheme, the locality of neighboring sites is preserved in the data structure, so updates are straightforward. The quad tree search algorithm also runs in $O(\log n)$ time, so there is no penalty over the binary tree in using it.

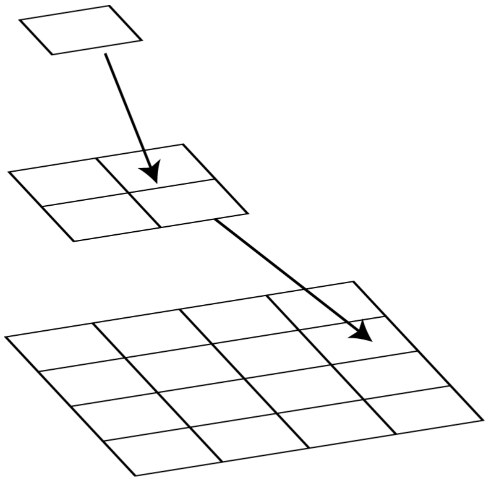


Figure 2: Traversing a quad tree. The top node is the root of the tree, and the bottom 4x4 grid contains the leaves of the tree in a 2^n x 2^n $n = 2$ tree.

To produce an efficient quad tree, the simulation grid is a square with edges 2^n in length, so that the depth of the tree is simply n. Each node in the tree has a value associated with it that is a rate. The top level node has the value r_{tot}, which is the sum of the rates in all of the leaf nodes in the tree. Each of its children nodes have values that are the sum of their particular quarter of the overall space. Each subdivision breaks the total sum r_{tot} into four sub-rates k_{ji} where j is the depth from the root in the quad tree and $i = 1..4$ is the index of the four children that are the sum of the rates in the branch. For example, $\sum k_{2i} = r_{\text{tot}}$, the sum of all the rates in the system, and $\sum k_{3i} = k_{2m}$ where m denotes one of the four children of the root node.

The algorithm to find the event associated with rate r_i is as follows. A random number R_0 is chosen using the above constraints. R_j, the modified random number at level j is then compared until $\sum_{i=1}^{n} k_{ji} > R_j$ to find the child node with index n that contains the specified branch of interest. The program then visits the child node with a modified random number $R_{j+1} = R_j - \sum_{i=1}^{n} k_{ji}$ until it reaches a leaf node. The leaf node is the event of interest and is executed.

After each event step, it is necessary to update the rate tree with new rates since the system has changed state. Instead of updating every leaf node in the tree, it is possible with quad tree to just update the nearest neighbors of the node of interest, taking $O(\log n)$ operations instead of $O(n)$ operations to update the entire tree. On a 128 x 128 grid, this decreases the computation time by roughly three orders of magnitude.

2.2 Thread Structure

A compute intensive task like Monte Carlo simulation generally makes interactive user interfaces difficult to create in a single-threaded environment. In order to circumvent this problem, many such programs are written with a multi-threaded design. KMCInteractive uses two threads: one for the user interface and display of data, and another for Monte Carlo calculations. Since both threads access the same data, the program uses standard mutual exclusion devices ensure only one thread is reading from or writing to the shared data[8].

2.3 Application Programming Interfaces (APIs)

The user interface for KMCInteractive is written in objective-C++ using the Cocoa API for Apple Macintosh OS X applications. The 3–dimensional display uses standard OpenGL API calls for portability. The Monte Carlo backend is written in standard C for maximum inter-platform portability, with an interface that is objective-C++ to handle the mutual exclusion operations on the data.

2.4 Rates

KMCInteractive has six types of events: hopping in four directions, desorption and adsorption. Each type of event at each lattice site is associated with a rate.

There are four types of hopping (diffusion) events: up, down, left and right. Each event has the rate:

$$r_{\text{hop}} = \nu \exp\left(-\frac{E_{\text{hop}} + n\Delta E}{k_b T}\right)$$

Where $\nu = (k_b T)/h$ is the vibrational frequency, typically of the order $10^{13} s^{-1}$, E_{hop} is a characteristic energy associated with diffusion, n is the decrease in number of nearest neighbors caused by a hop, and ΔE is the binding energy of two atoms.

The total adsorption rate is set by the incoming flux Φ in ML/s.

$$R_{\text{ads}} = A\Phi$$

where A is number of sites (equivalent to the area of the grid.) This simply gives the adsorption rate per site as $r_{\text{ads}} = \Phi$.

Desorption rates are calculated in similar fashion to hopping rates:

$$r_{\text{des}} = \nu \exp\left(-\frac{nE_{\text{des}}}{k_b T}\right)$$

where E_{des} is the energy required to break a bond to a nearest neighbor, and n is the number of nearest neighbors, including the underlying atom. In most simulations, E_{des} can be set to a large value to avoid adatom desorption, unless this is the focus of the study.

In practice, all rates are pre-computed and stored in arrays for quick access. Floating point operations can be very costly in terms of computation time, and since the rates do not change, pre-computation improves the execution time. When the user makes a change in the user interface, the rates are updated, and simulation continues.

2.5 Consequences of Rate Calculation Simplification

The simplifications that hopping rates in KMCInteractive only include nearest neighbors and atoms may only hop to nearest neighbor sites introduces artifacts in the simulation that may be alleviated by including second nearest neighbors in calculations. An artifact that is visible is that columns of atoms may form on the surface. When an atom binds to the end of a column, it has reached its local minimum in energy, since there is no direct hopping path to the side of a column and even if an atom could hop to the side of a column, the energy associated with that site is the same as at the end. In order to bind to the side of a column, the atom must first break its bond to the column to hop away, then hop to the edge, requiring a low-rate event in the nearest neighbor hopping framework. When including second nearest neighbors, the site on the side of a column adjacent to the end has a lower energy due to the additional bond to the second nearest neighbor. Also, the adatom may hop directly to the energetically favorable binding spot directly.

This artifacts is not removed in KMCInteractive to keep the simulation rate high. Many research Monte Carlo simulations use a similar rate calculation scheme with excellent agreement with experimental results, despite the problems mentioned here[2, 3].

2.6 Boundary Conditions

KMCInteractive uses periodic boundary conditions to preserve particle number. For stepped structures, one edge of the grid is assumed to be n particles higher than the other, where n is the number of steps. These boundary conditions can clearly be observed by noting that islands that form at the ends of the grid wrap around to the corresponding grid sites. Periodic boundary conditions may have an important effect on some growth conditions, since the grid size sets the maximum spacing between nucleating islands, and thus the maximum island size.

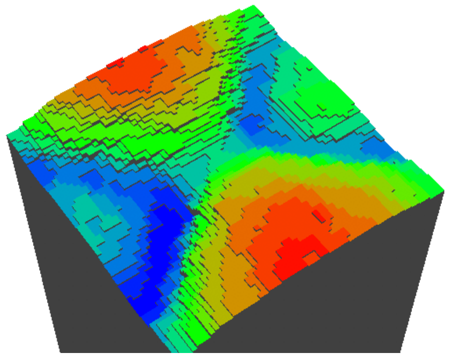


Figure 3: An island pinned in size by periodic boundary conditions. Note the island is split across a boundary.

3 Simulation Techniques

There are several techniques in KMCInteractive that can be used to simulate different growth conditions.

3.1 Rough Surfaces

KMCInteractive produces a smooth surface when the simulation starts. A rough surface can be obtained two different ways. First, the surface can be statistically roughened by reducing the surface mobility of the adatoms. This is easily accomplished by lowering the growth temperature to 0K. In this case, adatoms are randomly delivered to the system and stick where they land.

The other roughening technique is thermal roughening. To obtain a completely roughened surface, the temperature must be increased and the bond energy decreased. To find where a completely rough surface is obtained, one can plot the Gibb's free energy versus surface coverage function for the surface $f_g/w = 4\theta(1-\theta) + k_bT/w(\theta\ln(\theta) + (1-\theta)\ln(1-\theta))$ where θ is the surface coverage in monolayers and w is the bond strength and find the temperature and bond energy combination that provides a curve that satisfies $\frac{\partial f}{\partial T} > 0$ at all surface coverages. This yields a roughening temperature of $T_{\text{rough}} = 2w/k_b$[9]. This model does not take into account multi-layer roughening, which occurs in these simulations, but this value for T_{rough} is a good estimate of the roughening temperature, as is shown in section 4.2.

In practice, one may simply reduce the bond energy to zero. At this limit, the only component in the Gibb's free energy is entropy, so the surface will randomize.

3.2 Step-flow growth

Step-flow growth can be simulated on both rough and smooth miscut surfaces. To simulate growth starting on a smooth miscut surface, the number of steps may be selected in the Materials Parameters box, and the simulation should be reset. To simulate growth on a rough surface, the number of steps may be chosen, and the surface roughened using one of the techniques listed in section 3.1.

The proper growth conditions must be met to achieve step-flow growth. For example, usingthe default material parameters of $E_{\mathrm{des}} = 20\mathrm{eV}$, $E_{\mathrm{hop}} = 2.0\mathrm{eV}$, $\Delta E = .5\mathrm{eV}$, 3 steps, and $\Phi = 1\mathrm{ML/s}$, the growth temperature for step-flow is between 1250 K and 1300 K. See section 4.1 for more details.

3.3 Islanding

In an intermediate regime between statistical roughening and layer-by-layer or step-flow growth, the surface grows by nucleating islands. In this growth regime, the adatoms have a high enough surface mobility to migrate, but stick readily to other atoms migrating on the surface, thus nucleating islands.

3.4 Phase separated surfaces

To view a phase separated surface, stop the simulation using the 'stop' button. See section 4.2 for more details.

3.5 Annealing

Growth surfaces may be annealed by reducing the flux to 0 ML/s. It is important to keep the temperature above roughly 20K to avoid singularities in the event stack and reduce underflow error in the rate calculations while the flux is 0 ML/s. See section 4.2 for more details.

4 Simulations

4.1 Morphology Recovery

MBE has been shown to be effective at smoothing rough surfaces. To show this, one can create a rough surface in KMCInteractive using statistical roughening by lowering the temperature to 0K and letting about 40 ML of atoms deposit on the surface. After this, the simulation is stopped, and the temperature increased to 1100 K, 1200 K or 1300 K. The materials parameters used here are $E_{\text{des}} = 20\text{eV}$, $E_{\text{hop}} = 2.0\text{eV}$, $\Delta E = .5\text{eV}$, 3 steps, $\Phi = 1\text{ML/s}$, and a grid size of 64. Figure 4 shows a typical roughened surface before recovery takes place. Figure 5 shows the surface at various after various amounts of material deposition. At 1100 K, the surface has become ordered at 0.25 ML, but there is little step structure, while at 1300 K the step structure is already apparent at $\theta = .25\text{ML}$. At .5 ML surface coverage, the simulation at 1300 K has completely recovered the step structure, while at 1100 K the surface is still characterized by islands that nucleate on the terraces. Beyond .5ML the simulation at 1300 K grows by step-flow growth, and at 1200 K the simulation grows via step-flow after roughly 1ML. The 1100 K simulation starts to loosely resemble step-flow growth at 10ML; however, islands form and then are swept up and incorporated into the steps as they coalesce.

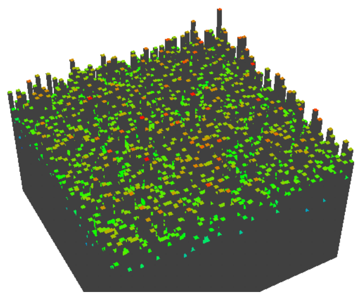


Figure 4: Statistically roughened surface

Future versions of KMCInteractive could calculate the step edge density in order to view the RHEED oscillations that occur during growth on terraced surfaces[2, 7]. If this were the case, oscillations would be quenched in the 1300K sample, since step-flow growth occurs almost immediately, and the step density (correlated to the number of dangling bonds in the plane of the substrate[7]) does not change drastically during growth.

4.2 Annealing

A study of surface morphology as a function of temperature may be carried out by depositing the desired coverage in KMCInteractive and then changing the flux to zero. Figure 6 shows the results of this method for a range of temperatures. The materials parameters used here are $E_{\text{des}} = 20\text{eV}$, $E_{\text{hop}} = 2.0\text{eV}$, $\Delta E = 0.2\text{eV}$, 3 steps, $\Phi = 0\text{ML/s}$, and a grid size of 64. One-half of a monolayer was deposited before carrying out the annealing simulations.

A simple model discussed by Tsao helps to quantitatively describe the system, though the model accounts for single layer roughening only. This is not the case in this KMC simulation, as can be seen in the snapshot at 2500 K in figure 6. The same formulation used to predict the roughening temperature in section 3.1 is used here to explain the differences in surface morphology as a function of temperature. The free energy of a single layer rough system is predicted to be

$f_g/w = 4\theta(1-\theta) + k_bT/w(\theta\ln(\theta) + (1-\theta)\ln(1-\theta))$ where θ is the surface coverage in monolayers and w is the bond strength. This is plotted in figure 6 for the different temperatures simulated. At 250 K, the local minima near $\theta = 1$ and $\theta = 0$ are not visible, but the snapshot shows that there are local minima, and the surface separates into two phases along the common tangent between these minima - occupied, and vacant, to satisfy $\theta = 0.5$. The surface at 500 K shows similar properties to 250 K, but some deviation from theory that may be attributed to additional layers being available to participate in roughening can be seen. At 1000 K, the surface is still separated into two phases, but the two phases are themselves somewhat rough, so the surface begins to roughen, though there is still order to it. At 2000 K, the surface is still slightly ordered, but may enter a metastable phase of being completely roughened. At 2500 K, the surface is completely rough and randomized.

5 Conclusions

KMCInteractive is a tool that may be used to teach about MBE topics such as growth modes, surface smoothing, and surface morphology in a visual and interactive way previously unavailable. The program's speed is increased with a quad tree to sort the rate data and update after each iteration step, and a user-friendly interface defines the material and growth parameters, which can be changed dynamically during the simulation. Not discussed at length in this paper is the ability of KMCInteractive to demonstrate layer-by-layer growth, and how this growth mode affects the growth window to achieve a smooth surface when compared to step-flow growth on a miscut surface, but this topic may be easily explored using the program.

References

[1] The CRAY X-MP/48. `http://www.scd.ucar.edu/computers/gallery/cray/xmp/xmp.html`. Last retrieved 7 June, 2005.

[2] S. Clarke and D.D. Vvedensky. Origin of reflection high-energy electron-diffraction intensity oscillations during molecular-beam epitaxy: A computational modeling approach. *Physical Review Letters*, 58(21):2235–2238, May 1987.

[3] Martha A. Gallivan. *Modeling and Control of Epitaxial Thin Film Growth.* PhD thesis, California Institute of Technology, 2003.

[4] IBM. *IBM PowerPC 970FX RISC Microprocessor User's Manual.*

[5] Peter Kratzer. Applications of kinetics to surface science and simulations of epitaxial growth. `http://w3.rz-berlin.mpg.de/~kratzer/kmc/application1.pdf`, 2002. An overview of various KMC methods. Last retrieved 7 June, 2005.

[6] A. Madhukar. Far from equilibrium vapour phase growth of lattice matched III-V compound semiconductor interfaces: some basic concepts and Monte-Carlo computer simulations. *Surface Science*, 132:344–374, 1983.

[7] T. Shitara, D. D. Vvedensky, M. R. Willby, J. Zhang, J. H. Neave, and B. A. Joyce. Morphological model of reflection high-energy electron-diffraction intensity oscillations during epitaxial growth on GaAs(001). *Applied Physics Letters*, 60(12):1504–1506, 23 March 1992.

[8] Abraham Silberschatz and Peter Baer Galvin. *Operating System Concepts*. Addison-Wesley, fifth edition, 1998.

[9] Jeffrey Y. Tsao. *Materials Fundamentals of Molecular Beam Epitaxy*. Academic Press, Inc., 1993.

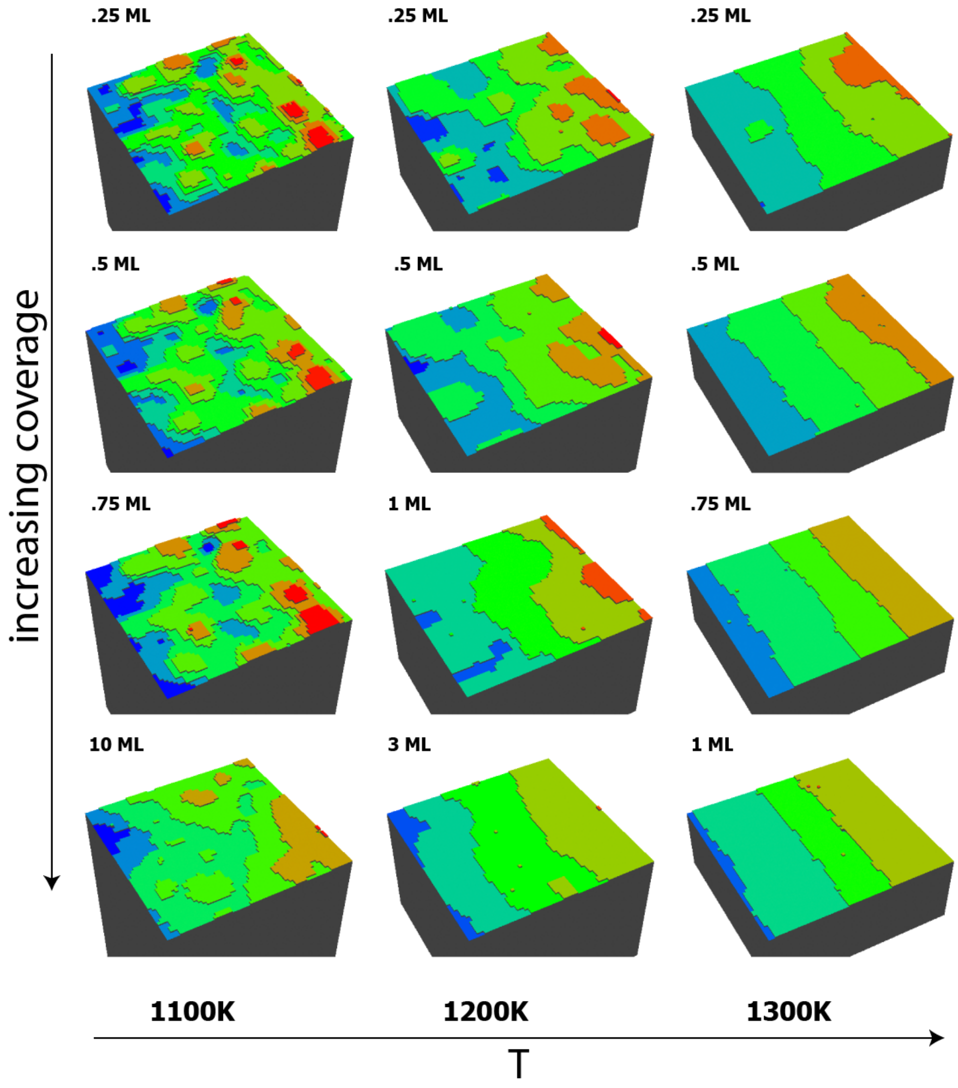


Figure 5: Surface recovery of a statistically roughened surface at 1100 K, 1200 K, and 1300 K. Colors cannot be directly compared between snapshots.

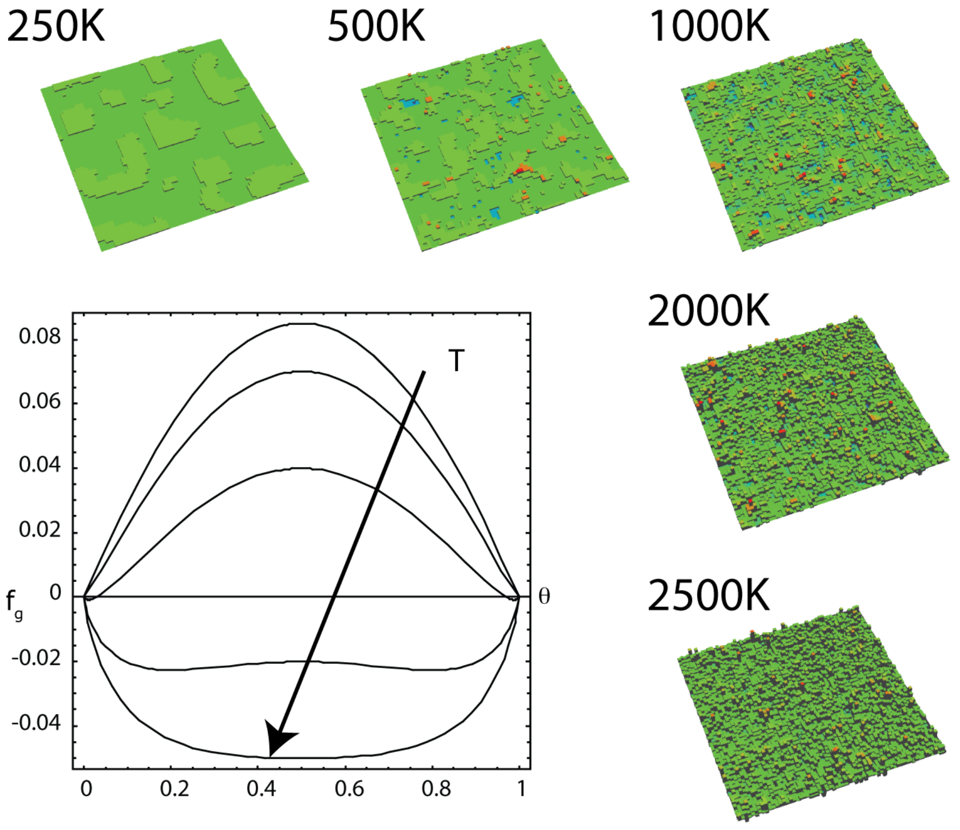


Figure 6: Snapshots of a surface with $\theta = 1/2$ annealed at 250 K, 500 K, 1000 K, 2000 K, and 2500 K. Plotted is the free energy of a single layer surface as a function of surface coverage for the temperatures simulated.