NORTHWESTERN UNIVERSITY

# DNA-Directed Nanoparticle Assembly via Multi-Scale Modeling and Simulation

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Applied Physics

By

Saijie Pan

EVANSTON, ILLINOIS

June 2018

## Abstract

DNA-Directed Nanoparticle Assembly via Multi-Scale Modeling and Simulation

Saijie Pan

Nano-scale materials possess many unique physical and chemical properties which are not found in bulk materials. The ability to synthesize these materials by design is one of the greatest challenges in materials science. Advances towards meeting this challenge will lead to discoveries in fields such as plasmonics, photonics, catalysis, and energy sciences. DNA-directed self assembly has emerged as a novel approach for generating nanoparticle superlattices, where nanoparticle "atoms" functionalized with a dense shell of DNA linkers, termed programmable atom equivalents (PAEs), are assembled into crystalline superlattices with tunable compositions, crystal symmetries, and lattice parameters. To bring the potential of this technique into real applications requires a deep understanding of the precise control of the spatial distribution and orientation of the nano-scale building blocks. Theoretical models and computer simulations can play an important role in the understanding of the assembly process over multiple length scales, and eventually predict various phase behaviours. This thesis studies DNA-directed nanoparticle assembly via multi-scale modeling and simulation. The original work can be divided into three parts:

(i) DNA-directed assembly is a well developed approach in constructing desired nano-scale architectures, while E-beam lithography is widely utilized for high resolution nano-scale

patterning. Recently, a new technique combining these two methods was developed to epitaxially grow DNA-mediated nanoparticle superlattices on patterned substrates with specific orientation and controllable sizes. However, defects were observed which restricted this technique from building large-scale superlattices for real applications. In order to optimize the epitaxial growth, we used molecular dynamics simulations to study the nature of the formation of these defects and further developed design rules to dramatically reduce defects.

(ii) Defects play an important role in materials science. Like any solid in nature, superlattices can contain different kinds of structural defects, which significantly alter their physical properties. They may provide material advantages or disadvantages. Further development of these materials requires a deeper understanding and good control over structural defect formation. We used Monte Carlo simulations to conduct a systematic study of defect formation in epitaxial growth of nanoparticle superlattices at a much larger length scale. The simulations show two main results. First, structural defects have long range correlations and form one-dimensional clusters with an exponential length distribution. Second, these linear defects exhibit spontaneous symmetry breaking and undergo a liquid crystal phase transition. Furthermore, we introduced a mean-field theoretical approach, which is in strong agreement with the simulation results.

(iii) In previous studies, we focused on spherical nanoparticle building blocks. However, non-spherical nanoparticles are ideal building blocks for self assembly into various functional nanomaterials due to their unique and anisotropic physical properties. With the advent of methods for preparing non-spherical building blocks, DNA-directed assembly of anisotropic nanoparticles has attracted the interest of both experimental and computational research. The challenge is how to assemble these anisotropic nanoparticles into

required target structures to obtain desired properties. Here we conducted multi-scale molecular dynamics simulations and Monte Carlo simulations to study the DNA-directed assembly of tetrahedron nanoparticles. We observed quasicrystalline structures with five fold symmetry in the assemblies. Further, we demonstrated that icosahedral nanocages can be formed by truncated tetrahedron building blocks with specific preconfiguration.

## Acknowledgements

First and foremost, I would like to thank my advisor, Professor Monica Olvera de la Cruz, for guiding and supporting me over the years. I really enjoyed every meeting and conversation with you. Thank you for all that you have taught me. You have set an example of excellence for me as a scientist, mentor and friend.

I would also like to thank the rest of my thesis committee members, Professor Pulak Dutta and Professor George C. Schatz for their guidance through this process. Your ideas and feedback have been absolutely invaluable.

Much of my success is due to the encouraging environment of the Olvera group. I would like to express my gratitude to all the group members, present and past. I would especially like to thank Dr. Ting Li for mentoring me during my first few years at Northwestern. I was extremely lucky to have you as the one who led me into the ‘DNA-nanoparticle’ field. I would also like to thank Dr. Niels Boon, for being a friendly office mate, a good collaborator, and a true friend.

My PhD journey would not have been so colorful and enjoyable without the support of my friends at Northwestern. I would like to thank two of my fantastic roommates, Ben and Jian. You are like family to me and have made Seabury my second home. I would like to thank all those at NCCF and Reading Club. Thank you for your friendship and countless meaningful Friday nights and wonderful Saturday nights. I would like to give special thanks to my best friend WeiTing. Thank you for the company and childish games during those days when I was struggling with the thesis writing. I would like to thank John and Jessica for your help and support from the Evanston community. You have made Evanston my second hometown.

Last but not least, I want to thank my amazing family for their endless love, support and constant encouragement. In particular, I would like to thank my fiancée, Yinge. Your love and support have always been my strength. Your patience and sacrifice will remain my inspiration throughout my life. Thank you, dear!

# Table of Contents

## List of Figures

CHAPTER 1

# Introduction

## 1.1. Motivation

DNA is not only genetic code, but also a means to design self-assembling nanomaterials[**7, 8**]. Grafting DNA onto nanoparticles can, in principle, 'program' them with information that tells them exactly how to self-assemble. Macroscopic assemblies of nanoscale materials possess unique electronic, optical, mechanical and magnetic properties, offering potential applications in medical diagnostics (programmable recognition) [**9, 10, 11, 12**], catalysis [**13, 14, 15**], energy conversion [**16**], nanophotonics [**17, 18**], and as plasmonic nanomaterials [**19, 20, 21, 22, 23, 24**]. A grand scientific challenge nowadays is to learn how to design and create these assemblies by controlling properties of nanoscale building blocks. To meet this challenge, theoretical modeling and computer simulations are proving to be essential.

Below, I will give a brief overview of the experimental developments and theoretical studies to show an overall understanding of DNA-directed assembly of nanoparticles.

## 1.2. Summary of Previous Work

### 1.2.1. Experimental developments

The idea of the DNA-directed assembly of nanoparticles was first introduced by both Mirkin *et al.* and Alivisatos *et al.* in 1996 [**7, 8**]. In one paper, Mirkin and co-workers utilized short, stiff double-stranded DNA (dsDNA) chains with complementary single-stranded DNA (ssDNA) ends to assemble colloidal gold nanoparticles into macroscopic aggregates [**7**]. This aggregation process is reversible by varying the temperature, owing to the DNA hybridization and dehybridization between complementary ssDNA. In the other paper, Alivisatos and co-workers utilized long, soft ssDNA chains to assemble gold nanoparticles into small "nanocrystal molecules" by DNA hybridization [**8**]. Both groups discussed the prospect of using DNA-coated nanoparticles as building blocks to construct more complex two- and three-dimensional nanostructures that do not exist in nature.

Based on the strategies mentioned above, substantial advances have been achieved in the DNA-directed assembly of nanoparticles [**25, 26, 27, 28, 29, 30, 1, 31, 32, 16, 11, 33, 2, 34, 35, 36, 37, 38, 39, 24**]. In 2008, the first experimental observations of nanoparticle superlattices were reported by the Gang group and Mirkin group [**32, 1**]. Body-centered cubic (BCC) and face-centered cubic (FCC) crystalline lattices were realized. These findings prove that synthetically programmable colloidal crystallization is achievable. Specifically, Mirkin and co-workers introduced two distinct schemes into DNA-nanoparticle assembly for the formation of BCC and FCC lattices, as shown in Figure 1.1. One scheme was designed for binary systems in which DNA linkers on the two building blocks are complementary to each other (such as $-$AAGG and TTCC$-$). This scheme led to BCC superlattices. The other scheme was designed for unary systems in which nanoparticles are coated with self-complementary DNA linkers (such as $-$GCGC). Instead the FCC superlattice was realized.

The successful assembly of these basic crystalline structures represents the beginning of the next important stage of DNA-programmable assembly, DNA-induced nanoparticle crystallization. By tuning the size of nanoparticles and the length or sequence of DNA chains, a variety of superlattices have been realized [**31, 32, 1, 33, 2**].

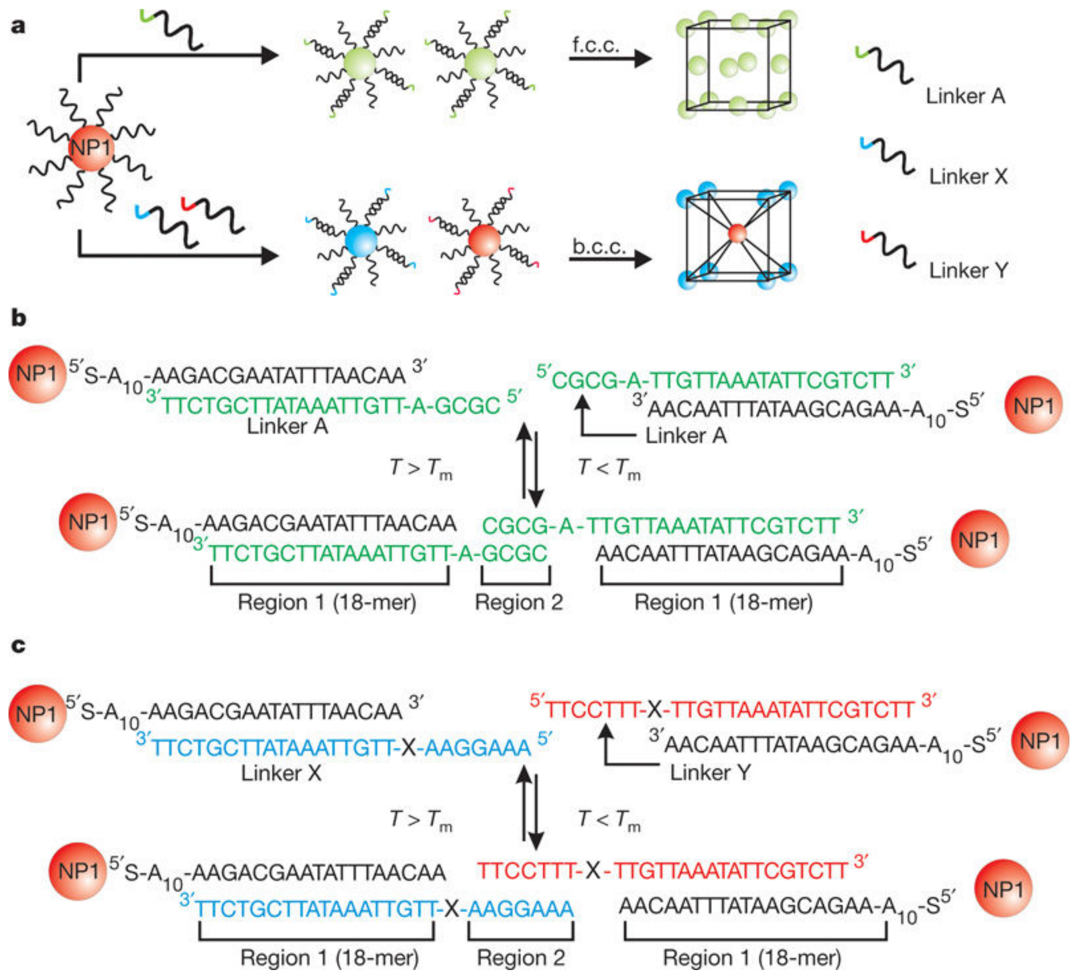

a
f.c.c.
b.c.c.
NP1
Linker A
Linker X
Linker Y
b
5′S-A10-AAGACGAATATTTAACAA 3′
3′TTCTGCTTATAAATTGTT-A-GCGC 5′
5′CGCG-A-TTGTTAAATATTCGTCTT 3′
3′AACAATTTATAAGCAGAA-A10-S5′
T > Tm
T < Tm
Region 1 (18-mer)
Region 2
c
3′TTCTGCTTATAAATTGTT-X-AAGGAAA 5′
5′TTCCTTT-X-TTGTTAAATATTCGTCTT 3′

Figure 1.1. Schemes of nanoparticle assembly method. (a) DNA-NPs can be programmed to assemble into FCC and BCC structures by changing the sequence of DNA linkers. (b) Unary system where self-complementary linkers are used. (c) Binary system in which the linkers on the two building blocks are complementary to each other. X in the DNA sequence denotes the flexor region: A, PEG6 or no base. NP1 indicates that the same NPs were used in all experiments. (Adopted from Ref. [**1**].)

In 2011, Macfarlane *et al.* [**2**] proposed six design rules to successfully predict nine distinct colloidal crystal structures, including Simple Cubic, FCC, hexagonal close-packed (HCP), BCC, CsCl, NaCl, $AlB_2$, $Cr_3Si$, and $Cs_6C_{60}$ symmetries (Figure 1.2).

Transition from the stage of polycrystalline superlattices to the stage of faceted single crystals took place in 2014[**35**]. The Mirkin group showed that crystal structure with a specific and uniform

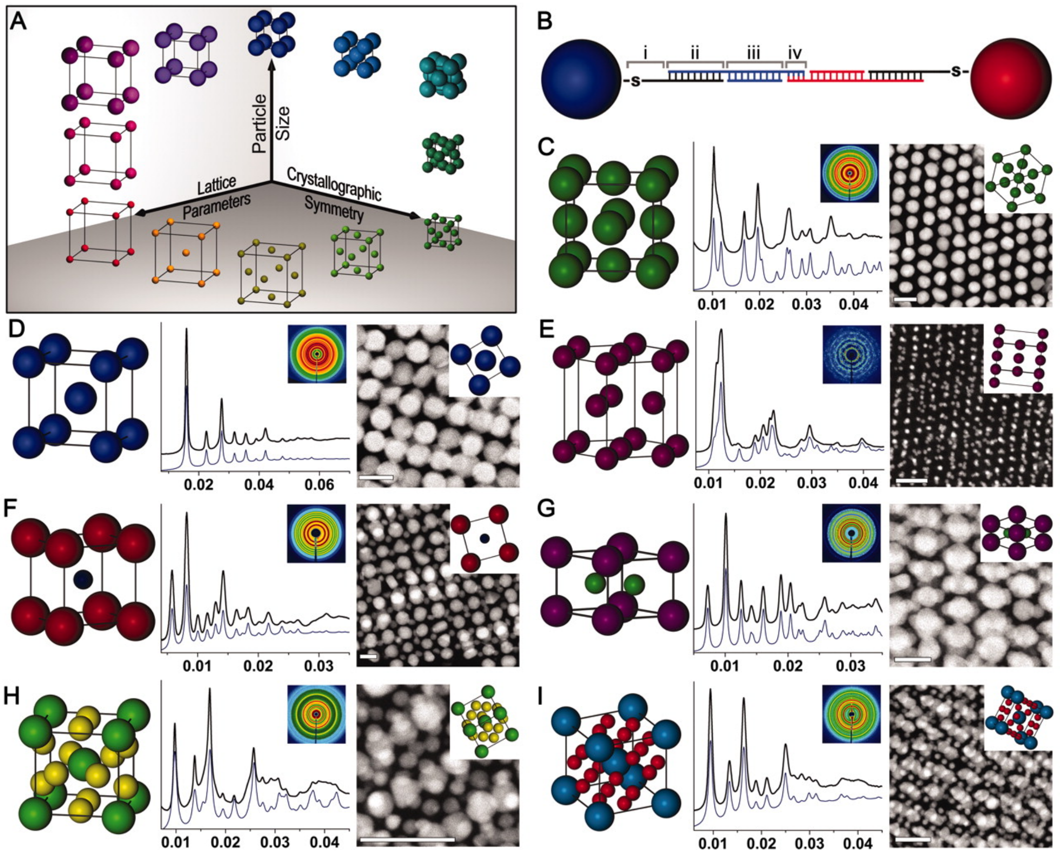


Figure 1.2. (A) Nanoparticle superlattice engineering with DNA. (B) The DNA chain is consisted of (i) an alkylthiolmoiety and 10-base nonbinding region, (ii) a recognition sequence that binds to a DNA linker, (iii) a spacer sequence of programmable length to control interparticle distances, and (iv) a 'sticky end' sequence that drives nanoparticle assembly via DNA hybridization interactions. (C) to (I) The superlattices reported herein are isostructural with (C) FCC, (D) BCC, (E) HCP, (F) CsCl, (G) $AlB_2$, (H) $Cr_3Si$, and (I) $Cs_6C_{60}$ lattices. From left to right, each panel contains a model unit cell, x-ray diffraction (SAXS) patterns, and a TEM image with the unit cell viewed along the appropriate projection axis (inset). (Adopted from Ref. [**2**].)

crystal habit was observed under special conditions of slow cooling and over several days. The synthesized nanoparticle assemblies were confirmed to have the Wulff equilibrium crystal structure that was predicted from theoretical considerations and molecular dynamics (MD) simulations, which were done by the Olvera de la Cruz group.

Although nanoparticle assemblies can have a broad range of crystal structures, only spherical nanoparticles with BCC symmetry yields single crystals with well-defined crystal lattices. In 2016, anisotropic nanoparticles with reduced symmetry were used to overcome this limitation[**3**]. Cubes, octahedra, and rhombic dodecahedra were investigated. They yielded single crystals with cube, rhombic dodecahedron, and octahedron crystal habits, respectively as in Figure 1.3.

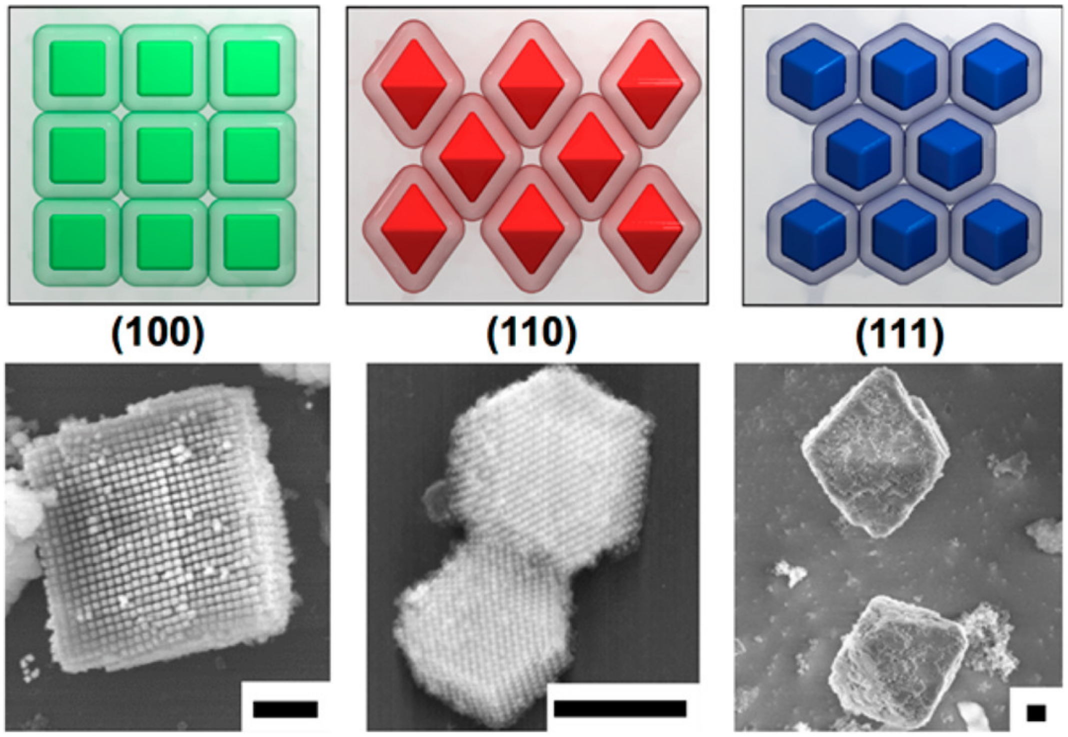


Figure 1.3. Nanoparticle shape can be used to control crystal habit in DNA-mediated nanoparticle crystallization. Each shape crystallizes into a lattice with a different closest-packed plane (top) and crystal habit (bottom). Cube, octahedron, and rhombic dodecahedron nanoparticles (left to right) are shown with cube, rhombic dodecahedron, and octahedron crystal habits, respectively. Scale bars: 1 $\mu$m. (Adopted from Ref. [**3**].)

### 1.2.2. Theoretical developments

Along with the progress in experiments, theoretical analysis [**40, 2, 41**] and simulations [**42, 43, 44, 45, 46, 47, 48, 49**] have been developed to not only confirm the experimental results but also stimulate the experiments by providing new predictions. Most of the theoretical approaches are based on mean-field or static models which are greatly simplified: The models only consider (i) the average overlapping volume of two complementary building blocks [**2**], (ii) the volume fraction and the average coordination number of each type of building block [**40**], or (iii) the nanoparticle packing density [**41**]. As a result, they do not address important aspects of DNA chain conformations or dynamics. To circumvent these problems, computer simulations with numerical models have been introduced. Due to the large system sizes and time scales required in the study of DNA-directed nanoparticle assembly, coarse-grained models are the most efficient way to describe the dynamics of self-assembly. Various coarse-grained models have been developed over the past few years [**50, 45, 47, 49**]. The model proposed by the Olvera de la Cruz group [**49**] is to date the most detailed model that is capable of describing the dynamics and thermodynamics of the self-assembly process.

## 1.3. Outline of Research

The objective of my thesis research is to study the DNA-directed self assembly of nanoparticles (both isotropic and anisotropic) into large scale ordered nanostructures via theoretical modeling and computational simulations. Following this introduction, Chapter 2 and Chapter 3 focus on the study of epitaxial growth of DNA-coated nanoparticles on a patterned substrate. In Chapter 2, a scale-accurate coarse-grained molecular dynamics model was used to study and predict defect formation in the epitaxial growth. We developed design rules to reduce defects. Furthermore, in Chapter 3, we used Monte Carlo simulations to systematically study the formation of defects in the above system. With the advantage of Monte Carlo simulations, we were able to study much larger systems and discovered a new phase transition. In Chapter 4, we used multi-scale modeling and simulations to study the DNA-directed assembly of anisotropic nanoparticles, especially tetrahedron and truncated tetrahedron building blocks. For regular tetrahedron and truncated tetrahedron systems, we showed how they may self assemble into quasicrystalline structures and icosahedral nanocages, respectively. Finally, Chapter 5 summarizes the work performed and discusses future directions for research.

CHAPTER 2

# Molecular Dynamics Simulation of DNA-Directed Assembly of Nanoparticle Superlattices Using Patterned Templates

DNA-directed assembly is a well developed approach in constructing desired nano-architectures. On the other hand, E-beam lithography is widely utilized for high resolution nano-scale patterning. Recently, a new technique combining these two methods was developed to epitaxially grow DNA-mediated nanoparticle superlattices on patterned substrate. However, defects are observed in epitaxial layers which restricts this technique from building large-scale superlattices for real applications. Here we use molecular dynamics simulations to study and predict defect formation on adsorbed superlattice monolayers. We demonstrate that this epitaxial growth is energetically driven by maximizing DNA hybridization between the epitaxial layer and the substrate and that the shape anisotropy of the DNA-mediated template posts leads to structural defects. We also develop design rules to dramatically reduce defects on epitaxial layers. Ultimately, with the assist of the computational study, this technique will open the door to constructing well-ordered, three-dimensional novel nanomaterials.

## 2.1. Introduction

DNA-programmable assembly of nanoparticles is a versatile technique in nano-architecture fabrication[**1, 2, 51, 35, 52, 53**]. Nanoparticle superlattices of various lattice structures can be obtained by precisely controlled parameters of the DNA linkers attached to those nanoparticles[**31, 32, 1, 33, 2**]. Materials generated from well ordered nanoparticles can have unique magnetic[**54**], electronic[**55**], photonic[**18, 17**], and plasmonic[**19, 20, 21, 22, 23, 56**] properties. Recently, it was reported that three-dimensional assemblies of nanoparticles with a controllable orientation can be achieved by layer by layer epitaxial growth on DNA-modified patterned substrates[**57**]. DNA hybridization between complimentary DNA linkers on nanoparticles and that on the substrate can precisely lead the DNA-coated nanoparticles to desired template sites and form the epitaxial layer. The adsorbed monolayer can then act as another substrate layer, on which a new monolayer can be epitaxially grown in a similar way. However, it was observed in experiments that the epitaxial layer generally contained a number of defects including misplaced nanoparticles and vacancies. Considering the mechanism of epitaxial growth, these defects will be extended during growth into the next adsorbed layer. The new monolayer can thus be of no higher quality than its sublayer. Therefore, to design nearly perfect, device-quality multilayer superlattices, it is significant to reduce defect densities in epitaxial layers. The goal of this research is to understand the origin of the defects in epitaxial layers and to avoid their formation.

Here, we apply a scale-accurate coarse-grained model[**49**] to this system. Using molecular dynamics simulations, we study the epitaxial growth of the DNA-coated nanoparticles on the patterned substrates. With the same parameters as in experiments[**57**], we reproduce the defective attached monolayer. We then study this epitaxial growth on an ideal template which is not yet feasible experimentally. Surprisingly, a perfect epitaxial layer is obtained. The simulation work seeks to find out why defects form under the experimental conditions and try to give optimal parameters which can be applied to experiments to generate a defect-free epitaxial layer, which is required to prepare three-dimensional, well ordered nanostructures.

## 2.2. Model and Methods

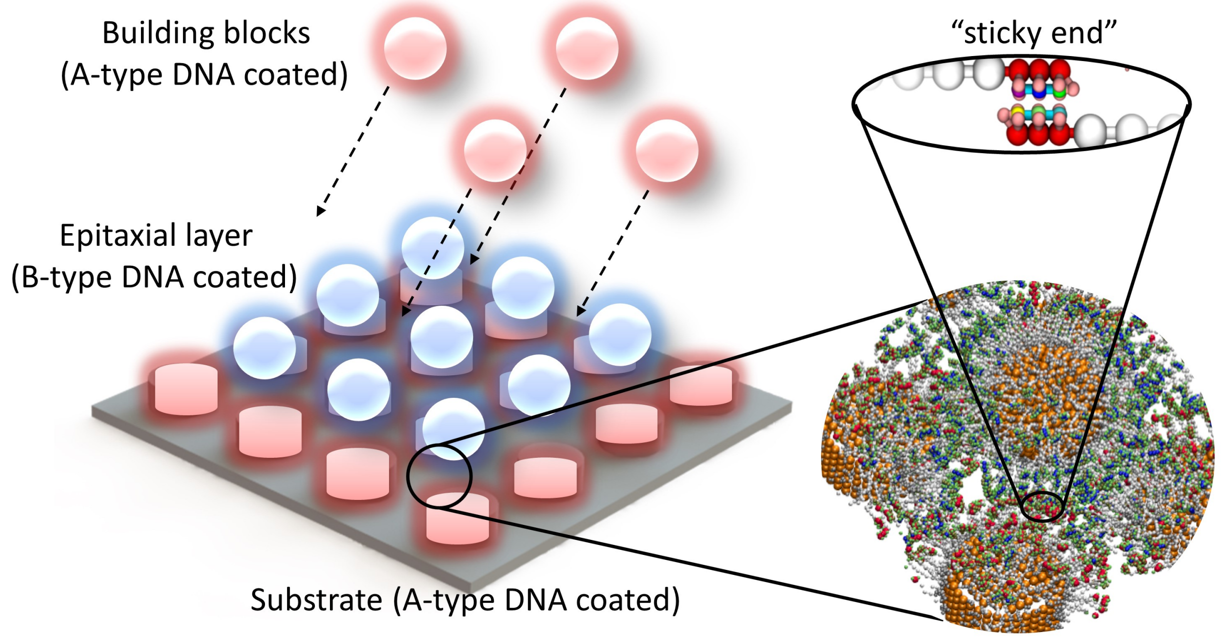


Figure 2.1. Schematics of epitaxial growth of DNA-coated nanoparticles on patterned BCC (100) substrates (three layers to show layer by layer growth). Zoom in window shows details of DNA hybridization between complementary "sticky ends".

In this work, we study the epitaxial growth of DNA-coated gold nanoparticles on a BCC (100) patterned substrate. The schematic view of the systems is shown in Figure 3.2. The template is a two-dimensional square array of nano-fabricated gold posts which is designed to be equivalent to a BCC (100) plane. All posts on the substrate are functionalized with A-type DNA linkers. The patterned substrate is then exposed to gold nanoparticles coated with B-type DNA linkers in the solution. A-type and B-type DNA linkers are programmed in such a way that DNA hybridization can only occur between distinct types of DNA linkers, suppressing nucleation in solution and island growth on the substrate. The BCC superlattice can be grown layer by layer by exposing the surface alternatively to A-type and B-type building blocks.

### 2.2.1. Coarse-grained model of building blocks

In order to capture key aspects of the DNA assembly process, a scale-accurate coarse-grained model is used for DNA-functionalized nanoparticles. This model is a minor variation of the one previously developed in Travesset group[**47**], which in turn is a generalization of the model developed by Sciortino, Starr, and collaborators[**43**]. Details of our model have been discussed in previous papers[**49, 58**]. In brief, each DNA-nanoparticle building block is modeled as a rigid body with a fixed number of beads on the surface, to which hundreds of DNA linkers are attached. Each DNA linker is composed of a series of beads representing different regions. Except for those in the "sticky end" region, most beads are used to control the properties of the linker such as hydrodynamic length, stiffness and so on. The "sticky end" region is used to mimic the directional hydrogen bonds between complementary bases (A-T, C-G). For example, A-type DNA linker has a -AACCCAA "sticky end" while B-type DNA linker has a complementary -TTGGGTT "sticky end".

### 2.2.2. Molecular dynamics simulation

Molecular dynamics simulations are performed with the HOOMD-Blue package[**59, 60, 61, 62**] under the $NVT$ ensemble with the temperature controlled via a Nosé-Hoover thermostat[**63, 64**]. In the simulation box, a substrate plane with $n \times n$ template posts is placed at $z = 0$. The substrate has a mirror symmetry about the $z = 0$ plane so that both the top and bottom surfaces of the template can be adsorbed by nanoparticles. In this way, we save computational time by doubling the template size. All the building blocks in the solution are originally located in a region furthest away from the substrate. The total number of building blocks is 1.5 times that of sites for adsorption. Periodic boundary conditions are applied to all three dimensions of the system. Consistent with the experimental values[**57**], the lattice parameter is set to $a = 62$ nm and the diameter of a gold nanoparticle is set to 30 nm. To form a perfect BCC structure, the hydrodynamic diameter of a DNA-coated nanoparticle should be $D_{hydro} = \sqrt{3}/2 \times a \approx 53.7$ nm. DNA linkers are precisely programmed to satisfy this condition in simulations.

### 2.2.3. A real case: cylindrical template posts

The template acts as a seed single crystal in epitaxial growth. To grow a BCC superlattice, the template is designed to be equivalent to a BCC (100) plane of nanoparticle superlattices. However, the fabrication of such template is not feasible in experiment due to the limitation of the lithography technique. Instead, a square array of cylindrical posts is fabricated in experiment to best resemble a BCC (100) monolayer of nanoparticles. To reproduce the experimental findings, we create the same patterned template in our simulation. The height of the posts is set suitably to avoid interaction between two attached layers above and below the substrate. A-type DNA linkers are attached to the surface of the posts with the same grafting density as DNA-coated nanoparticles.

### 2.2.4. An ideal case: hemispherical template posts

In simulation, we can easily overcome the limitation of fabrication process. To study the epitaxial growth on an ideally patterned template, we create the template using a square array of hemispherical posts which is identical to the (100) surface of a BCC nanoparticle superlattice. As the substrate in the simulation box has a mirror symmetry about the substrate plane, we actually create a monolayer of DNA-coated nanoparticles to represent the substrate. These nanoparticles are the same as the building blocks in the solution with the only difference being the type of “sticky end”. Templates are coated with A-type DNA linkers while nanoparticles in solution are coated with B-type DNA linkers.

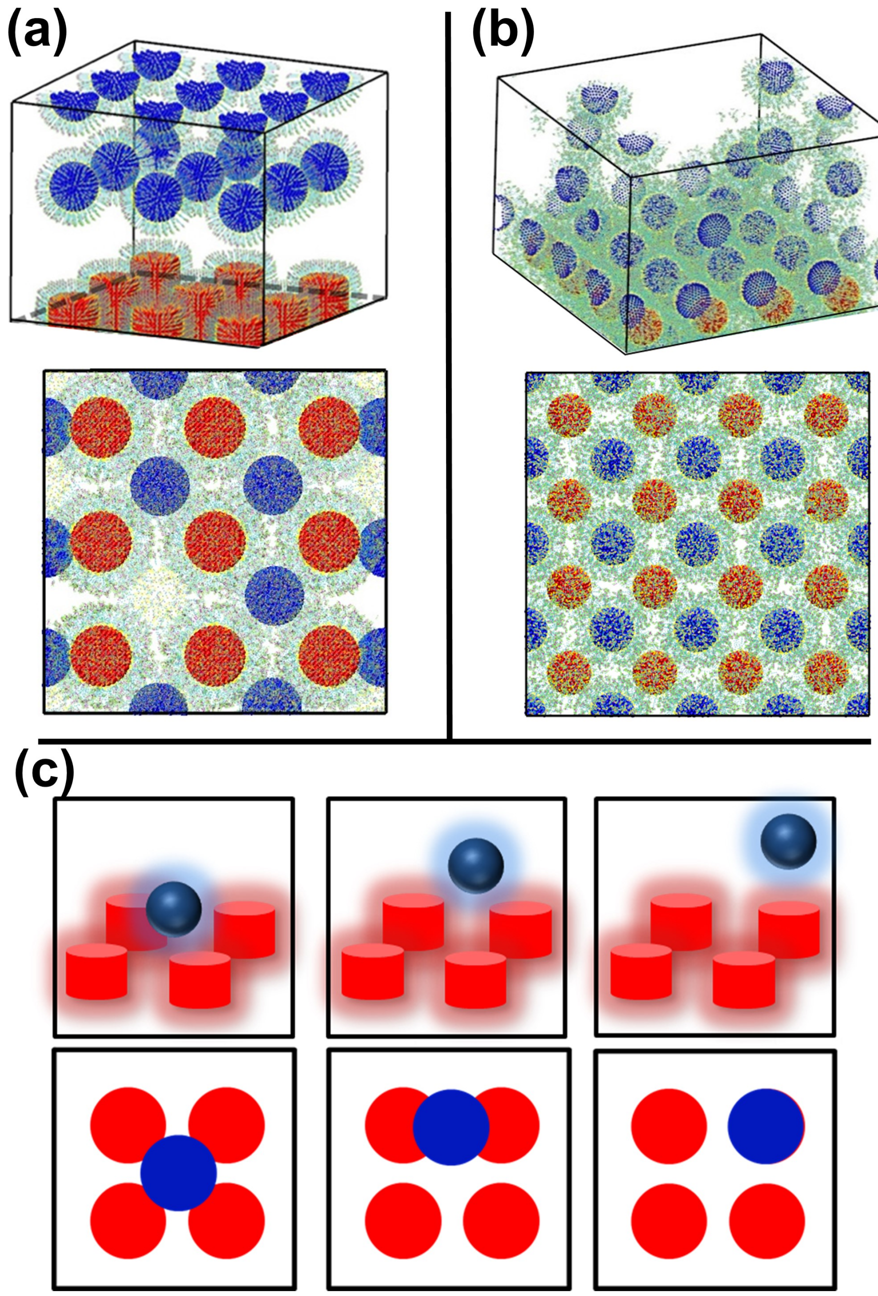


Figure 2.2. (a) Snapshots of system states before and after the system reaches equilibrium in the cylindrical template case. All snapshots in this manuscript are generated with the Visual Molecular Dynamics (VMD) package[**4**] and rendered with Tachyon ray-tracer[**5**]. (b) Snapshots of system states before and after the system reaches equilibrium in the hemispherical template case. (c) Schematics of three different types of attachments that are observed in the simulations. From left to right, they are center-, edge-, and corner-bound attachments.

## 2.3. Results and Discussion

### 2.3.1. Attachment profile at equilibrium for cylindrical templates

Figure 2.2(a) shows simulation snapshots in the cylindrical template case. In equlibrium most of the nanoparticles are adsorbed on the center-bound sites of the substrate which form another BCC (100) layer. Besides the desired attached sites, we observe one vacancy and three edge-bound nanoparticles. A few independent simulations of the system with the same parameters have been employed to statistically calculate system quantities. Similar attachment profiles with defects are obtained. Figure 2.2(c) shows all three types of attachment in experiments which are also observed in simulations.

### 2.3.2. Attachment profile at equilibrium for hemispherical templates

For an ideal case shown in Figure 2.2(b), all the available center-bound sites are covered with nanoparticles which form a perfect (100) layer of a BCC superlattice. The attached layer can then be used as a template to grow another layer epitaxially which will lead to our goal of building up three-dimensional superlattices layer by layer. Again the perfect attachment profile is repeatable in many independent copies of simulations. Comparing between the attachment profiles for the ideal and real cases, we can at least conclude that the difference in template design leads to the defect structure in the epitaxial layer.

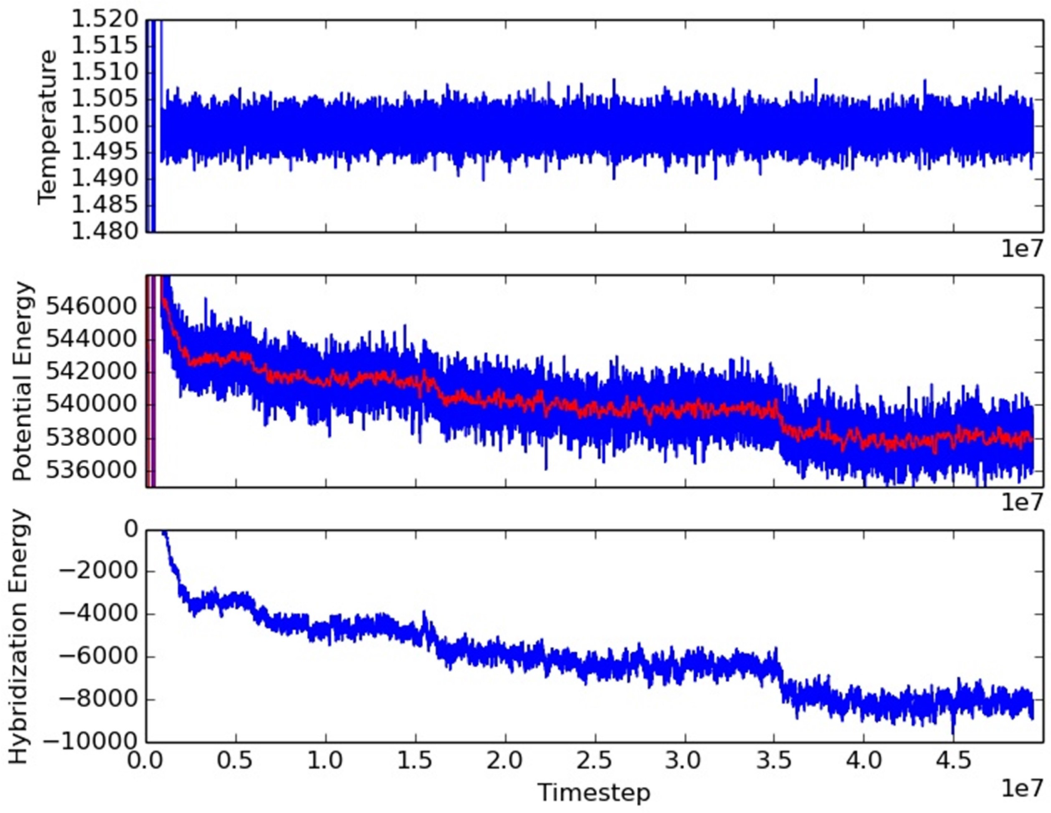


Figure 2.3. Top: Temperature of the system as a function of time step. It is fixed to 1.5 in the NVT ensemble. Middle: Total potential energy of the whole system as a function of time step. The average of the potential energy in a short period is also plotted as a red curve. Bottom: Hybridization energy resulted from the DNA hybridization as a function of time step.

### 2.3.3. System quantities as as a function of time step

Figure 3.5 shows three system quantities - temperature, potential energy and DNA hybridization energy as a function of time step in a simulation during the process of epitaxial growth. We set the system as canonical ensemble in which temperature is fixed during the process so that the kinetic energy of the system remains the same. The total potential energy is lowered as the system evolves and we can associate each drop of energy to new nanoparticle adsorption or nanoparticle diffusion from higher energy site to lower energy site. We calculate the DNA hybridization energy as a function of time step and find that the trend of the hybridization energy is nearly identical to that of the averaged potential energy over a few hundred time steps during the entire range of time steps. We can conclude that except for the DNA hybridization energy, other terms of energy like harmonic potential, angle potential and hard core repulsion have no influence on expectation of the total energy but instead contribute to the fluctuation of the total energy. The change of the total internal energy is dominated by that of the hybridization energy, which demonstrates that the epitaxial growth is energetically driven by the DNA hybridization process. It is the difference of DNA hybridization in ideal case and real case that gives defect-free or defect-rich epitaxial layers. Next step it is reasonable to analyze how the template shape affects the DNA hybridization.

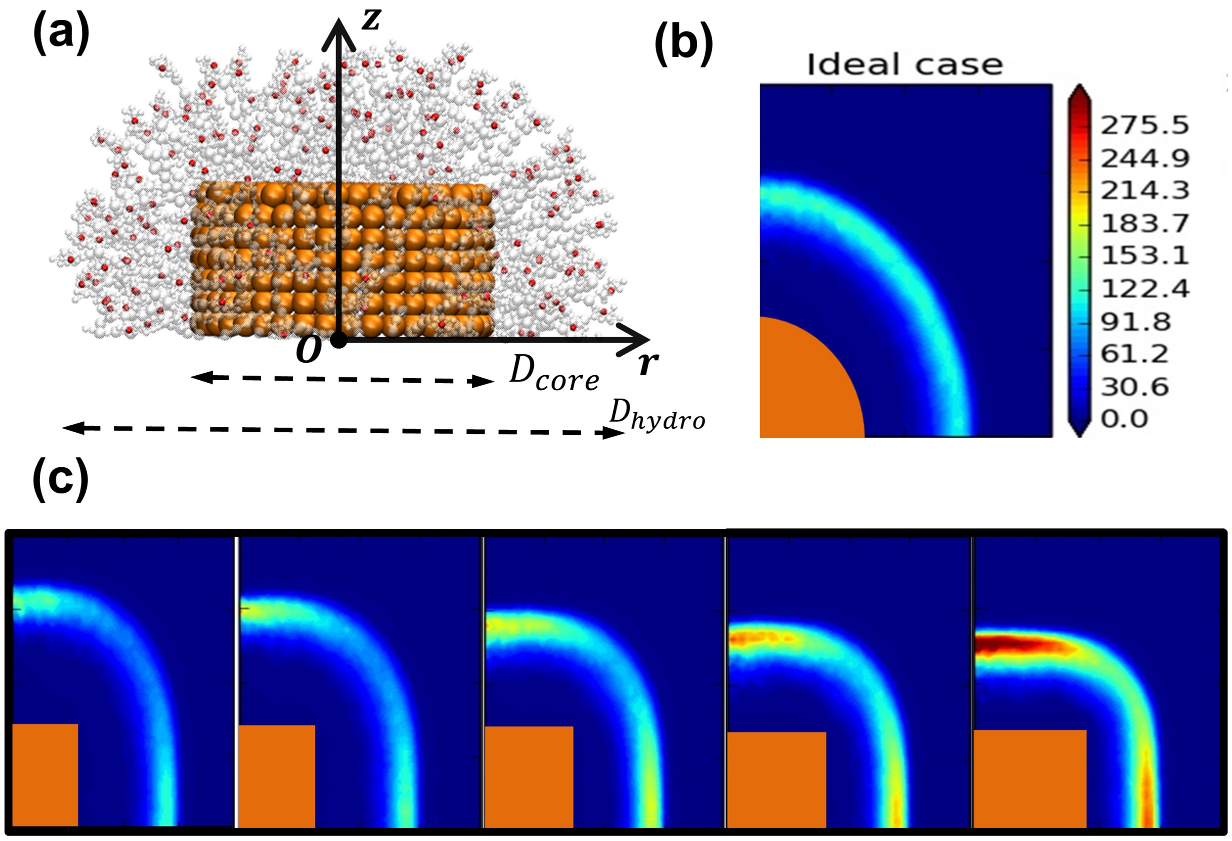


Figure 2.4. Shape anisotropy analysis of the templates. (a) A picture showing the profile of a DNA capped cylindrical template (one bead in the “sticky end” of each DNA chain is colored in red). (b) A contour plot of the local “sticky end” density distribution around a hemispherical template. (c) From left to right: five contour plots of the local “sticky end” density distribution with increasing core sizes and fixed hydrodynamic size.

## 2.3.4. Probability of DNA hybridization around different templates

The difference of attachment profiles results from the difference of shapes of template posts. To reduce defects in the epitaxial layer, hemispherical template posts are the best choice. However, fabricating such posts is not technically feasible by the electron beam lithography method. Therefore, it is important to study epitaxial growth on different templates that are currently experimentally feasible. It is known that DNA hybridization is crucial for the adsorption of epitaxial layers. The adsorption strength is proportional to the number of hybridized ”stick ends” during the process. Therefore, it is crucial to find the distribution of “sticky ends” around each template post. Both the hemispherical and cylindrical templates have azimuthal symmetry. The volume density only depends on coordinates $r$ and $z$ in cylindrical coordinates (see Figure 3.6(a)). Therefore, we count the number of “sticky ends” between $r, r + dr$ and $z, z + dz$ and average over a long period of time steps after the system reaches equilibrium. We first calculate the “sticky end” density distribution around a hemispherical template post as shown in Figure 3.6(b). As expected, the “sticky end” density is evenly distributed in a thin shell region whose mean radius is the hydrodynamic radius.

Each attached nanoparticle has 4, 2 and 1 binding posts for center, edge and corner attachments, as shown in Figure 2.2. The ratio of the hybridization energy is 4:2:1 correspondingly. Nearly all nanoparticles tend to attach to the center-bound sites which are energetically favorable. For the real case using cylindrical posts, the surface coordination number remains the same but the local density of “sticky ends” changes, as shown in Figure 3.6(c). Moreover, as the shape anisotropy increases when increasing the template core size, the local density of “sticky ends” in center-bound contact regions decreases relative to the other two types of attachment. For a template with large core size, nanoparticles have more chance to bind to edge-bound or corner-bound sites forming defects. On the other hand, if very small core size is selected to weaken shape anisotropy, the absolute density of “sticky ends” is also reduced since fewer DNA linkers can attach to the surface of the core, which leads to a vacancy-rich epitaxial layer. Therefore, there should be an optimal condition between these two extreme cases.

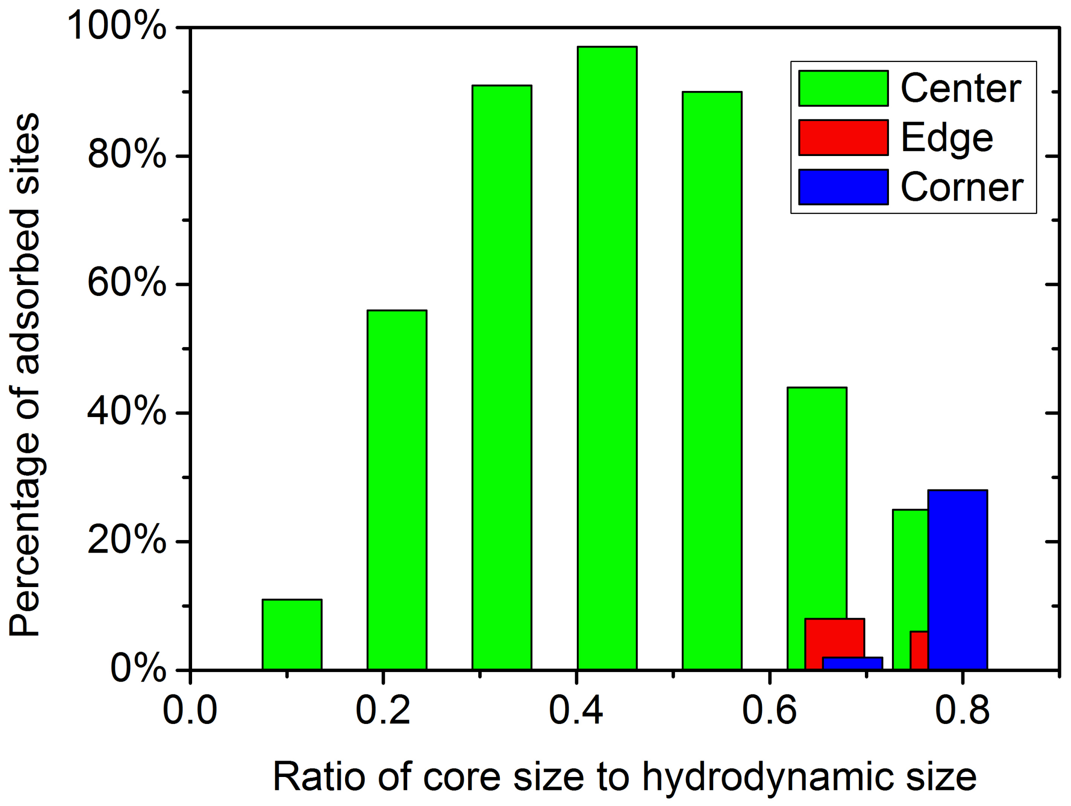


Figure 2.5. Percentage of the adsorbed particles as a function of the ratio of core size to hydrodynamic size.

### 2.3.5. Optimization of template design

We study cylindrical templates in order to find an optimal template that is available for fabrication in experiments. We fix the hydrodynamic diameter $D_{hydro}$ of DNA-coated templates to that of the DNA-nanoparticles and vary the template core diameter $D_{core}$. The DNA grafting area density is fixed for all DNA functionalization. For each template, 20 independent simulations have been employed to statistically calculate the percent of different types of adsorbed particles. Figure 3.7 shows that percent of center-bound nanoparticles achieves a maximum value when the ratio $D_{core}/D_{hydro}$ is around 0.45. In the optimal condition, more than 95$\%$ sites are occupied by center-bound particles in a epitaxial layer and there are no defects besides vacancies.

## 2.4. Conclusions

Coarse-grained molecular dynamics simulations are used here to investigate the DNA-directed assembly of nanoparticles on patterned substrates with the goal of building defect-free three-dimensional superlattices. The simulations reveal that DNA-coated nanoparticles can self-assemble onto DNA-coated templates driven by the DNA hybridization between complementary DNA chains linked on nanoparticles and template posts.

We focus our study on the formation and structure of the first monolayer epitaxially grown on a BCC (100) plane substrate as done in the experiments. We find that the shape anisotropy of cylindrical template posts contributes to defect formation in the epitaxial layer. We conclude that reducing defects was to adjust the spatial probability of DNA hybridization by precisely controlling parameters such as shape of the template post, size of the template core, length of the DNA linkers, and find the optimal parameters for fabricating two-dimensional template patterns to dramatically reduce defects on adsorbed monolayers.

Furthermore, we anticipate that our computational study can be applied to the epitaxial growth of other superlattice structures. The simulation model is proved to be an easy and useful tool to obtain crucial experimental parameters which are generally difficult or expensive to obtain from experiments. As a result, we envision that with the assist of our simulation study, this technique combining both DNA-directed assembly and template-directed assembly will lead to device fab-rications of well ordered, three-dimensional nanomaterials with magnetic, electronic, photonic, plasmonic or other unique properties.

CHAPTER 3

# Liquid Crystal Phase Transition in Epitaxial Monolayers of DNA Functionalized Nanoparticle Superlattices

Epitaxial growth of DNA functionalized nanoparticles is used to grow extended superlattices with a preferred orientation for optimizing the physical properties of metamaterials for real applications. Like any solid in nature, superlattices can contain different kinds of structural defects, which significantly alter their physical properties. Further development of these materials requires a deeper understanding of, as well as precise control over, structural defect formation. Here we use Monte Carlo simulations to conduct a systematic study of the equilibrium structures of the adsorbed nanoparticle monolayers by changing the binding energies of different attachment sites. The simulations show two main results. First, the structural defects form one-dimensional clusters with an exponential length distribution. Second, these linear defects exhibit spontaneous symmetry breaking and undergo a liquid crystal phase transition. Subsequently, a mean-field approach is introduced to provide theoretical descriptions for the system. Our theory matches with the simulation results. We anticipate that this theoretical framework will be highly applicable to other two-dimensional assemblies. Our work demonstrates that defects can be engineered to design two-dimensional superlattices with interesting physical properties.

## 3.1. Introduction

The study of the self-assembly of nanoparticles is an important area in nanotechnology research [**65, 52**], focusing on the creation of nanomaterials with electronic, photonic, plasmonic, mechanical or magnetic properties.[**54, 55, 18, 17, 19, 20, 21, 22, 23, 56, 66**] Such properties are largely unseen in nature, and are a direct result of the microscopic structure of the assembly. Therefore, the assembly must be designed and controlled at the nanometer scale. DNA-directed assembly[**67**] is a particularly attractive approach in constructing these materials because the DNA functionalized nanoparticle building blocks can be precisely programmed with controllable size, shape, composition and bonding interactions.[**68, 69, 70, 71, 72, 73, 74, 75, 1, 76, 77**]

Recently, researchers have studied the epitaxial growth of DNA functionalized nanoparticles on patterned substrates, achieving templated growth of nanoparticle superlattices with controllable orientation and sizes that extend from the microscale to the mesoscale.[**6**] Both the template design and DNA modifications of the template determine the binding energies of nanoparticles at different binding sites. Nanoparticles in solution coated with complementary DNA can attach to specific binding sites, a process that can be understood as site-specific adsorption. However, defects are inevitable in the attached monolayers. They arise from the binding of a nanoparticle at unexpected locations (Figure 3.1). Although defects can often be unwanted in the growth of a crystal, they can also provide the resulting material with advantageous properties that do not exist in the perfect crystal structure.[**78**] Therefore, a comprehensive understanding of these defects is valuable.

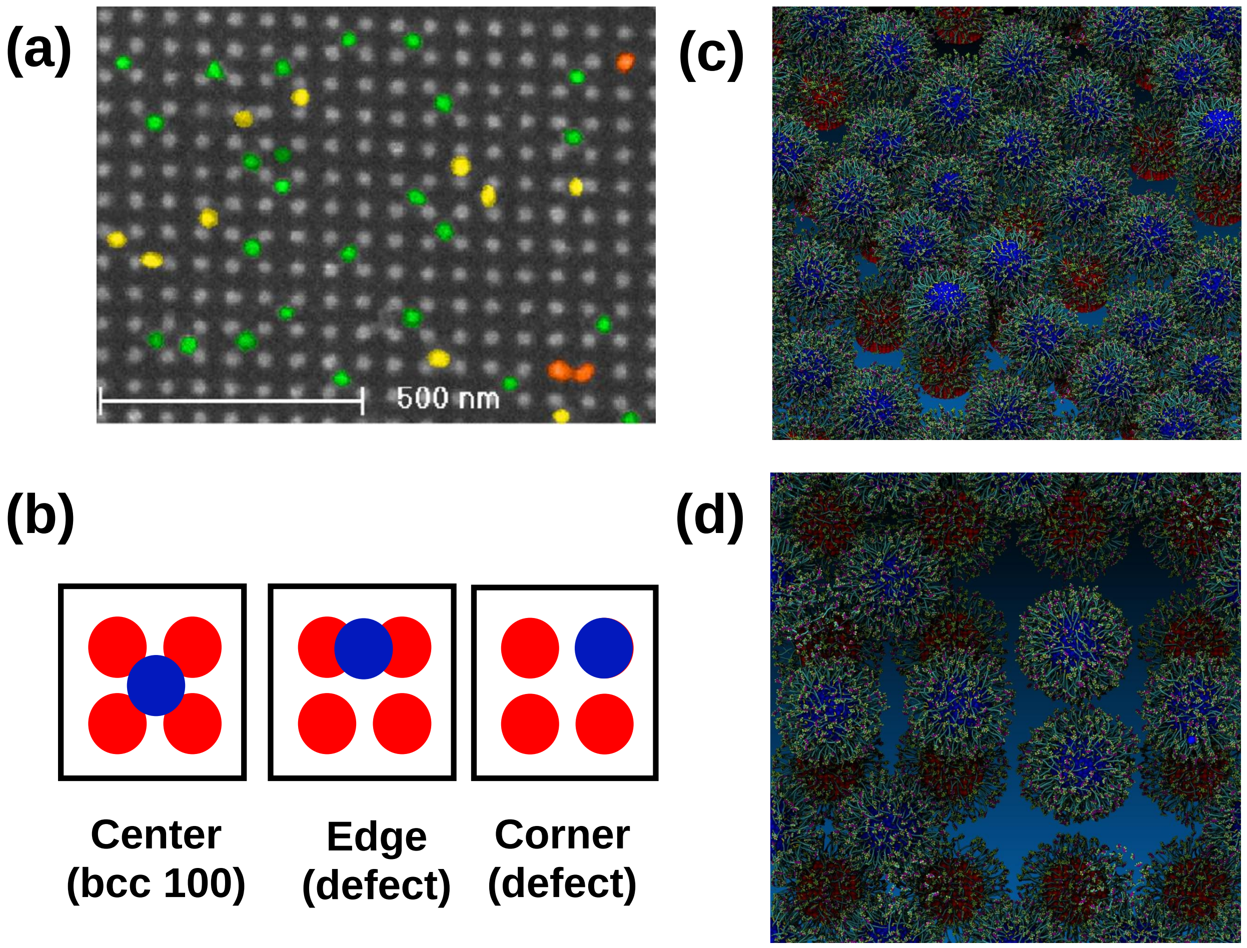


Figure 3.1. Epitaxial growth of DNA-coated nanoparticles on a BCC (100) template. (a) An epitaxial layer imaged *via* SEM. Nanoparticles are in false color to illustrate three types of attachments: center-attached in green, edge-attached in yellow, and corner-attached in orange. Adapted with permission from Hellstrom *et al.*[**6**] Copyright (2013) American Chemical Society. (b) Schematics of three types of attachments. Center-attached particles form a layer of BCC (100) plane, while edge- and corner-attached particles are defined as defects in the context of growing a BCC superlattice. (c) A snapshot of an epitaxial layer from a scale-accurate coarse grained MD simulation (side view). (d) A snapshot of an epitaxial layer from a scale-accurate coarse grained MD simulation (top view). It reproduces all three types of nanoparticle attachments found in experiments.

In this work, we use Monte Carlo (MC) simulations to systematically study the formation of defects in the epitaxial growth process. The simulation parameters can be computed from Molecular Dynamics (MD) simulations using the scale-accurate coarse grained model for epitaxial growth of DNA-coated nanoparticles on patterned substrates.[**79**] The MD models reproduce the experimental results very well, as shown in Figure 3.1(c, d). However, we cannot access large-scale physical phenomena by MD simulations with the explicit DNA chain model due to the computational cost. Here, we use a Monte Carlo model that enables the study of large scale systems. We demonstrate that defects in the epitaxial layer tend to aggregate in linear clusters that have either a horizontal or vertical orientation. These linear defects can undergo a liquid crystal phase transition below a critical temperature, which fundamentally changes the physical properties of the resulting metamaterial. Such precise control of defects could offer a pathway towards applications of 2D and 3D nanoparticle superlattices. Moreover, we will show how the physical properties of the resulting structure can be controlled by carefully engineering the defects.

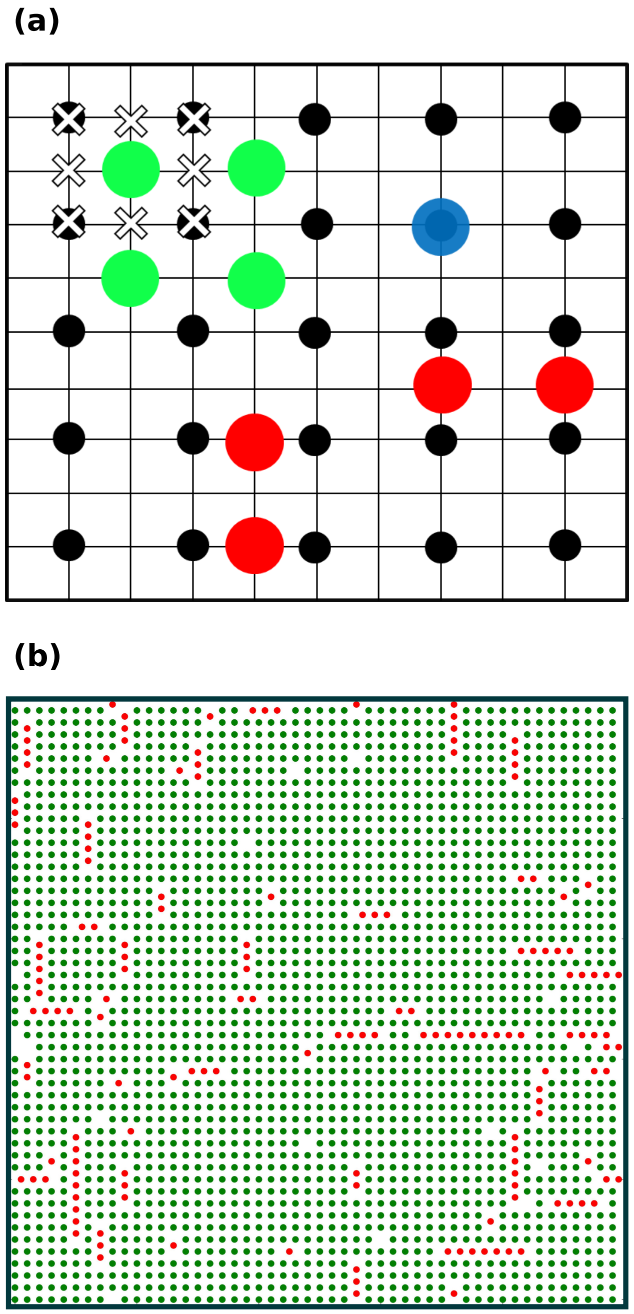


Figure 3.2. (a) A schematic diagram of a $10 \times 10$ lattice model. Black dots denote the BCC (100) patterned template with lattice parameter $a = 2$. Green, red, and blue dots denote center-, edge-, and corner-attached nanoparticles respectively. We show that no overlapping conditions arise due to excluded volume effects around the center-attached nanoparticle. Similar non-overlapping conditions exist for edge- and corner-attached nanoparticles. (b) A snapshot of an epitaxial layer at equilibrium for a $100 \times 100$ lattice.

## 3.2. Model

In this work, we focus on the epitaxial growth of body-centered cubic (BCC) superlattices along the (100) plane. In experiments, the template is fabricated by electron beam lithography to resemble the (100) surface, which yields a square lattice of nano-posts. Unlike the atomic lattice, DNA functionalized nanoparticles can attach to three different types of sites due to the shape effect of the template posts and the functionalization of coated DNA linkers.[**6**] As shown in Figure 3.2(a), nanoparticles can attach to center sites, edge sites or corner sites. Center-attached nanoparticles are located in the center of a unit cell and bounded by four template posts. Edge-attached nanoparticles are located in between two template posts, and therefore are on the edge of the unit cell. Corner-attached nanoparticles are located in the corner of a unit cell and are bounded by one template post. For the sake of clarity, three different types of attached nanoparticles are indicated in different colors.

Here, we adopt a modified 2D Ising model to study the adsorption of nanoparticles on a BCC (100) patterned template. Each lattice site in the Ising model corresponds to a possible binding site adsorbing a nanoparticle from solution. We denote the adsorption status of each site by $\sigma_{ij}$, which can take only two values, 0 and 1, representing unoccupied and occupied sites respectively. To account for all three adsorption patterns, we set the binding energy of each site $U_{ij}$, to the corresponding attachment type, which is $U_{\text{center}}$ for center sites, $U_{\text{edge}}$ for edge sites, or $U_{\text{corner}}$ for corner sites. The non-overlapping condition between adjacently attached nanoparticles imposes constraints on our model. In particular, each attached nanoparticle excludes the eight most neighboring sites from having a nanoparticle attached as well, as can be seen in Figure 3.2(a). Following the Monte Carlo method for the grand canonical ensemble, we invoke both adsorption/desorption and diffusion moves to ensure the system is able to equilibrate quickly.

In the grand canonical ensemble, the probability $p$ of finding the system in a certain microstate is given by:

$$p = \frac{1}{\mathcal{Z}} \exp(\beta(\mu M - H)), \tag{3.1}$$

where $\mathcal{Z}$ is the grand partition function which is a constant, $\beta = 1/kT$ is the inverse temperature, $\mu$ is the chemical potential, $M = M_{\text{center}} + M_{\text{edge}} + M_{\text{corner}}$ is the total number of attached particles, and $H = U_{\text{center}}M_{\text{center}} + U_{\text{edge}}M_{\text{edge}} + U_{\text{corner}}M_{\text{corner}}$ is the Hamiltonian of the system. Note that we can write $p$ in a convenient way for implementing simulations:

$$p = \frac{1}{\mathcal{Z}} \exp(-\beta(E_{\text{center}}M_{\text{center}} + E_{\text{edge}}M_{\text{edge}} + E_{\text{corner}}M_{\text{corner}})), \tag{3.2}$$

where $E_x = U_x - \mu$ ($x = \text{center}, \text{edge}, \text{corner}$), are the only parameters we control in the simulations. These parameters can be further mapped to experimental parameters with the help of a scale-accurate coarse grained model discussed in Pan *et al.*[**79**] For example, for a system with an ideal template, the ratio $\beta E_{\text{center}} : \beta E_{\text{edge}} : \beta E_{\text{corner}} = -4 : -2 : -1$. Ratios of binding energies can be altered by changing the shape and size of the templates. The absolute values of binding energies can be controlled by adjusting DNA grafting density and salt concentration.

## 3.3. Results and Discussion

We first start with a general simulation system where $(\beta E_{\text{center}}, \beta E_{\text{corner}}, \beta E_{\text{edge}}) = (-5, -4.5, -4.5)$. As shown in Figure 3.2(b), most of the attached particles are in center sites, which form a BCC (100) monolayer. Edge-attached defects are visible, while corner-attached defects are rarely seen. We observe that edge defects prefer aggregating to form linear clusters in order to lower the energy of the system. This is because when two linear defect clusters combine, they release a vacancy for a center-attached particle to bind. However, this process reduces the entropy of defect clusters, which corresponds to the number of configurations. Let us consider a simple example of two isolated edge-attached particles in one dimension. When two particles combine to form a two-particle cluster, the entropy decreases because the number of configurations for a two-particle cluster is smaller than that for two isolated particles. As a result of the energy-entropy balance, the system thermalizes to configurations of defect clusters with a certain distribution at equilibrium. As Figure 3.2(a) shows, each edge defect can only belong to one defect cluster, in either horizontal or vertical directions. Due to the excluded volume effect of each edge defect, a horizontal cluster cannot combine with a vertical cluster. Note that both horizontal and vertical clusters occur in the epitaxial layer, as shown in Figure 3.2(b). There is no obvious orientation preference in this case.

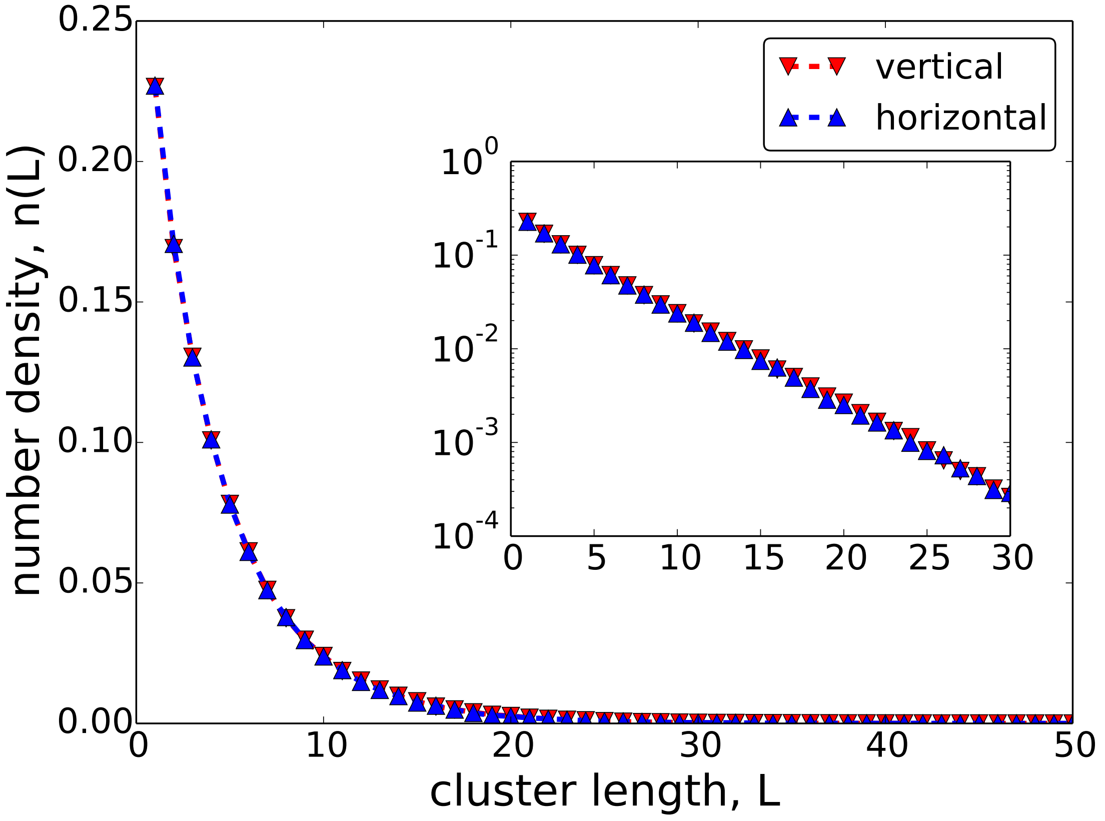


Figure 3.3. The length distribution of edge-defect clusters, plotted on a linear scale, as well as logarithmic, showing the exponential decrease of occurrence of larger clusters.

To analyze the distribution of these linear defects, we calculate the number density of defect clusters of length $L$. The length of a defect cluster is defined as the number of adjacent edge-attached particles. We count the defect clusters in both vertical and horizontal orientations separately at each equilibrium state, and average over enough MC steps. We plot the length distribution of defect clusters in each orientation, and then fit the resulting function to an exponential. As shown in Figure 3.3, both vertical and horizontal defects have a nearly same exponential distribution. The inset plots the length distributions on a log-linear scale, showing an almost perfect exponential decaying distribution. Interestingly, we observe a similar distribution that arises in solutions of worm-like surfactant micelles. We can describe our system with an analytical model which is adapted from that of the worm-like cylindrical micelle system.[**80**] The free energy may be written as,

$$F \sim kT \sum_L c(L) \log c(L) + \sum_L c(L) E_{\text{scission}} + F_{\text{other}}, \tag{3.3}$$

where $c(L)$ is the number density of edge defect clusters of length $L$; $E_{\text{scission}}$ is the scission energy of a defect cluster, which is the energy required to break into two pieces; $F_{\text{other}}$ is the free energy associated with the corner defects and center-attached particles. For simplicity, we consider here the free energy associated with the edge defects,

$$F_{\text{edge}} \sim kT \sum_L c(L) \log c(L) + \sum_L c(L) E_{\text{scission}}. \tag{3.4}$$

The first term describes the entropy effect, while the second describes the energy effect. At a fixed volume fraction of edge defects,

$$\Phi_{\text{edge}} = \sum_L c(L) L, \tag{3.5}$$

minimizing $F_{\text{edge}}$ with respect to $c(L)$ yields an exponential distribution,

$$c(L) \propto \exp(-L/\overline{L}), \tag{3.6}$$

where $\overline{L} \sim \sqrt{\Phi_{\text{edge}} \exp\left(\beta E_{\text{scission}}\right)}$. These similarities will be discussed further in the section Theoretical Analysis.

### 3.3.1. Phase transition for the special case: $E_{\text{center}} = E_{\text{edge}}$

The majority of defects are edge-attached defects which prefer to form linear clusters. First, we consider the special case $E_{\text{center}} = E_{\text{edge}} = E_0 < 0$. Furthermore, we set $E_{\text{corner}} = 0$ from now on, which allows us to focus on the properties of the edge defects only. We vary the parameter $\beta E_{\text{center}}$ from -1 to -8 with a step of 0.2. We explore this region using simulations on a $400 \times 400$ lattice. For the sake of clarity, we show snapshots of equilibrium states in a smaller $100 \times 100$ square lattice (Figure 3.4).

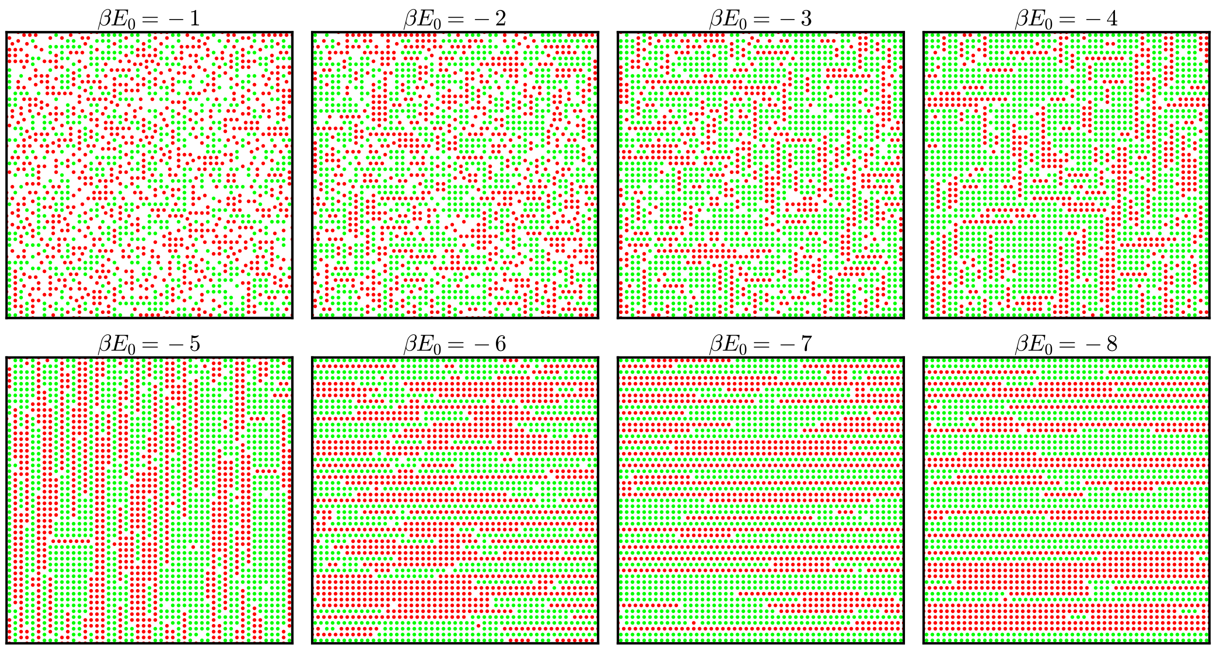


Figure 3.4. Eight snapshots of an epitaxial layer on a $100 \times 100$ square lattice for $\beta E_{\text{center}}$ from -1 to -8. Center-attached particles are in green while edge-attached particles are in red.

We calculate the equilibrium site fractions of center-attached particles and edge-attached particles. The site fraction is defined as the number of attached particles divided by the total number of unit cells in the system:

$$\Phi_i = M_i/N, \tag{3.7}$$

where $N$ is the total number of unit cells in the system, *i.e.* for the $400 \times 400$ square lattice, $N = 200 \times 200 = 40000$. We also define the average length of defects:

$$\overline{L} = M_{\text{edge}}/n_{\text{edge}}, \tag{3.8}$$

where $n_{\text{edge}}$ is the total number of linear clusters formed by edge defects. To elucidate the competition between entropy and energy driving the formation of clusters, we plot the site fractions and the average defect length as a function of the inverse temperature ($\beta E_0$), as shown in Figure 3.5.

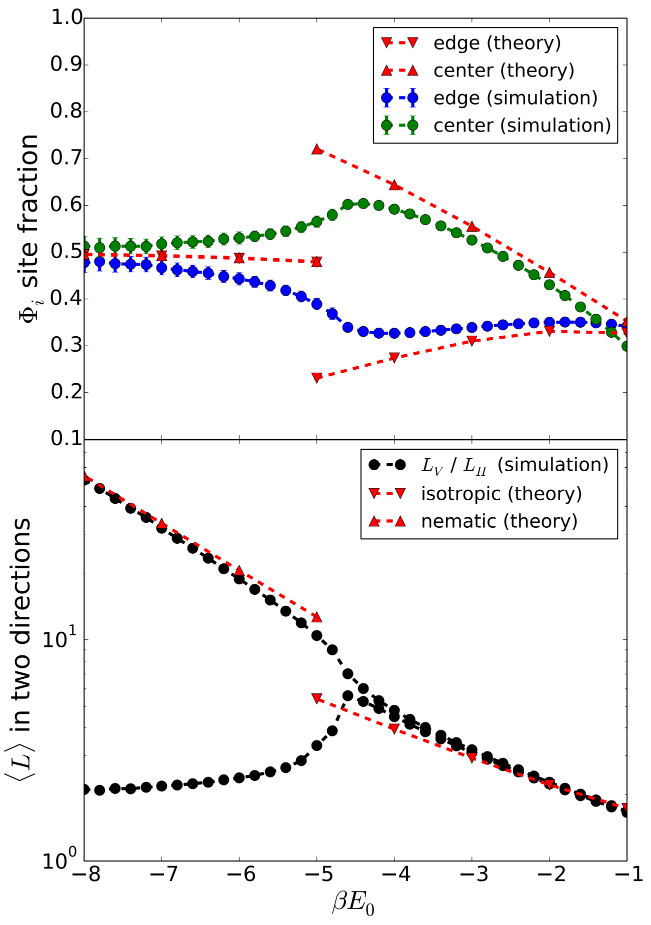


Figure 3.5. The site fractions of center- and edge-attached particles (top) and the average defect length (bottom) as a function of the inverse temperature ($\beta E_0$) for the special case $E_{\text{center}} = E_{\text{edge}} = E_0$. Both simulation results and theoretical results are plotted.

In the high temperature regions, as $\beta E_0$ is lowered from -1 to -4, we observe a nearly constant concentration of the edge defects around 0.3. However, the center-site fraction nearly doubles from 0.3 to 0.6. Meanwhile, the average length of defect clusters also grows. This can be explained by the cooling induced aggregation of defect clusters. As temperature decreases, the energy part of the free energy becomes more important than the entropy part. When two defect clusters aggregate, they release a vacancy which can be occupied by a center-attached particle. The energy is lowered while the entropy is reduced. The increasing aggregation between defect clusters causes the system to reach a balance between the gain in energy and the loss in entropy.

As the temperature decreases further, defect clusters will grow longer to minimize the free energy of the system. However, the system will reach a critical state where growth of defect clusters in one orientation is blocked by that in the other orientation, which means that defect cluster cannot grow continuously in a disordered state where both horizontal and vertical clusters are evenly present. At $\beta E_0 \simeq -4.5$, the plots of site fractions and average defect length show a distinct change, implying a phase transition in this region. This is confirmed by the emergence of an ordered phase for $-5 > \beta E_0 > -8$, as can be observed in Figure 3.4. This phase, in which the edge defect clusters predominantly lie in one preferred orientation, enables the average cluster length to keep increasing, while the cluster length in the other orientation quickly decreases.

At extremely low temperatures, all defect clusters lie in one direction (see Figure 3.4). The complete disappearance of either the horizontal or the vertical defect clusters introduces a symmetry in the system, as the center- and the preferred edge-attached sites become effectively equivalent. This results in equal fractions of both, which can be observed in Figure 3.5 at $\beta E_0 = -8$. The average length of the defect clusters reaches the order of the lattice dimensions.

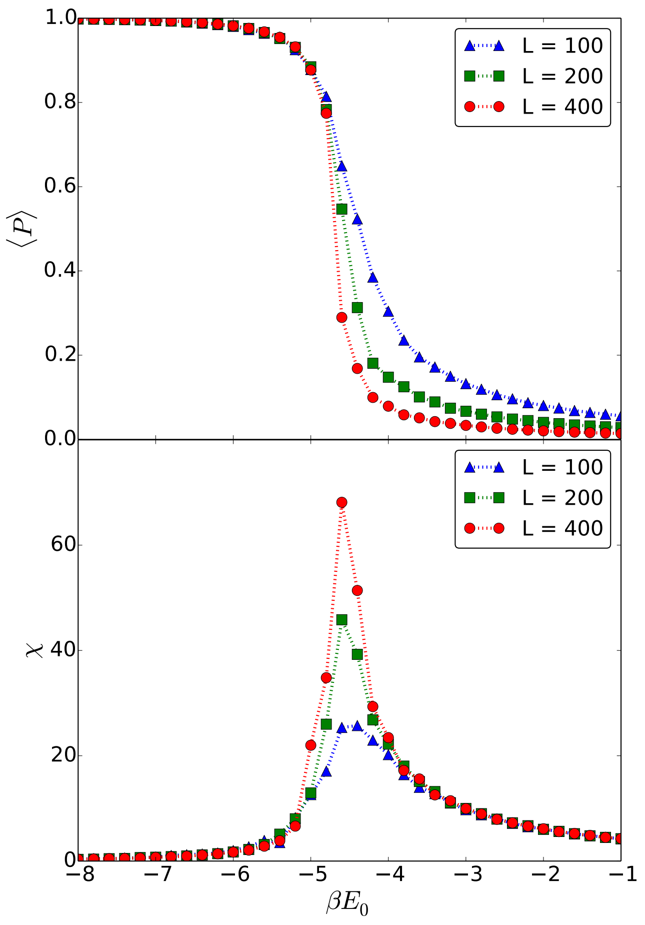


Figure 3.6. The orientational order parameter $\langle P \rangle$ (top) and the orientational susceptibility $\chi$ (bottom) as a function of the inverse temperature ($\beta E_0$) with different lattice sizes for the special case $E_{\text{center}} = E_{\text{edge}} = E_0$.

The orientational disordered-ordered phase transition can be described by introducing an order parameter

$$P = \left| \frac{\Phi_V - \Phi_H}{\Phi_V + \Phi_H} \right|, \tag{3.9}$$

where $\Phi_V$ and $\Phi_H$ are site fractions of vertical and horizontal clusters respectively.

Within the disordered phase, $\Phi_V$ and $\Phi_H$ are nearly the same, and $\langle P \rangle$ is approximately zero for large systems. Within the ordered phase, defect clusters for one of the orientations dominate such that the order parameter moves towards one. The order parameter describes the rapid phase transition at $\beta E_0 \simeq -4.5$ when $\langle P \rangle$ jumps from zero to one.

We will, therefore, define the disordered and the ordered phases to be characterized by $\langle P \rangle < \frac{1}{2}$ and $\langle P \rangle > \frac{1}{2}$ respectively. Analogous to the magnetic susceptibility in the Ising model, we define the orientational susceptibility as $\chi \equiv \beta N(\langle P^2 \rangle - \langle P \rangle^2)$, which shows expected critically-divergent behavior at the critical point that is located at $\beta E_0 \simeq -4.5$, analogous to what is observed for magnetic susceptibility in the Ising model.

As expected, the critical behavior becomes more obvious as the lattice size increases, as demonstrated by Figure 3.6. Our results indicate that the phase transition from $\langle P \rangle = 0$ at high temperatures to nonzero $\langle P \rangle$ at low temperatures will be infinitely sharp for experimental systems.

In contrast to magnetic systems, there is no obvious external field that could induce symmetry breaking close to the critical temperature. We speculate, however, that alternating electric fields can be used to induce dipole forces between the nanoparticles, yielding symmetry breaking between horizontally and vertically aligned defect clusters by altering the effective nearest-neighbor interaction energies. This near-critical susceptibility provides a possible mechanism to engineer materials with high control of the defect cluster orientation.

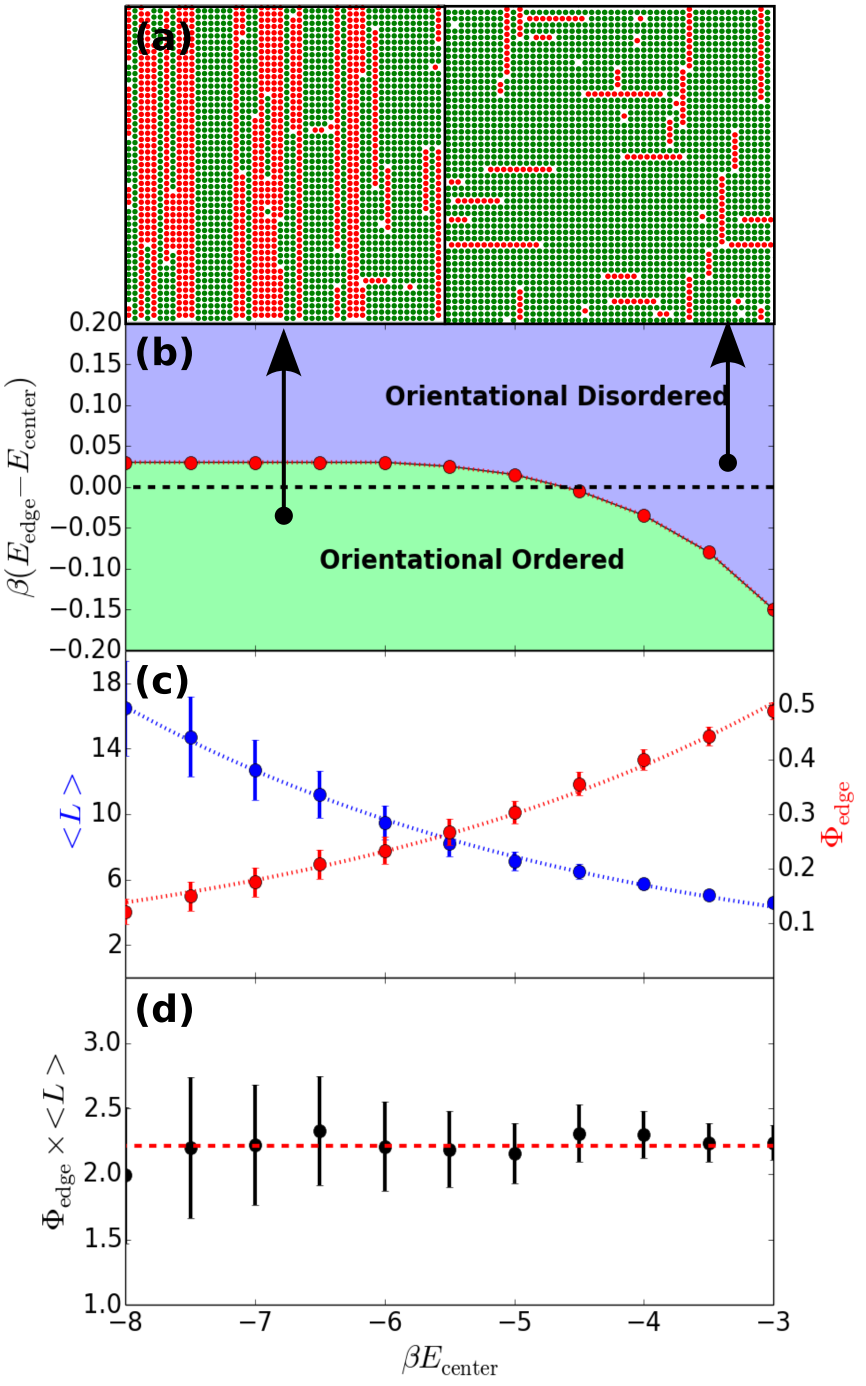


Figure 3.7. (a) Snapshots of the equilibrated systems in two distinct phases. (b) Phase diagram of the system in $(\beta E_{\mathrm{center}}, \beta E_{\mathrm{edge}})$ parameter space. (c) The average defect length $\langle L \rangle$ (left) and edge defect fraction $\Phi_{\mathrm{edge}}$ (right) at the phase boundaries as a function of $\beta E_{\mathrm{center}}$. (d) The product of $\langle L \rangle$ and $\Phi_{\mathrm{edge}}$ as a function of $\beta E_{\mathrm{center}}$ as well as a linear fitting line of the data points.

### 3.3.2. Phase diagram for the general case: $E_{\text{center}} \neq E_{\text{edge}}$

With some understanding of the phase transition in this special case, we now investigate the phase transition in the whole parameter phase of $(\beta E_{\text{center}}, \beta E_{\text{edge}})$. For each $\beta E_{\text{center}}$ from -8 to -3 in a step of 0.5, we explore a full range of $\beta E_{\text{edge}}$ to explore the phase transition and determine phase boundaries according to the value of $\langle P \rangle$. For the sake of clarity, the phase diagram is plotted in the parameter space of $\beta E_{\text{center}}$ and $\beta(E_{\text{edge}} - E_{\text{center}})$ as shown in Figure 3.7(b). A dashed line is plotted in the phase diagram, which represents the special case: $E_{\text{center}} = E_{\text{edge}}$. It also reveals its critical point at $\beta E_{\text{center}} \simeq -4.5$.

From the snapshots (Figure 3.4) and the simulation movies (see Supporting Information) of the system, we observe that the phase transition of our system is analogous to the isotropic-nematic phase transition in liquid crystals.

Let us consider a simplified system with a uniform length distribution of defect clusters in which the length is set as the average defect length $\overline{L}$. The excluded volume (strictly, area in 2D) between two defect clusters in different orientations is $\overline{L}^2$, while that between two aligned defect clusters is negligible. In the isotropic phase, the total excluded volume is $V_{\text{ex}} = \overline{L}^2(n/2)$, where $n$ is the number of defect clusters. In the limit of densely packed defect clusters, the total excluded volume $V_{\text{ex}}$ cannot exceed the volume of the system $V$. At the phase transition, $V_{\text{ex}} = \overline{L}^2(n/2) = V$. Substituting the defect fraction $\phi = n\overline{L}/V$ into the previous equation, we obtain the phase transition condition $\phi \times \overline{L} \equiv 2$, which is in qualitative agreement with the simulation results shown in Figure 3.7(d). This condition holds for all phase boundaries in the parameter space.

## 3.4. Theoretical Analysis

To further validate the simulation results, we perform a theoretical analysis based on mean-field theory. To facilitate a mean-field analysis, we regard the system as a collection of unit cells.

Each of these cells contains a center binding site as well as the neighboring four edge binding sites (left/right/up/down), which are shared with the neighboring unit cells. For simplicity, we do not include corner sites here. Note that the number of unit cells matches the number of lattice particles $N$. As Figure 3.8 shows, the number of occupied sites in a cell can be either one or two, and can

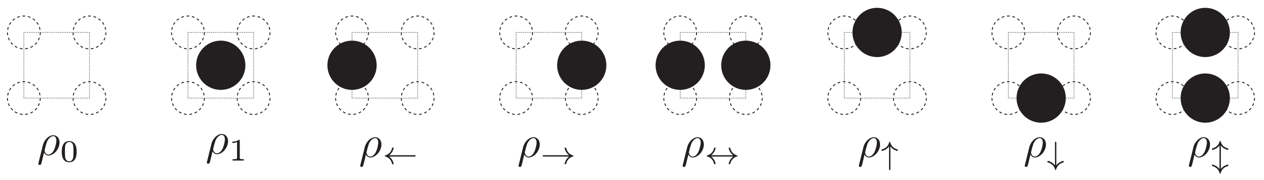


Figure 3.8. The various particle configuration within a unit cell. The unit cells are spanned by four template particles and are indicated by squares. The fraction of cells are denoted by the $\rho_\alpha$, each representing a different particle configuration.

be a center site, horizontal defects, or vertical defects. We denote the configuration of the unit cells by the position of the occupied site in the cell. This can be on the left edge ($\leftarrow$) and/or on the right edge ($\rightarrow$) of the unit cell for horizontal defects, and upper edge ($\uparrow$) and/or lower edge ($\downarrow$) of the unit cell for vertical defects. Other possible cell configurations are either empty cells and cells that have an occupied center site, denoted by $0$ and $1$ respectively. In total, each unit cell can have eight different configurations. The corresponding fractions of the various cell configurations should add up to unity,

$$\rho_\leftarrow + \rho_\rightarrow + \rho_\leftrightarrow + \rho_\uparrow + \rho_\downarrow + \rho_\updownarrow + \rho_0 + \rho_1 = 1. \tag{3.10}$$

The shared edge sites between cells effectively lead to a strong interaction between neighboring cells because not all combinations of cell configurations can be adjacent to each other. The fractions $\rho_i$ of the various cell configurations are determined by the horizontal and vertical edge defect densities, denoted by $\phi_H = M_H/N$ and $\phi_V = M_V/N$ respectively, and also by the average defect line lengths $\overline{L}_H$ and $\overline{L}_V$. A cell with a single occupied edge site corresponds to an end-point of a defect line. Therefore, the fraction of such cells is related to the number of defects as well as the average line length $\overline{L}_H$ *via* $\rho_\leftarrow = \rho_\rightarrow = \phi_H/\overline{L}_H$ and $\rho_\uparrow = \rho_\downarrow = \phi_V/\overline{L}_V$ for horizontal and vertical defects respectively. The fraction of unit cells with two occupied edge sites equals the number of edge defects that are not the end point (either left or right) of a defect line, that is, $\rho_\leftrightarrow = \phi_H(1 - 1/\overline{L}_H)$ and $\rho_\updownarrow = \phi_V(1 - 1/\overline{L}_V)$, for horizontal and vertical defects respectively.

### 3.4.1. Length distribution of defect clusters

We now assume that the distribution of the various cells is random, yet we must keep in mind that cells share edge sites and, therefore, put restrictions on their direct neighbors. For example, a cell with two horizontal defects ($\leftrightarrow$) cannot have an empty cell on the left or the right. Its right neighbor can either have one ($\leftarrow$), or two ($\leftrightarrow$) horizontal edge defects. A defect line of length $n$ spans a row of $n+1$ cells from which only the first and the last cell have a single defect. Given a cell containing a starting point of a line ($\rightarrow$), the probability that the line does not terminate at its right neighboring cell is that of finding not one but two edge defects in this neighboring cell, that is, $\rho_\leftrightarrow/(\rho_\leftrightarrow + \rho_\leftarrow)$. By applying this argument repetitively, we find that the mean-field defect-line length distribution for horizontal defect lines of length $n$ is given by

$$f_{H,n} = \frac{p_\leftrightarrow^{n-1} p_\leftarrow}{(p_\leftrightarrow + p_\leftarrow)^n} = (1/\overline{L}_H)(1 - 1/\overline{L}_H)^{n-1}. \tag{3.11}$$

The derivation for vertical defect lines is analogous.

### 3.4.2. Average length of defect clusters

The length distribution described in Eq. (3.11) does not specify the average length of the defect lines $\overline{L}_H$ itself. For that we focus on scission/fusion events, in which one defect line splits up into two smaller ones and vice versa.

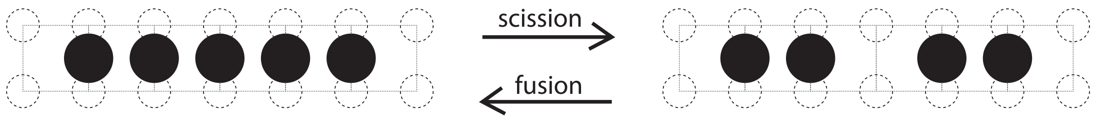


Figure 3.9. A horizontal defect line can be split into two smaller defect lines by removing one internal particle. We refer to such events and their reverse as scission and fusion respectively.

In our lattice model, scission of defect lines yields the removal of one of the internal edge-defect particles from a defect line, as Figure 3.9 illustrates. Again we focus on horizontal defects, although the derivation for vertical defect lines is analogous. An occupied edge site that is available for scission must have occupied neighboring edges on either side. The edge defect is therefore part of two horizontally neighboring cells which both have two horizontal defects and therefore form three defects in a row (triplet defects). The probability of finding two occupied horizontal edges sites in the first cell is $\rho_{\leftrightarrow}$ by definition. The configurations of the second cell, once the configuration of the first cell is known, is either one or two horizontal defects. Therefore, the probability for the second cell to have two horizontal defects as well is $\frac{\rho_{\leftrightarrow}}{\rho_{\leftrightarrow}+\rho_{\leftarrow}}$, yielding a probability of finding an edge site that is available for scission $p_{3h} \simeq \rho_{\leftrightarrow} \times \frac{\rho_{\leftrightarrow}}{\rho_{\leftrightarrow}+\rho_{\leftarrow}}$. On the other hand, the occurrence of scissions, which is a vacant edge site sandwiched between two occupied edge sites on either site, is $p_{sh} \simeq \rho_{\leftarrow} \times \frac{\rho_{\rightarrow}}{1-\rho_{\leftrightarrow}-\rho_{\leftarrow}}$. Here we use the probability that the first cell has an occupied defect site on its left edge, and calculate the probability that the cell on the right has a defect on its right edge. This probability can be calculated by noticing that the latter cell is allowed to have any configuration except for those that have an occupied defect site on their left edge. Triplet defects can be converted into scissions (and back) by removing (inserting) the middle edge defect particle. We therefore assert detailed balance, $p_{3h}/p_{sh} = e^{-\beta E_{\mathrm{edge}}}$, resulting in a mean-field expression for the value for the average cluster length,

$$\overline{L}_H = 1 + \sqrt{\frac{\phi_H e^{-\beta E_{\mathrm{edge}}}}{1-\phi_H}}. \tag{3.12}$$

Note that this result is independent on the number of vertical defects, which, in turn, obey a similar relation,

(3.13) $$\overline{L}_V = 1 + \sqrt{\frac{\phi_V e^{-\beta E_{\text{edge}}}}{1-\phi_V}}.$$

Both the relations for the cluster-size distribution (Eq. (3.11)), as well as the average defect-line length, are in good agreement with the theory on worm-like micelles.[**80**] For these systems, the micelle length was both experimentally and theoretically found to scale as $L \sim \sqrt{\phi \exp\left(\beta E_{\text{scission}}\right)}$, where $\phi$ is the volume fraction of micelles, and $E_{\text{scission}}$ is the scission energy that is the required energy to split a micelle into two smaller copies. In our system, the energy needed in splitting a cluster is $-E_{\text{edge}}$.

### 3.4.3. Orientational disordered phase

In the disordered phase, where $\phi_H = \phi_V$ and $\overline{L} \equiv \overline{L}_H = \overline{L}_V$, it is possible to calculate directly the defect density $\phi = \phi_H + \phi_V$. Let us consider two horizontally neighboring cells. These cells span a row of three binding sites for horizontal defects, from which the middle one is shared between the cells. We calculate the probability that the first two binding sites of this row are occupied and the third one is empty. This yields finding the end of a defect line containing multiple defects. The probability of finding two horizontal defects in the left cell, and only one horizontal defect in the right cell is $p_\alpha \simeq \rho_{\leftrightarrow} \times \frac{\rho_{\leftarrow}}{\rho_{\leftrightarrow}+\rho_{\leftarrow}}$. Likewise, the probability of finding a configuration in which the middle binding site is empty is $p_\beta \simeq \rho_{\leftarrow} \times \frac{\rho_0}{1-\rho_{\leftrightarrow}-\rho_{\leftarrow}}$, corresponding to removing the last defect from the line. Using detailed balance between both configurations $\alpha$ and $\beta$ we get $\rho_0 e^{-\beta E_{\text{edge}}}\overline{L} = (\overline{L}-1)(1-\phi_H)$. In this equation, the average length $\overline{L}$ can be related to the defect density $\phi$ *via* Eq. (3.12). The density of empty cells $\rho_0$ can be found by noticing $\rho_1 = \rho_0 e^{-\beta E_{\text{center}}}$, which in combination with Eq. (3.10) yields $\rho_0 = (1-\phi(1+1/\overline{L}))/(1+e^{-\beta E_{\text{center}}})$. This yields an expression for $\phi$, which can be solved numerically. For the special case $E_{\text{edge}} = E_{\text{center}} = E_0$, the solution is found by the first real root of

$$-8h + (36h + 4h^2)\phi - (44h + 4h^2)\phi^2 + (1 + 14h + 2h^2)\phi^3 = 0, \tag{3.14}$$

where $h = \exp(\beta E_0)$. With the relation between $\phi$ and $h$, we can then obtain the relation between $\rho_1$ and $h$. The plot of edge and center site fractions predicted by mean-field theory is shown in Figure 3.5, which is in good agreement with the simulation results in the corresponding regimes. $\overline{L}$ can also be calculated numerically in this phase, which is also plotted in Figure 3.5. It also shows good agreement with the simulation results for $\beta E_{\text{center}} > -4.5$.

### 3.4.4. Orientational ordered phase

In this orientational ordered phase, for simplicity we consider the thermodynamic limit where $\phi_H = \phi$ and $\phi_V = 0$. Similarly, we can get an expression for $\phi$ which can be solved analytically in the same way. We obtain the expression

$$\phi = \frac{4 + h + \sqrt{h(4+h)}}{2(4+h)}, \tag{3.15}$$

where $h = \exp(\beta E_0)$. We then calculate the site fraction of center-attached particles and average length of defect lines in this phase. We show the numerical results in Figure 3.5, which again has good agreement with the simulation results in the ordered regimes.

## 3.5. Conclusions

We have used Monte Carlo simulations to study the defect structures in the epitaxial growth of DNA functionalized nanoparticles on a patterned substrate. By investigating the influence of tailored parameters including the three binding energies of different attachment sites, we observe that structural defects within epitaxial layers are correlated and aggregate to form one-dimensional clusters with an exponential length distribution. This distribution resembles that found in worm-like micelles.[**80**] These linear defects undergo an orientational disordered-ordered phase transition, which is analogous to the spontaneous magnetization transition in a 2D ferromagnetic Ising model. Many features between these two models are similar, including the critical behavior of the order parameter and susceptibility, as well as finite size effects.

These two phases of structural defects also resemble the isotropic and nematic phases in 2D liquid crystals. We show that the ordering transition of defect lines is entropy-driven due to the excluded volume effects of defect lines in perpendicular orientations. We deduce a phase boundary condition that is similar to that in a hard-rod view of liquid crystals. By further theoretical analysis, we can calculate quantities including site fraction, average defect length, and defect length distribution. These results are in good agreement with the simulation results.

Real crystals are never perfect; there are always defects. However, the possibility of making imperfect materials enables scientists to tailor material properties into the diverse combinations that modern devices require. By studying defect formation in the epitaxial layer of DNA functionalized nanoparticles and how to engineer these defects into different structures of great interest, we hope to open avenues for designing and fabricating nanomaterials with controllable defect structures for real applications.

## 3.6. Methods

### 3.6.1. Monte Carlo simulation for grand canonical ensemble

The probability associated with a microstate in the grand canonical ensemble is

$$P = \frac{1}{\mathcal{Z}} \exp(-\beta(E_{\text{center}}M_{\text{center}} + E_{\text{edge}}M_{\text{edge}} + E_{\text{corner}}M_{\text{corner}}))$$

In our model we choose $E_x \leq 0$ ($x = \mathrm{center}, \mathrm{edge}, \mathrm{corner}$) because the particles have a preference to bind to the lattice. Adsorption and desorption are implemented by choosing a random site $\sigma_{ij}$, and if we find the site to be unoccupied we set $\sigma_{ij} \to 1$ only if the final configuration satisfies non-overlapping conditions. Oppositely, if we find the site is occupied, we update it to $\sigma_{ij} \to 0$ with probability $P_{\text{desorb}} = \exp(+E_x)$, where $x$ represents the site type. We also include diffusion moves. For that we pick a random occupied site as well as a random diffusion direction (up/down/left/right). If the final configuration satisfies the non-overlapping condition we perform this move with acceptance ratio $P_{\text{diffuse}} = \min(1, \exp(+E_F - E_I))$ with $F$ and $I$ the site type of the final and initial state respectively.

CHAPTER 4

# Multi-Scale Simulations of DNA-Directed Nanoparticle Assembly into Icosahedral Nanocages

Nowadays, researchers are paying more and more attention to anisotropic nanostructures due to their rich assembling behaviors and novel properties owing to their unusual shape and mutual interaction. A remarkable variety of anisotropic nanoparticles have been synthesized in the past few years. The challenge is how to assemble these anisotropic nanoparticles into required target structures to obtain desired properties. The ability to assemble particles into ordered structures with great complexity and sophistication depends on the understanding of mechanism of assembly and controlling of the interparticle interaction. Computer simulations can play an important role in the understanding of assembly process at different length scales, and eventually predict the phase behavior of nanoparticle assemblies. We conducted multi-scale molecular dynamics simulations and Monte Carlo simulations to demonstrate that quasicrystalline five-fold symmetry is formed in DNA-directed assembly of tetrahedron nanoparticles. Further, icosahedral cages can be formed by truncated tetrahedron nanoparticles with specific preconfigurations. We expect and the results will help understand self assembly of spherical viruses into capsids with icosahedral symmetry.

## 4.1. Introduction

DNA-directed assembly is a versatile strategy to precisely control a large variety of nanoparticle building blocks to be assembled into desired structures which produce novel functions for the nanomaterials. For spherical nanoparticles, different kinds of well ordered superlattice have been achieved via this approach and with the help of computational modeling[**1, 2, 51, 35, 52, 53**].

For non-spherical nanoparticles such as cubes and other regular polyhedrons[**81, 82, 83**], single crystals with corresponding equilibrium Wulff shapes are synthesized[**3, 84**]. The assembly of tetrahedron nanoparticles with DNA remains unknown, which has potentials of forming quasicrystals[**85, 86, 87, 88**]. Although there are many computational studies about packing hard tetrahedron particles, assembly of tetrahedron nanoparticles with DNA is a different story. The former is driven by entropic force only, which requires external pressure. The latter is driven by minimizing the free energy of the system, which includes both the entropy and enthalpy. Therefore, a computational study with reliable scale-accurate models is necessary and will guide the experimentalists to synthesize ordered superlattices with specific experimental conditions which can be predicted by simulations. Furthermore, for truncated tetrahedron building blocks, an icosahedral nanocage is possible to be formed by DNA-guided assembling, which is not only important in applications but also in understanding the biological self-assembling processes of icosahedral protein nanocages.

We first applied the detailed-accurate coarse grained model to the system of regular tetrahedron nanoparticle building blocks. Although it is computationally expensive, we can get a good understanding of the interaction between nanoparticles within clusters. We observed the five-fold symmetry in nanoparticle clusters at temperatures right below melting point in the NVT ensemble. To further test the stability of the clusters, we then implemented the simulation in the NPT ensemble. We demonstrated that the five-fold symmetry is not local but expand to all the plane in a specific direction. Due to the limitation of this model, we can only study a relatively small system. To convince that boundary effects do not play an important role in the assembly process and characterize the assemblies at a larger length scale, we then conducted the simulation with a more coarse-grained model with implicit DNA linkers. We used the effective potential between two building blocks calculated from our scale-accurate models to represent the soft potential between surface pseudo-particles from different building blocks in the new model, by which we included the DNA hybridization to the model and in the meantime saved time computing the explicit DNA interactions. Two results from two models are in consistent with each other, which means we can precisely control the assembly of tetrahedron nanoparticles to form patterns existed only in quasicrystals. Furthermore, we applied the simulations to truncated tetrahedron systems. We initialized the system with twenty DNA-coated truncated tetrahedron nanoparticles. Each building block has specific position and orientation which form an icosahedral nanocage with a designed gap. We tuned the gap between neighbouring building blocks and truncation ratios to test the stability of the assemblies.

## 4.2. Models and Methods

### 4.2.1. Hard-particle model

In the hard-particle model as in Figure 4.1, all the building blocks are simply identical regular tetrahedrons. They interact with each other under non-overlapping condition. The potential between particle i and particle j,

$$V(i,j) = \begin{cases} +\infty, & \text{if overlap} \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

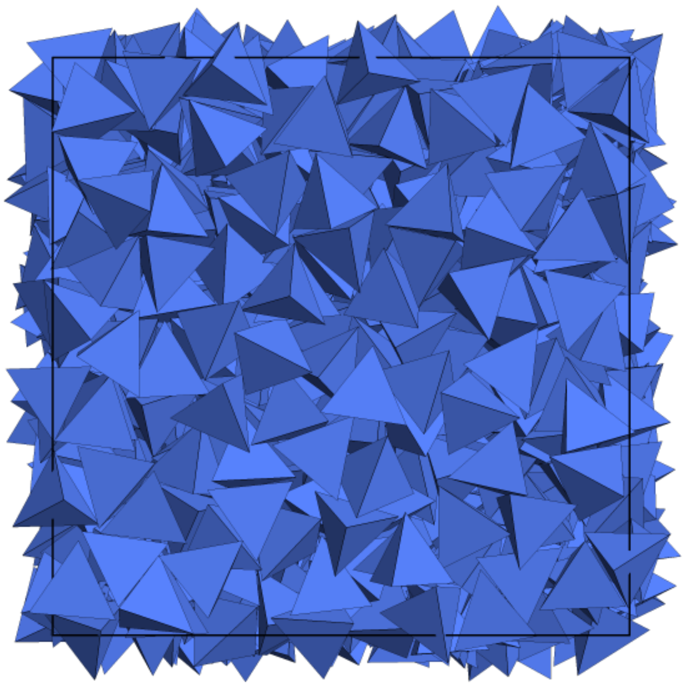


Figure 4.1. Hard particle model using Monte Carlo simulation. The system contains 1000 regular tetrahedron particles.

We performed hard-particle Monte Carlo (MC) simulations with HPMC[**89**], a plugin to the HOOMD-blue simulation package[**59**], in the NPT thermodynamic ensemble to study the behavior of self assembly of hard tetrahedrons. The polyhedra are modeled as perfectly faceted shapes of unit length, with sharp vertices and edges. Simulations were carried out in cubic boxes with periodic boundary conditions.

### 4.2.2. Scale-accurate coarse-grained model

As shown in Figure 4.2, each DNA-NP building block is modeled as a rigid body NP core with bead-spring DNA linkers. A NP core contains a fixed number of surface beads which mimic the size and shape of NP. Pseudo-particles are created uniformly on the surface of the NP core. Self complementary DNA linkers are attached to part of the surface particles randomly with a desired grafting density and finally forms a core-shell DNA-nanoparticle building block. At the end of each DNA linker, there is a "sticky end" region used to model the directional bonding between complementary bases (A-T, C-G). Here we apply self-complementary sequence "AATT" to our DNA models.

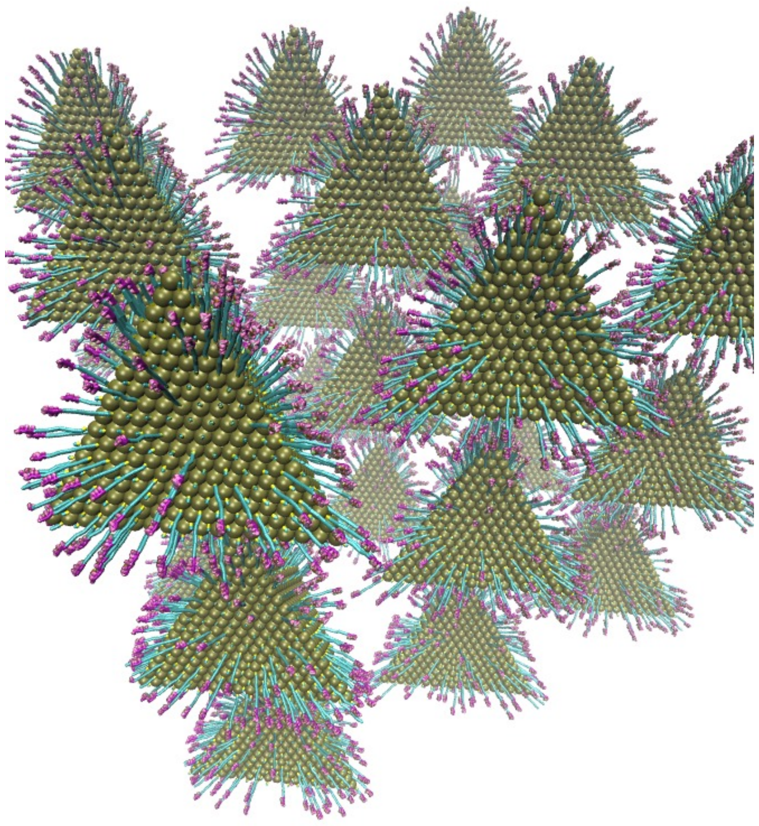


Figure 4.2. Snapshot of the initial configuration for a DNA-coated tetrahedron system using a scale accurate coarse grained model. Nanoparticle are represented by surface particles to capture the size and shape of the core; DNA linkers are modeled using bead-bond model as polymers. Specially the DNA linkers have regions which mimic the "sticky end" of single strand DNA. The system contains 27 building blocks.

The interaction between the sticky-end beads was modeled with a shifted Lennard-Jones potential,

$$V(r) = \begin{cases} 4\epsilon[(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^{6}] - V(r_c), & \text{for } r <= r_c \\ 0, & \text{for } r > r_c, \end{cases} \tag{4.2}$$

where $r$ is the distance between two "sticky end" beads and $r_c$ is the cutoff distance. Interaction beetween any other pairs of beads was modeled with the Weeks-Chandler-Andersen (WCA) soft-core repulsive potential[**90**]. It is simply the shifted Lennard-Johns potential with $r_c = 2^{1/6}\sigma$. Harmonic bond and angle potentials are also used to capture the flexibility of DNA linkers.

### 4.2.3. Simplified colloidal model

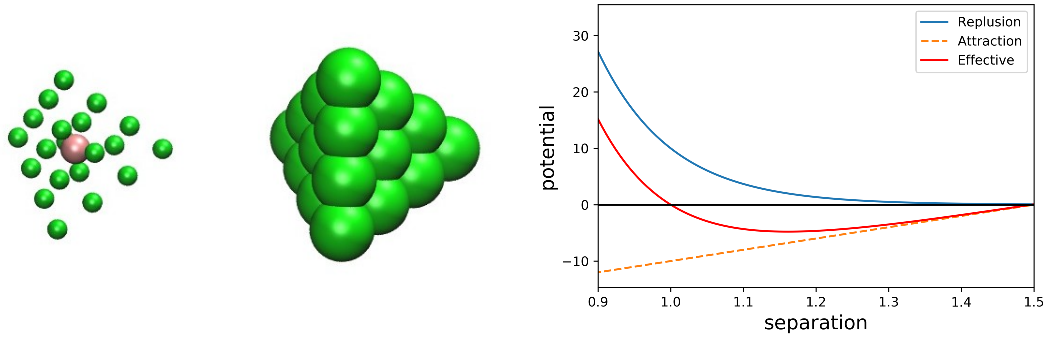


Figure 4.3. Left: a tetrahedron building block in a colloidal model. 20 green particles represent the tetrahedron core; 1 pink particle represents the mass of center. Particle size is rescaled for clarity. Right: effective potential between surface particles of different building blocks in a colloidal model.

Due to the limitation of previous scale-accurate model, a simplified colloidal model is used to accelerate the simulation and study the self assembly system at a much larger length scale. We use the effective potential calculated from the scale-accurate coarse grained model to capture the interaction between two building blocks in the colloidal model[**35**]. The potential is scaled according to the nanoparticle size and number of surface particles. In colloidal models, we only keep the surface particles on the nanoparticle core and include DNA attraction implicitly by defining the effective potential between surface particles on different building blocks. The interactions contain both attractive and repulsive parts. For each building blocks, we include a mass of center pseudo particle used for structural analysis later.

## 4.3. Results and Discussion

### 4.3.1. Tetrahedrons self assemble into ordered structure

**4.3.1.1. Results of hard particle models.** Previous research has shown that hard particles with regular tetrahedron shape can form quasicrystalline structures under high pressure[**87, 85**]. As shown in Figure 4.4, building blocks in the hard particle system form some five-fold symmetry. Furthermore, a group of building blocks aggregate into an icosahedral structure. The five-fold symmetry is not compatible with crystal periodicity, which implies that the self-assembled clusters have quasi-crystalline structures.

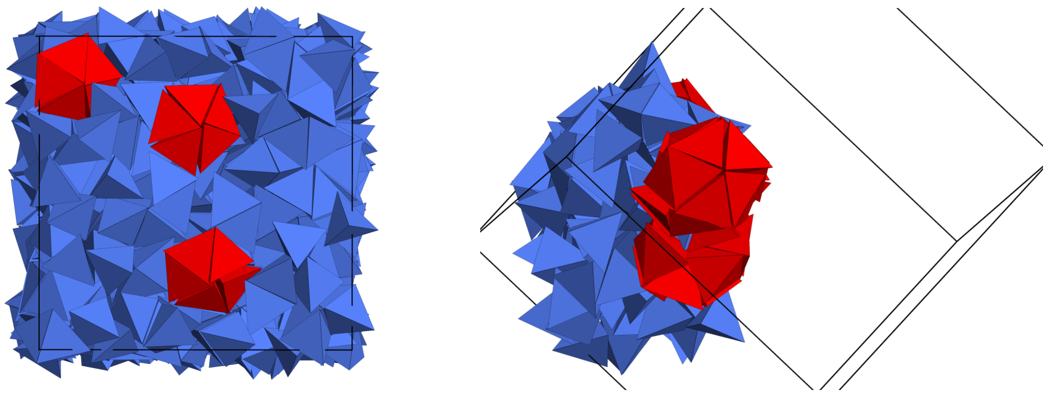


Figure 4.4. Snapshots of simulation results of 1000 hard tetrahedron particles at equilibrium in NPT ensemble. Left: the five fold symmetry is highlighted from a cluster of tetrahedron building blocks. Right: the icosahedral structure is highlighted which contains twenty building blocks. Some neighbouring building blocks are removed for clarity.

**4.3.1.2. Results of scale-accurate coarse-grained models.** However, it is impossible to self assemble nanoparticles in solution at a high pressure. With the aid of DNA hybridization, we are trying to achieve the self assembly of DNA-coated tetrahedron nanoparticles into some quasicrystalline structures at a normal pressure in solutions.

We initialized a system containing 27 DNA-coated tetrahedrons with a random configuration. After the system reached equilibrium in a NVT ensemble, five fold symmetry was observed in a cluster of five building blocks as shown in Figure 4.5. Furthermore, we observed a similar cluster as the icosahedral structure seen in hard particle system, as in Figure 4.6. Due to the system size limit of the coarse grained model, we may not see a complete icosahedral cluster. We hypothesized that these quasicrystalline patterns can be observed in systems with large length scales.

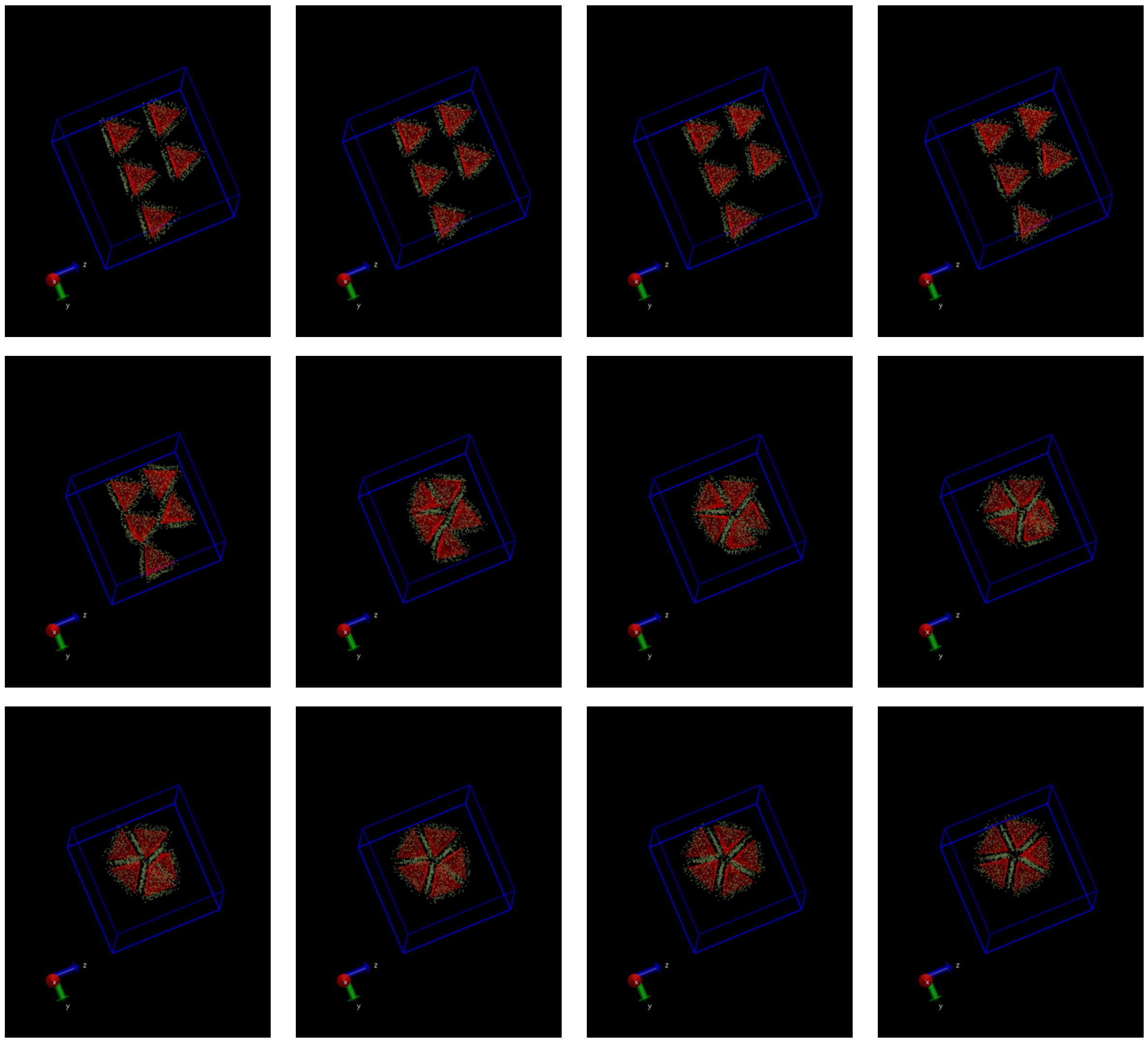


Figure 4.5. From left to right and from top to bottom, 16 snapshots of a movie show the self assembly process in scale-accurate coarse-grained model. Only five building blocks are highlighted to show the process that five fold symmetry is formed. The angle of view is precisely tuned and the rest of the building blocks are invisible for clarity.

We initialized a system containing 27 DNA-coated tetrahedrons with a random configuration. After the system reached equilibrium in a NVT ensemble, five fold symmetry was observed in a cluster of five building blocks as shown in Figure 4.5. Furthermore, we observed a similar cluster as the icosahedral structure seen in hard particle system, as in Figure 4.6. Due to the system size limit of the coarse grained model, we may not see a complete icosahedral cluster. We hypothesized that these quasicrystalline patterns can be observed in systems with large length scales.

We implemented the simulation in NPT ensemble to further test the stability of the quasicrystalline structures. In the NPT integration scheme we used for our simulations, the simulation box is fully deformable to vary in shape and size. As shown in Figure 4.6, the five fold symmetry was reserved in NPT ensemble and expand to the whole plane at a specific direction.

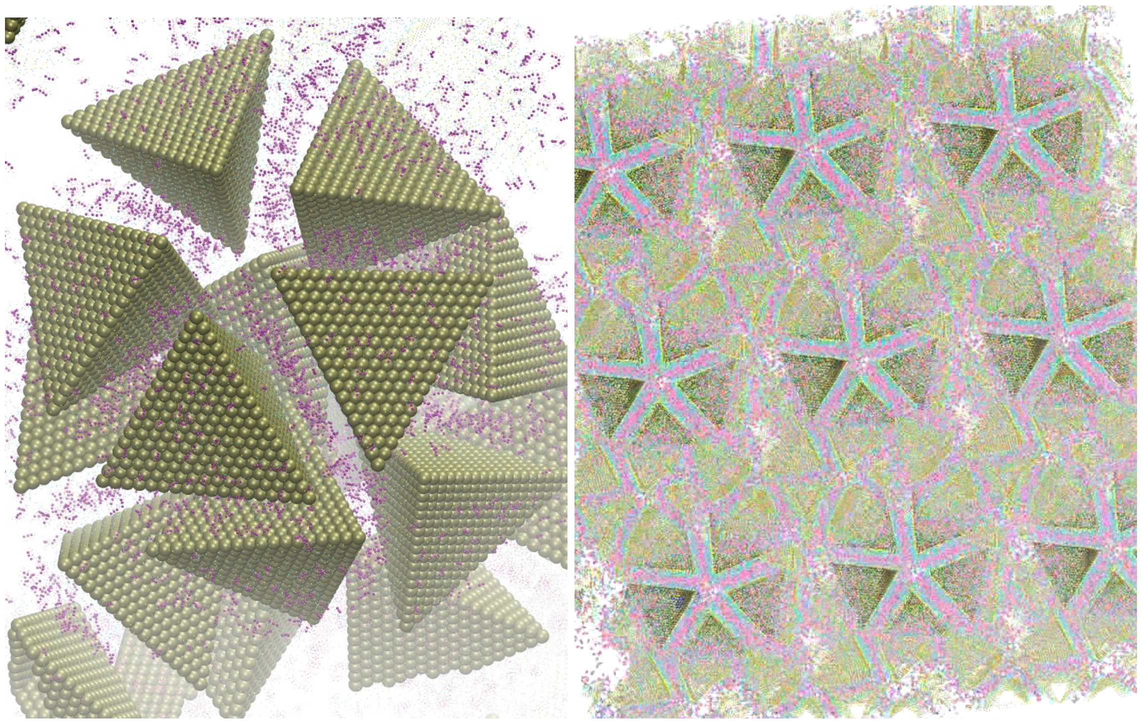


Figure 4.6. Molecular dynamics simulations of DNA-coated tetrahedron particles using scale accurate coarse grained model. Left: an icosahedral at equilibrium in a NVT ensemble. Right: quasicrystalline patterns at equilibrium in a NPT ensemble.

**4.3.1.3. Results of colloidal models.** Due to the computational cost of the scale-accurate coarse grained model, we can run simulations for systems up to tens of building blocks. It is impossible to strictly characterize the structural order of the system. We need to further simplify our models to study the self assembly at a much larger length scale. We apply the colloidal model to the system as discussed above.

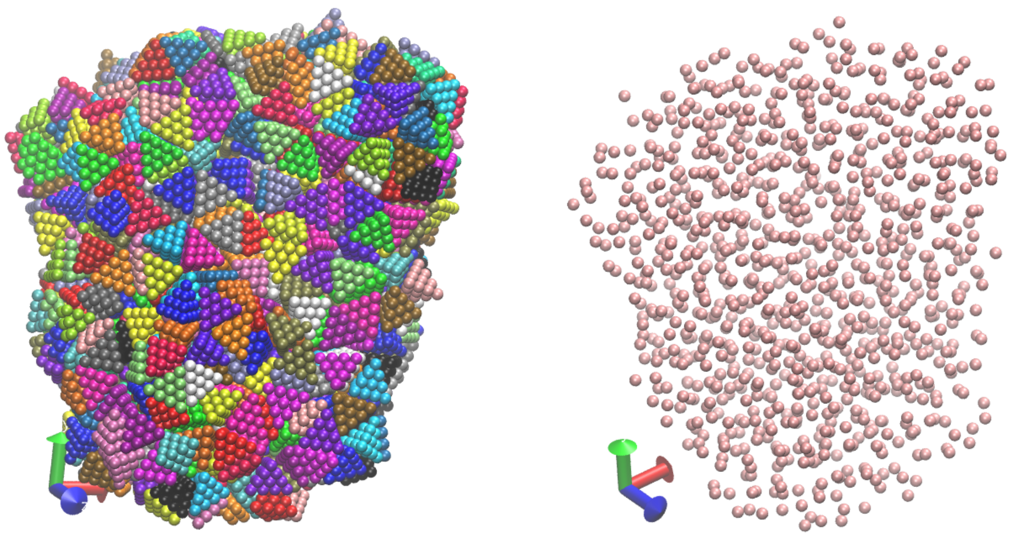


Figure 4.7. Snapshots of a system at equilibrium using colloidal models. Left: surface particles; right: mass of center particles.

As shown in Figure 4.7, we can simulate a system containing up to $1000$ building blocks. Each building block contains a mass of center particle for further structural analysis. Expectedly, we observed the five fold symmetry and icosahedral clusters in colloidal models, as shown in Figure 4.8.

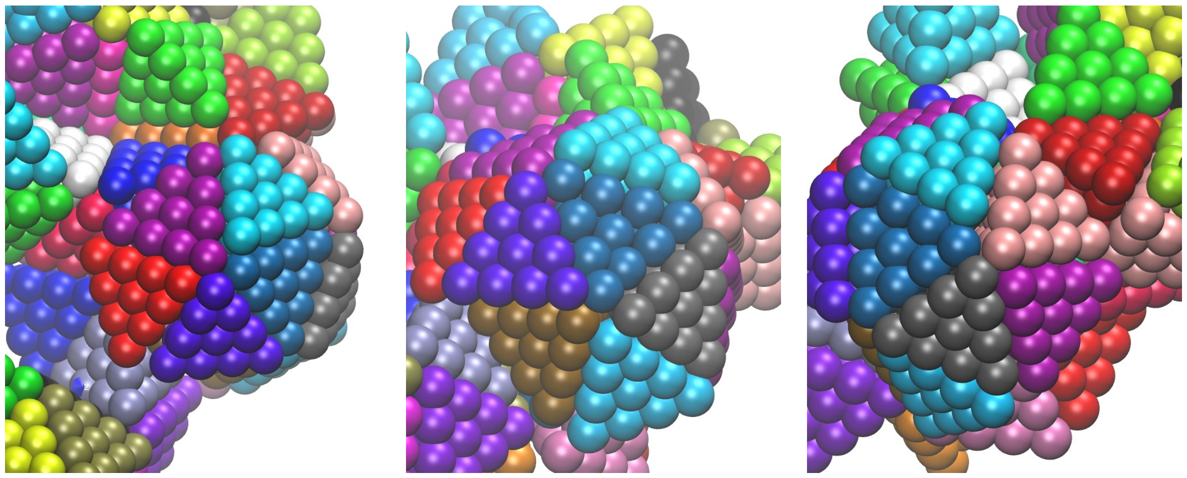


Figure 4.8. Snapshots of a cluster in a colloidal model at three different angles of view. Highlighted is part of an icosahedral structure which shows five-fold symmetry. Particles are identical but colored differently for clarity.

**4.3.1.4. Order parameter to quantify the structure.** Perfect quasicrystalls are aperiodic while extending to infinity. Therefore, they cannot be realized in simulations, which are finite systems. However, we observed patterns of five-fold symmetry in systems of regular tetrahedron assemblies at multiple length scales. The observed patterns with five-fold symmetry, which are not compatible with periodicity, are strong evidence in practice for the identification of the assembled structures as quasicrystalline structures.

To further verify our hypothesis, bond order parameter is introduced to compare and determine the symmetry and order of the resulting structures. This method is independent of the specific crystal structures and does not require the definition of a reference frame, which is suitable for the determination of quasicrystalline structures. The bond order parameters $q_l$ and $w_l$ can be used to characterize the local orientational order and the phase state of considered systems[**91**].

$$q_l(i) = \left( \frac{4\pi}{(2l+1)} \sum_{m=-l}^{m=l} |q_{lm}(i)|^2 \right)^{1/2} \tag{4.3}$$

$$w_l(i) = \sum_{m_1+m_2+m_3=0} \begin{bmatrix} l & l & l \\ m_1 & m_2 & m_3 \end{bmatrix} q_{lm_1}(i) q_{lm_2}(i) q_{lm_3}(i) \tag{4.4}$$

where $q_{lm}(i) = \frac{1}{N_{nn}} \sum_{j=1}^{N_{nn}(i)} Y_{lm}(\mathbf{r}_{ij})$, $Y_{lm}$ are the spherical harmonics. Depending on the choice of $l$, these parameters are sensitive to different crystal symmetries. Each of them depends on the angles between the vectors to the neighboring particles (“bond”) and therefore are independent of a reference frame.

We calculated the bond order parameters for both the hard particle and colloidal systems at equilibrium as shown in Figure 4.9. Surprisingly, the distributions of the $q_6$ and $w_6$ parameters match well for both systems, which means that they have the same local structural order and symmetry. Since for the hard particle system, the spontaneous formation of a quasicrystal is proved[**87**]. We can conclude that the quasicrystalline ordered structures are formed in DNA-mediated tetrahedron nanoparticle systems at multiple length scales.

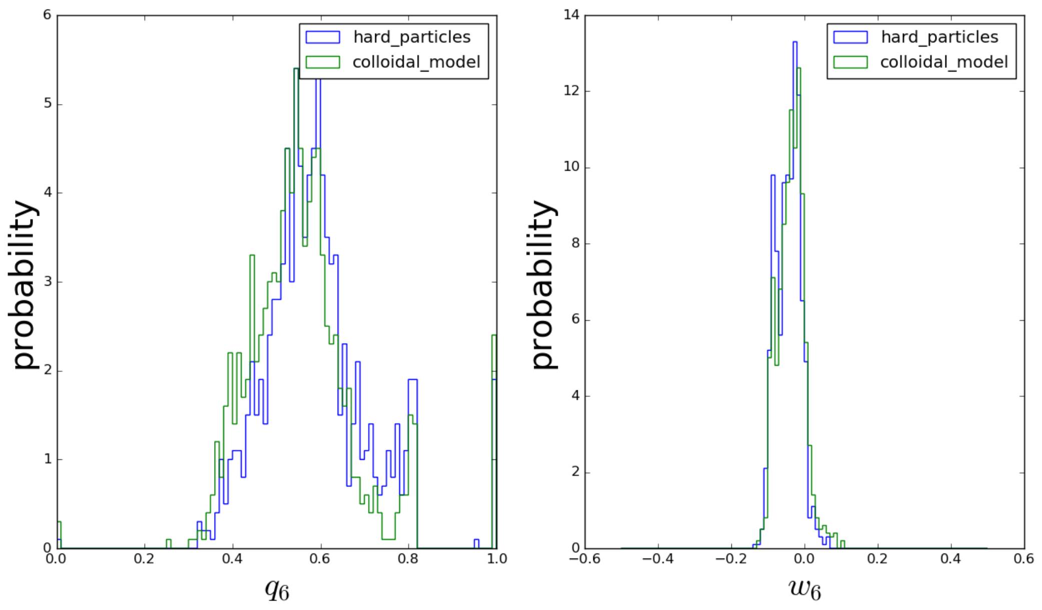


Figure 4.9. Bond order parameter of $q_6$ and $w_6$ for the hard particle system and colloidal model system. Both systems have reached equilibrium before calculating the bond order parameter.

### 4.3.2. Truncated tetrahedrons self assemble into nanocages

A new building block, truncated tetrahedron, was synthesized recently by the Mirkin group. Due to the interesting results of tetrahedron self assembly, we studied the assembly behaviour of this new building block.

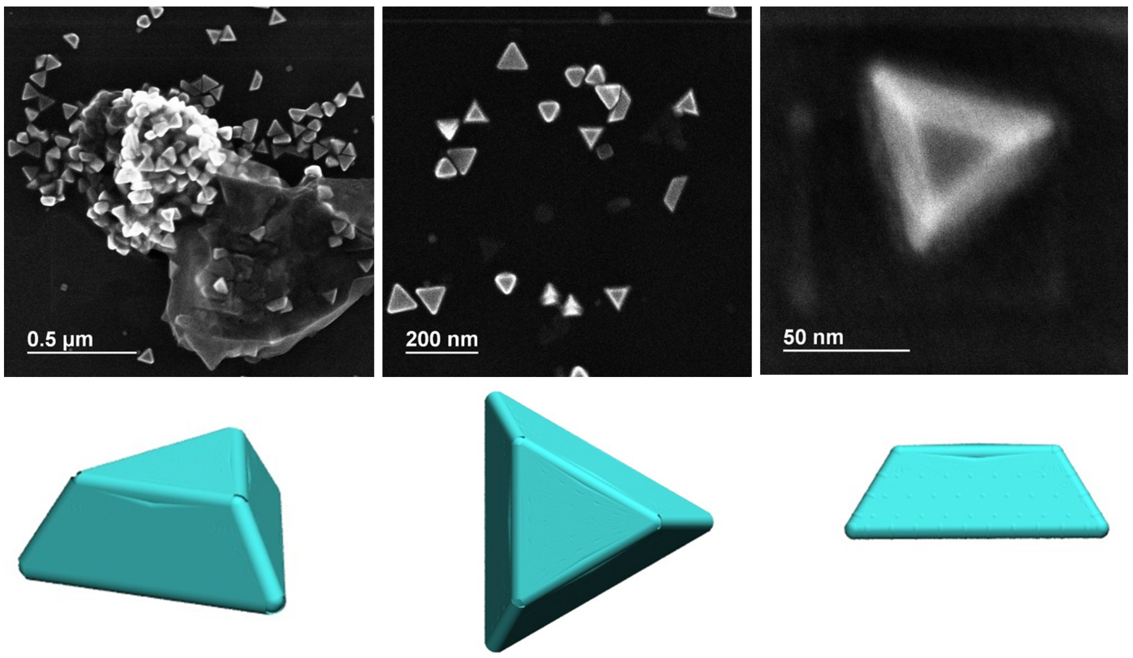


Figure 4.10. Top: SEM images of one-corner truncated tetrahedra at different length scales. Bottom: hard particle model of truncated tetrahedron building block.

As shown in Figure 4.10, the building block is one corner truncated from a regular tetrahedron, which has reduced symmetry. The side length of the top face is $L_{top}$, and the side length of the bottom face is $L_{bottom}$. We defined a truncation ratio to quantify the truncation of the nanoparticles, $\alpha_{tr} = 1 - L_{top}/L_{bottom}$. This ratio ranges from 0, which is plate-like to 1, which is a regular tetrahedron.

At $\alpha_{tr} = 0.1$ as shown in Figure 4.11, the snapshots of a molecular dynamics simulation using scale-accurate coarse-grained model show that one dimensional assembly of building blocks is formed at equilibrium. Building blocks attach to each other by the top and bottom faces to maximize the DNA hybridization. This result is consistent with the piling of DNA-coated triangular prisms[**81**].

At $\alpha_{tr} = 0.9$ as shown in Figure 4.12, the snapshots show that at equilibrium building blocks form some patterns with five-fold symmetry, which is similar to the assembly of regular tetrahedrons. It is not difficult to understand since building blocks in two systems are quite similar.

At intermediate truncation ratio $\alpha_{tr} = 0.5$ as shown in Figure 4.13, the simulation results are much more interesting, which show that at equilibrium the building blocks tend to form a cluster with icosahedral symmetry. Since the building block is one corner truncated, the resulting cluster will be an icosahedral cage, which has potential applications in optics, electronics and catalysis due to hollow interiors. Within the current simulation time, we only observed part of a closed nanocage. The current state may be far away from the equilibrium state. However, it gives us some insights that the system may finally evolve into a closed cage with icosahedral symmetry.

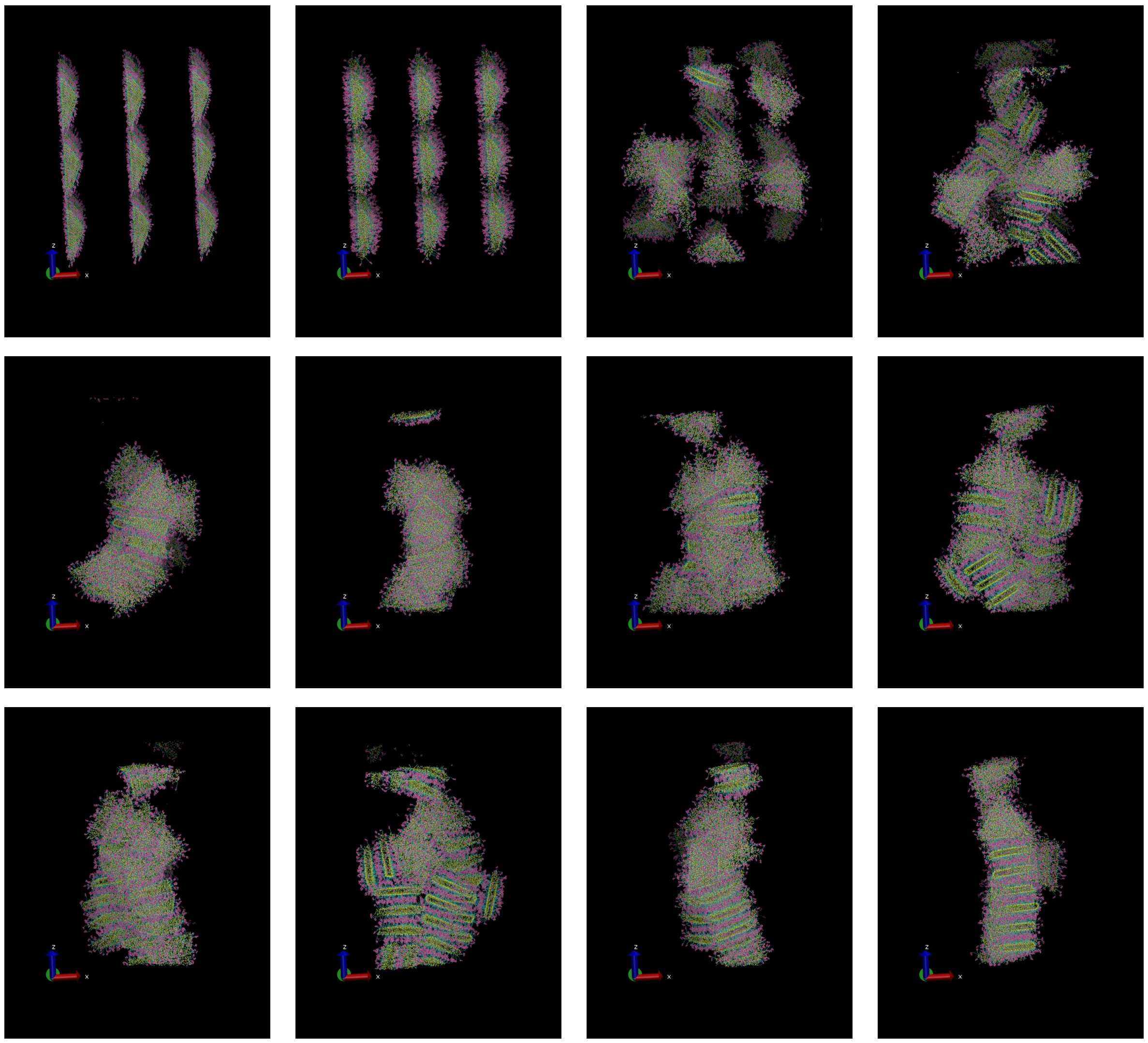


Figure 4.11. Snapshots of a molecular dynamics simulation using scale accurate coarse grained model. The building block is truncated tetrahedron with $\alpha_{tr} = 0.1$.

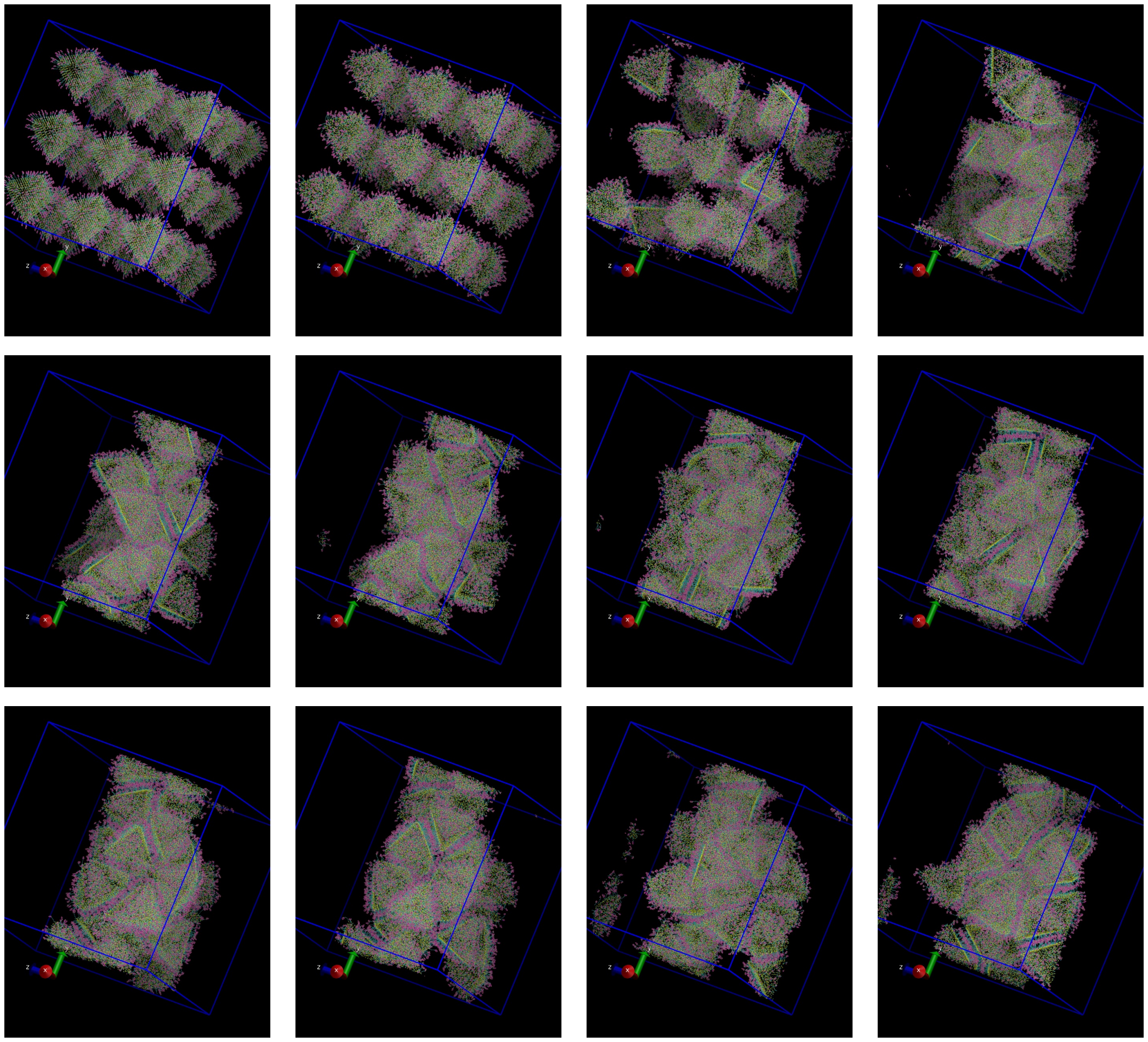


Figure 4.12. Snapshots of a molecular dynamics simulation using scale accurate coarse grained model. The building block is truncated tetrahedron with $\alpha_{tr} = 0.9$.

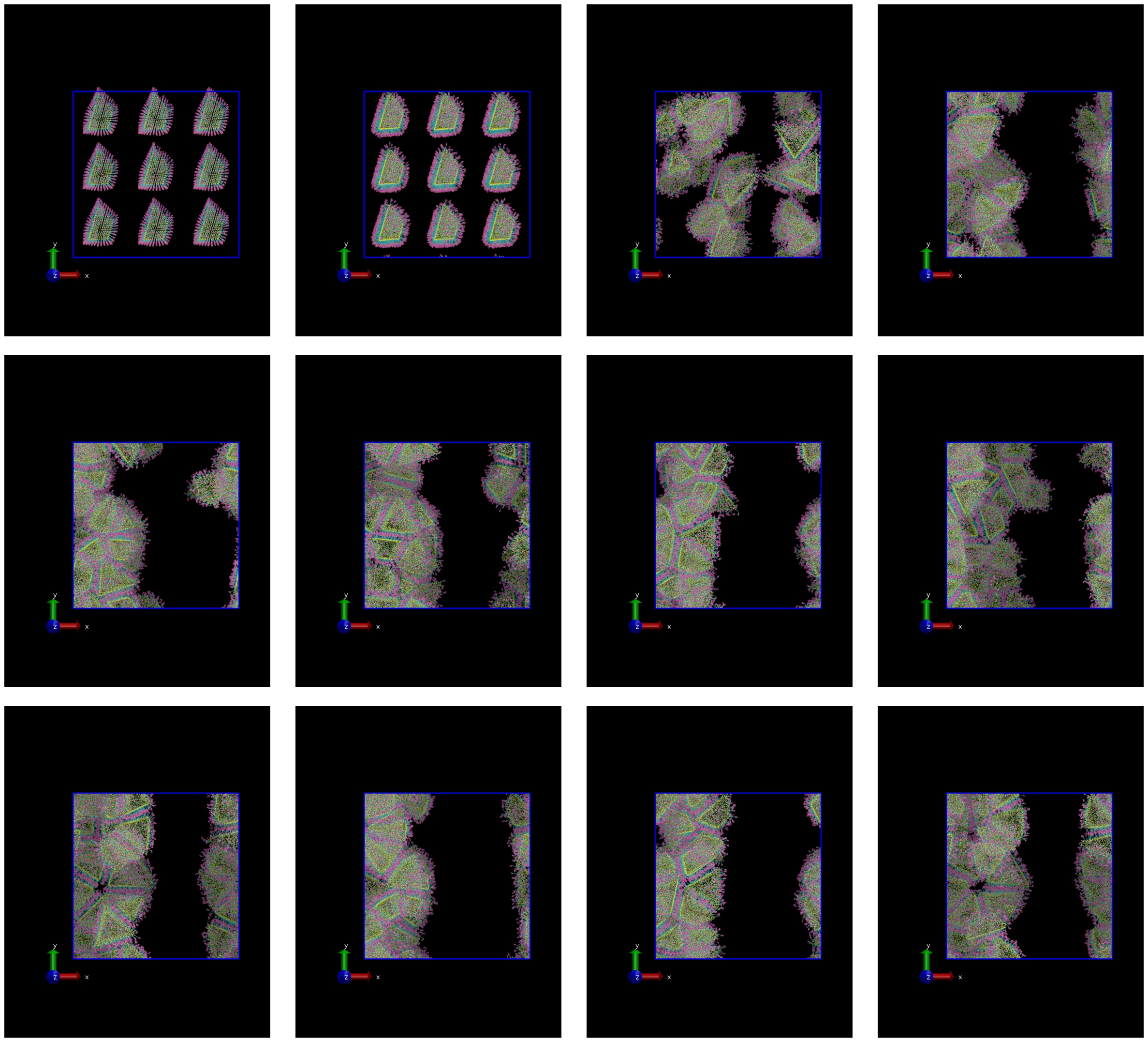
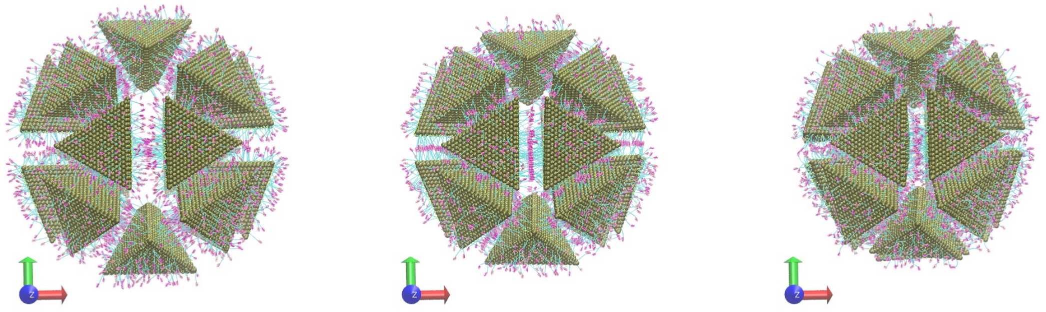


Figure 4.13. Snapshots of a molecular dynamics simulation using scale accurate coarse grained model. The building block is truncated tetrahedron with $\alpha_{tr} = 0.5$.

### 4.3.3. Stability of icosahedral cages formed by truncated-tetrahedron building blocks



(a) DNA warm up (b) start NVT (c) at equilibrium

Figure 4.14. Snapshots of molecular dynamics simulation of an icosahedral cage.

A closed icosahedral cage is nearly impossible to be formed simultaneously in the given simulation time. We may preconfigure the building blocks close to the targeted structure to test the stability of the structure. One icosahedron cluster is composed of twenty building blocks of truncated tetrahedron with truncated facet facing inside. We calculate the rotation and translation matrix for each building block and translate and rotate each building block to its targeted position. The initial configuration is an icosahedral cage as shown in Figure 4.14. The gap between neighboring building blocks is tunable. Twenty center of mass particles form a regular dodecahedron. We need to calculate the pair correlation functions for these particles to determine the structure later.

During each molecular dynamics simulation, we first freeze the nanoparticles core and let DNA linkers warm up in NVE ensemble. Then we free all the building blocks and start to run in NVT ensemble and gradually increase the temperature to a target one. We keep the system at final temperature and run a long time in NVT ensemble to test the stability of the resulting structures.

**4.3.3.1. How the gaps between building blocks affect the stability.** We study various initial configurations with (i) different gaps equals 5, 8, 10, 12, and 15 ,and (ii) three truncation ratios, $\alpha_{tr} = 0.2, 0.5, 0.8$. Except for the case $\mathrm{gap} = 15$, all the systems still keep the structures of icosahedral cages at equilibrium with some fluctuations. We calculate the pair correlation functions of center of mass particles at initial configuration and at equilibrium. Results are shown in Figure 4.15, 4.16, 4.17.

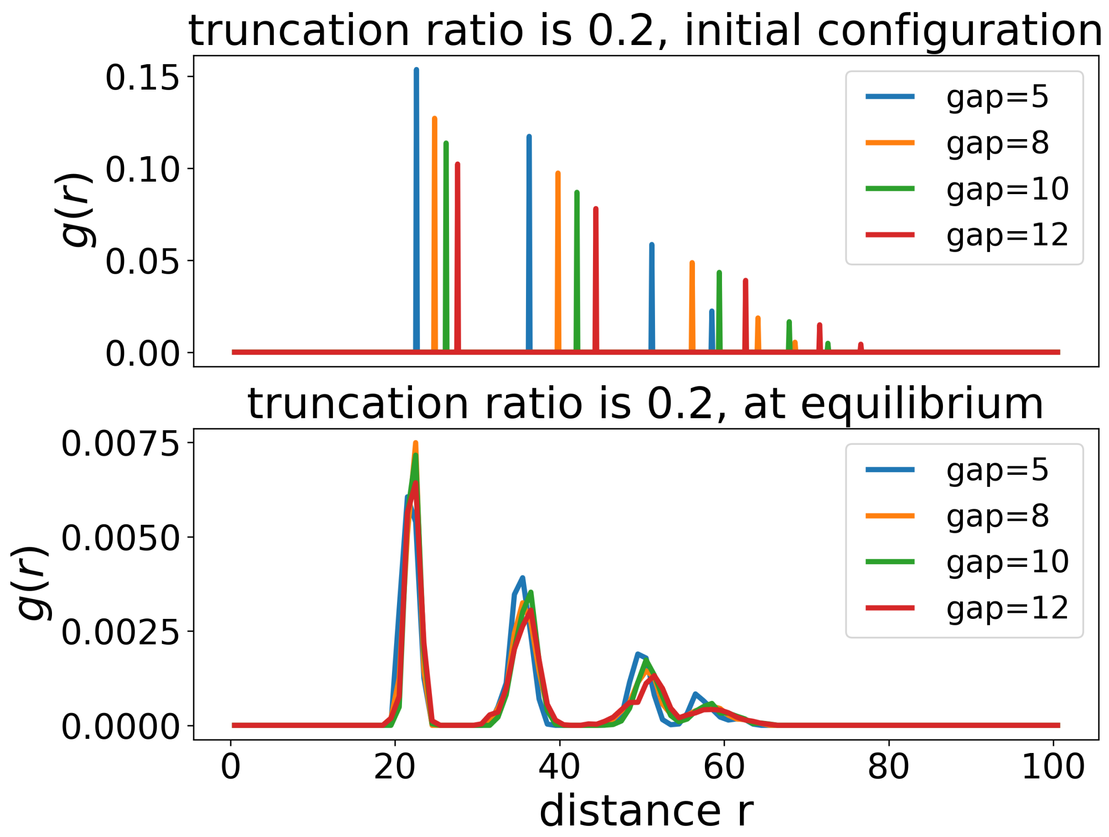


Figure 4.15. Pair correlation functions of mass centers for 20 building blocks when truncation ratio equals 0.2. Five sharp peaks represent the initial configuration of an ideal icosahedral configuration. At equilibrium, it becomes five lower and wider peaks. Initially, four systems have different sizes of icosahedral cages, while at equilibrium they all shrink to some degrees to reach the same equilibrium positions (close to the case $\mathrm{gap} = 5$) and vibrate around the equilibrium positions.

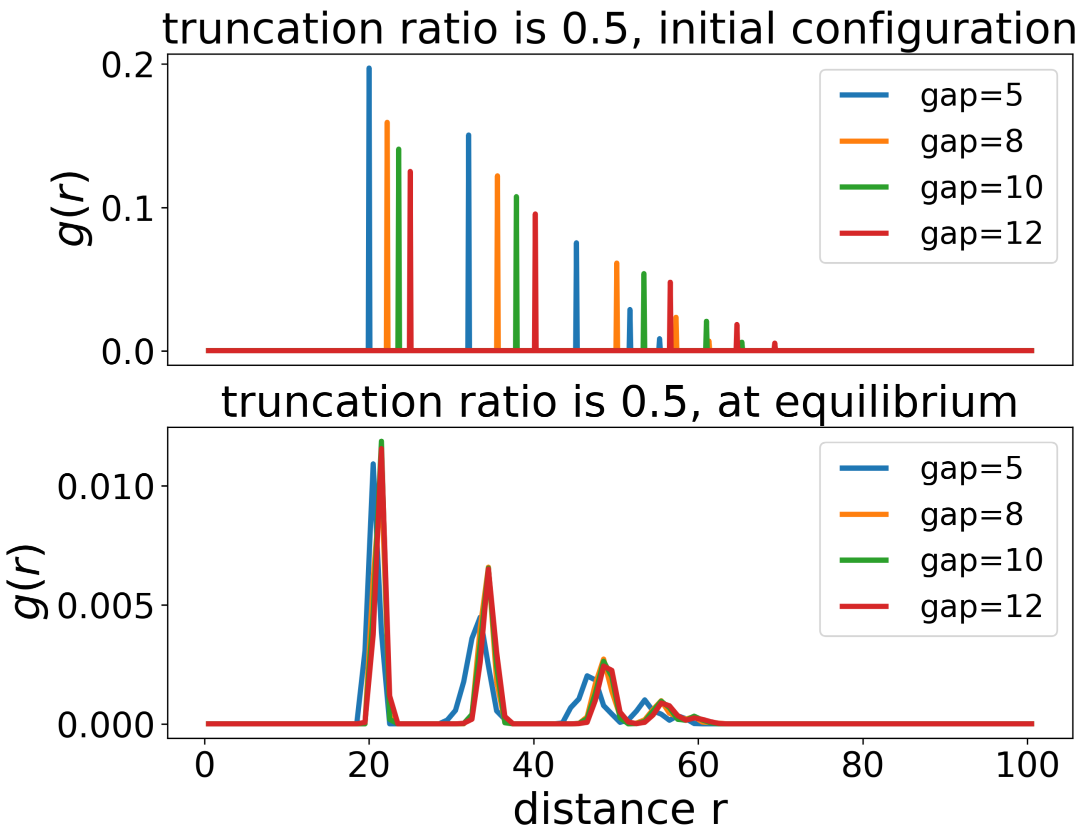


Figure 4.16. Pair correlation functions of mass centers for 20 building blocks when truncation ratio equals 0.5. Five sharp peaks represent the initial configuration of an ideal icosahedral configuration. At equilibrium, it becomes five lower and wider peaks. Initially, four systems have different sizes of icosahedral cages, while at equilibrium they shrink ($\mathrm{gap} = 10, 12$) or expand ($\mathrm{gap} = 5$) accordingly to reach the same equilibrium positions (close to the case $\mathrm{gap} = 8$) and vibrate around the equilibrium positions.

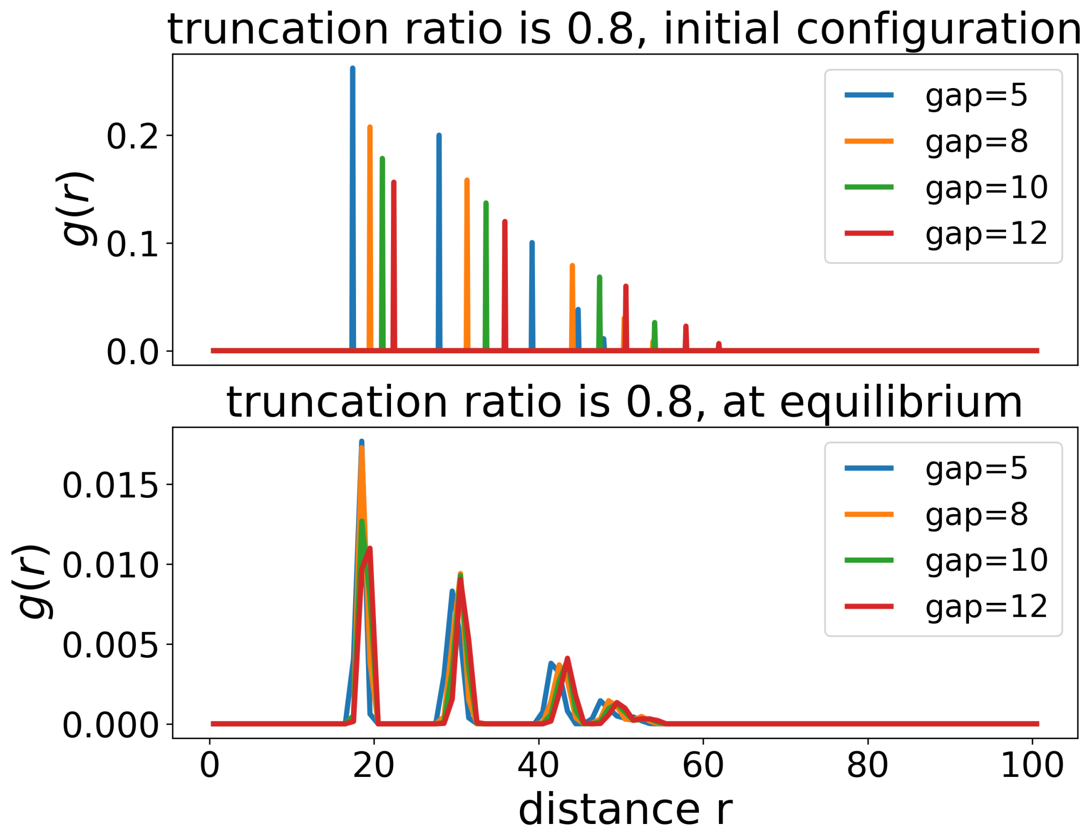


Figure 4.17. Pair correlation functions of mass centers for 20 building blocks when truncation ratio equals 0.5. Five sharp peaks represent the initial configuration of an ideal icosahedral configuration. At equilibrium, it becomes five lower and wider peaks. Initially, four systems have different sizes of icosahedral cages, while at equilibrium they shrink ($\mathrm{gap} = 8, 10, 12$) or expand ($\mathrm{gap} = 5$) accordingly to reach the same equilibrium positions (between $\mathrm{gap} = 5$ and $8$) and vibrate around the equilibrium positions.

**4.3.3.2. How truncation ratio affects the stability.** We fix the gaps between building blocks for different truncation ratios. As Figure 4.18 shows, as the truncation ratio increases, the peaks in pair correlation functions become higher and shaper, which implies that the stability of the icosahedral cage is improving.

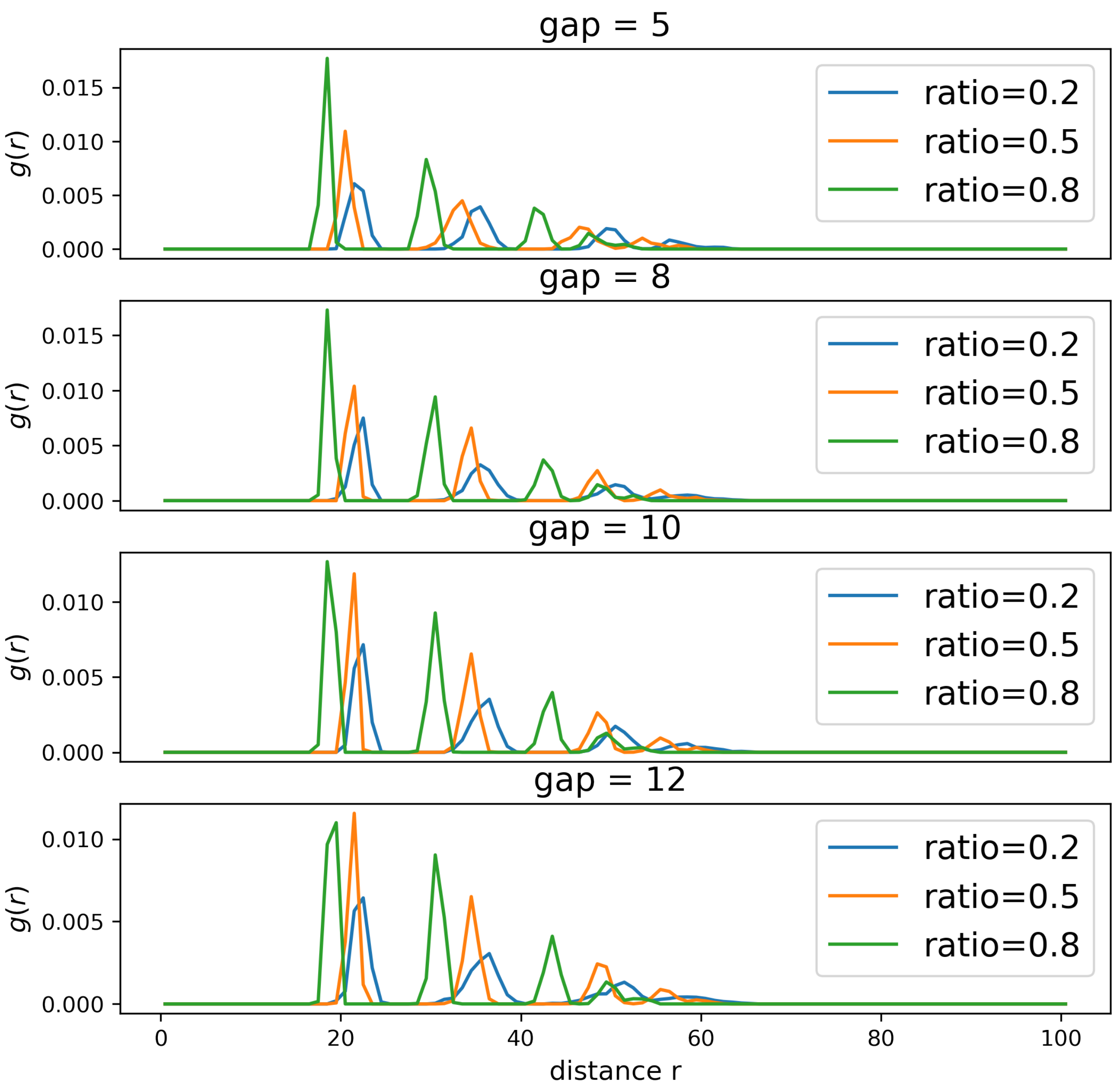


Figure 4.18. Pair correlation functions of mass center for 20 building blocks at equilibrium for different truncation ratio at a fixed initial gap between building blocks.

## 4.4. Conclusions

We conduct multi-scale molecular dynamics simulations and Monte Carlo simulations to demonstrate that quasi-crystalline five-fold symmetry is formed in DNA-directed assembly of tetrahedron nanoparticles. To further verify the structure, we calculated the bond order parameters to compare the results of the assemblies with that of known quasicrystalline structures. We conclude that tetrahedron nanoparticles can self assembly into quasicrystalline structures by DNA hybridization at multiple length scales. These simulation methods can be applied to more complex nanoparticles building blocks such as truncated tetrahedron to predict the structures of the assembly. For truncated tetrahedron building blocks, at small truncation ratios, the assembly is similar to that of nanoprisms; at high truncation ratios, the assembly is similar to that of regular tetrahedrons. Furthermore, truncated tetrahedron building blocks have a trend to form icosahedral nanocages. We showed that an icosahedral cage composed of twenty building blocks is energetically stable and will not break down if the configuration is a bit away from the equilibrium state. This process can also be driven by templates and specific DNA grafting of nanoparticles.

This work may open new avenues toward the controlled growth of nanocrystals with unconventional geometries and the tunable self-assembly of nanoparticle superstructures with increased complexity and novel functionality.

CHAPTER 5

# Conclusion

## 5.1. Summary

In Chapter 2 and Chapter 3, we studied defects in the epitaxial growth of DNA nanoparticle superlattices over multiple length scales. In Chapter 2, scale-accurate coarse-grained molecular dynamics simulations were used to investigate the DNA-directed assembly of nanoparticles on patterned substrates with the goal of building defect-free three-dimensional superlattices. The simulations revealed that DNA-coated nanoparticles can self-assemble onto DNA-coated templates driven by the DNA hybridization between complementary DNA chains linked on nanoparticles and template posts. We found that the shape anisotropy of cylindrical template posts contributes to defect formation in the epitaxial layer. We concluded that defects could be reduced by adjusting the spatial probability of DNA hybridization by precisely controlling parameters such as shape of the template post, size of the template core, and length of the DNA linkers. We developed design rules for fabricating two-dimensional template patterns to dramatically reduce defects on adsorbed monolayers. This detailed model has proved to be an easy and useful tool to obtain crucial experimental parameters, which are generally difficult or expensive to obtain from experiments.

In Chapter 3, we performed Monte Carlo simulations to systematically study the defect structures in the epitaxial growth of DNA functionalized nanoparticles on a patterned substrate. With the advantage of Monte Carlo simulations, we were able to study the system in a macroscopic scale rather than a microscopic scale. By investigating the influence of tailored parameters, including the three binding energies of different attachment sites, we observed that structural defects within epitaxial layers are correlated and aggregate to form one-dimensional clusters with an exponential length distribution, which resembles the distribution found in worm-like micelles. These linear defects undergo an orientational disordered-ordered phase transition, which is analogous to the spontaneous magnetization transition in the 2D ferromagnetic Ising model. We showed that the ordering transition of defect lines is entropy-driven due to the excluded volume effects of defect lines in perpendicular orientations. We deduced a phase boundary condition that is similar to that in a hard-rod view of liquid crystals. By studying defect formation in the epitaxial layer of DNA functionalized nanoparticles and how to engineer these defects into different structures of great interest, we hope to open new ways to design and fabricate nanomaterials with controllable defect structures for real applications.

In Chapter 4, we conducted both multi-scale molecular dynamics simulations and Monte Carlo simulations to demonstrate that quasi-crystalline five-fold symmetry is formed in the DNA-directed assembly of tetrahedron nanoparticles at different length scales. These simulation methods can be applied to the DNA-directed assembly of more complex nanoparticles (truncated tetrahedron) to predict the structures of these assemblies. We showed that an icosahedral cage composed of twenty building blocks is energetically stable and this assembly process can be driven by templates and specific DNA grafting of nanoparticles. Due to the unique and tunable optical, electronic, and catalytic properties of nanostructures, these structures have many potential applications. This work may open new avenues toward the controlled growth of nanocrystals with unconventional geometries and the tunable self-assembly of nanoparticle superstructures with increased complexity and novel functionality.

## 5.2. Future Work

### 5.2.1. Proteins as new building blocks

Proteins, the most versatile building blocks in nature, play an important role in the structural and functional complexity of life [**92**]. The study of protein self-assembly not only helps to better understand natural processes, but also provide novel functional materials with various potential applications. Researchers are employing proteins as new building blocks to self assemble into one-, two- and three-dimensional nanostructures. This work takes advantage of the diverse structures, orthogonal surface chemistries, and site-specific modification of proteins to assemble single- and multicomponent superlattices with symmetries that are difficult to obtain using spherical PAEs. For such complex building blocks, multi-scale modeling and simulations are quite helpful in achieving the goal of precise control over the orientation and spatial arrangement of the nanostructures.

# Bibliography

[1] Park, S. Y, Lytton-Jean, A. K. R, Lee, B, Weigand, S, Schatz, G. C, & Mirkin, C. A. (2008) DNA-programmable nanoparticle crystallization. *Nature* **451**, 553–556.

[2] Macfarlane, R. J, Lee, B, Jones, M. R, Harris, N, Schatz, G. C, & Mirkin, C. A. (2011) Nanoparticle superlattice engineering with DNA. *Science* **334**, 204–208.

[3] O'Brien, M. N, Lin, H.-X, Girard, M, Olvera de la Cruz, M, & Mirkin, C. A. (2016) Programming colloidal crystal habit with anisotropic nanoparticle building blocks and DNA bonds. *Journal of the American Chemical Society* **138**, 14562–14565.

[4] Humphrey, W, Dalke, A, & Schulten, K. (1996) VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics* **14**, 33–38.

[5] Stone, J. (1998) Master's thesis (Computer Science Department, University of Missouri-Rolla).

[6] Hellstrom, S. L, Kim, Y, Fakonas, J. S, Senesi, A. J, Macfarlane, R. J, Mirkin, C. A, & Atwater, H. A. (2013) Epitaxial Growth of DNA-Assembled Nanoparticle Superlattices on Patterned Substrates. *Nano Letters* **13**, 6084–6090.

[7] Mirkin, C. A, Letsinger, R. L, & Mucic, R. C. (1996) A DNA-based method for rationally assembling nanoparticles into macroscopic materials. *Nature* **382**, 607–609.

[8] Alivisatos, A. P, Johnsson, K. P, Peng, X, & Wilson, T. E. (1996) Organization of ‘nanocrystal’ molecules using DNA. *Nature* **382**, 609–611.

[9] Mirkin, C. A. (2010) The polyvalent gold nanoparticle conjugate materials synthesis , and intracellular gene regulation. *MRS Bulletin* **35**, 532–540.

[10] Carter, J. D & LaBean, T. H. (2011) Organization of inorganic nanomaterials via programmable DNA self-assembly and peptide molecular recognition. *ACS Nano* **5**, 2200–2205.

[11] Cutler, J. I, Auyeung, E, & Mirkin, C. A. (2012) Spherical nucleic acids. *Journal of the American Chemical Society* **134**, 1376–1391.

[12] Pompa, P. P, Martiradonna, L, Della Torre, A, Della Sala, F, Manna, L, De Vittorio, M, Calabi, F, Cingolani, R, & Rinaldi, R. (2006) Metal-enhanced fluorescence of colloidal nanocrystals with nanoscale control. *Nature Nanotechnology* **1**, 126–130.

[13] Grunes, J, Zhu, J, Anderson, E. A, & Somorjai, G. A. (2002) Ethylene hydrogenation over platinum nanoparticle array model catalysts fabricated by electron beam lithography: Determination of active metal surface area. *The Journal of Physical Chemistry B* **106**, 11463–11468.

[14] Bell, A. T. (2003) The impact of nanoscience on heterogeneous catalysis. *Science* **299**, 1688–1691.

[15] Lin, L, Liu, Y, Tang, L, & Li, J. (2011) Electrochemical DNA sensor by the assembly of graphene and DNA-conjugated gold nanoparticles with silver enhancement strategy. *The Analyst* **136**, 4732–4737.

[16] Maye, M. M, Kumara, M. T, Nykypanchuk, D, Sherman, W. B, & Gang, O. (2010) Switching binary states of nanoparticle superlattices and dimer clusters by DNA strands. *Nature Nanotechnology* **5**, 116–120.

[17] Kildishev, A. V, Boltasseva, A, & Shalaev, V. M. (2013) Planar photonics with metasurfaces. *Science* **339**, 1232009.

[18] Stebe, K. J, Lewandowski, E, & Ghosh, M. (2009) Oriented assembly of metamaterials. *Science* **325**, 159–160.

[19] Sebba, D. S, Mock, J. J, Smith, D. R, LaBean, T. H, & Lazarides, A. A. (2008) Reconfigurable core- satellite nanoassemblies as molecularly-driven plasmonic switches. *Nano Letters* **8**, 1803–1808.

[20] Wang, F, Li, C, Chen, H, Jiang, R, Sun, L.-D, Li, Q, Wang, J, Yu, J. C, & Yan, C.-H. (2013) Plasmonic harvesting of light energy for suzuki coupling reactions. *Journal of the American Chemical Society* **135**, 5588–5601.

[21] Jones, M. R, Osberg, K. D, Macfarlane, R. J, Langille, M. R, & Mirkin, C. A. (2011) Templated techniques for the synthesis and assembly of plasmonic nanostructures. *Chemical Reviews* **111**, 3736–3827.

[22] Kelly, K. L, Coronado, E, Zhao, L. L, & Schatz, G. C. (2003) The optical properties of metal nanoparticles: The influence of size, shape, and dielectric environment. *The Journal of Physical Chemistry B* **107**, 668–677.

[23] Fan, J. A, Wu, C, Bao, K, Bao, J, Bardhan, R, Halas, N. J, Manoharan, V. N, Nordlander, P, Shvets, G, & Capasso, F. (2010) Self-assembled plasmonic nanoparticle clusters. *Science* **328**, 1135–1138.

[24] Park, D. J, Zhang, C, Ku, J. C, Zhou, Y, Schatz, G. C, & Mirkin, C. A. (2014) Plasmonic photonic crystals realized through DNA-programmable assembly. *Proceedings of the National Academy of Sciences* p. 201422649.

[25] Mucic, R. C, Storhoff, J. J, Mirkin, C. A, & Letsinger, R. L. (1998) DNA-directed synthesis of binary nanoparticle network materials. *Journal of the American Chemical Society* **120**, 12674–12675.

[26] Jin, R, Wu, G, Li, Z, Mirkin, C. A, & Schatz, G. C. (2003) What controls the melting properties of DNA-linked gold nanoparticle assemblies? *Journal of the American Chemical Society* **125**, 1643–1654.

[27] Valignat, M.-P, Theodoly, O, Crocker, J. C, Russel, W. B, & Chaikin, P. M. (2005) Reversible self-assembly and directed assembly of DNA-linked micrometer-sized colloids. *Proceedings of the National Academy of Sciences* **102**, 4225–4229.

[28] Maye, M. M, Nykypanchuk, D, Lelie, D. V. D, & Gang, O. (2006) A simple method for kinetic control of DNA-induced nanoparticle assembly. *Journal of the American Chemical Society* **128**, 14020–14021.

[29] Geerts, N, Schmatko, T, & Eiser, E. (2008) Clustering versus percolation in the assembly of colloids coated with long DNA. *Langmuir : the ACS Journal of Surfaces and Colloids* **24**, 5118–5123.

[30] Biancaniello, P, Kim, A. J, & Crocker, J. (2005) Colloidal interactions and self-assembly using DNA hybridization. *Physical Review Letters* **94**, 94–97.

[31] Hill, H. D, Macfarlane, R. J, Senesi, A. J, Lee, B, Park, S. Y, & Mirkin, C. A. (2008) Controlling the lattice parameters of gold nanoparticle FCC crystals with duplex DNA linkers. *Nano Letters* **8**, 2341–2344.

[32] Nykypanchuk, D, Maye, M. M, van der Lelie, D, & Gang, O. (2008) DNA-guided crystallization of colloidal nanoparticles. *Nature* **451**, 549–552.

[33] Macfarlane, R. J, Jones, M. R, Senesi, A. J, Young, K. L, Lee, B, Wu, J, & Mirkin, C. A. (2010) Establishing the design rules for DNA-mediated programmable colloidal crystallization. *Angewandte Chemie International Edition* **49**, 4589–4592.

[34] Auyeung, E, Cutler, J. I, Macfarlane, R. J, Jones, M. R, Wu, J, Liu, G, Zhang, K, Osberg, K. D, & Mirkin, C. A. (2012) Synthetically programmable nanoparticle superlattices using a hollow three-dimensional spacer approach. *Nature Nanotechnology* **7**, 24–28.

[35] Auyeung, E, Li, T. I. N. G, Senesi, A. J, Schmucker, A. L, Pals, B. C, Olvera de La Cruz, M, & Mirkin, C. A. (2014) DNA-mediated nanoparticle crystallization into Wulff polyhedra. *Nature* **505**, 73–77.

[36] Macfarlane, R. J, Jones, M. R, Lee, B, Auyeung, E, & Mirkin, C. A. (2013) Topotactic interconversion of nanoparticle superlattices. *Science* **341**, 1222–1225.

[37] Sun, W, Boulais, E, Hakobyan, Y, Wang, W. L, Guan, A, Bathe, M, & Yin, P. (2014) Casting inorganic structures with DNA molds. *Science* **346**.

[38] Jones, M. R, Seeman, N. C, & Mirkin, C. A. (2015) Programmable materials and the nature of the DNA bond. *Science* **347**, 1260901.

[39] Radha, B, Senesi, A. J, O'Brien, M. N, Wang, M. X, Auyeung, E, Lee, B, & Mirkin, C. A. (2014) Reconstitutable nanoparticle superlattices. *Nano Letters* **14**, 2162–2167.

[40] Tkachenko, A. V. (2011) Theory of programmable hierarchic self-assembly. *Physical Review Letters* **106**, 255501.

[41] Shevchenko, E. V, Talapin, D. V, Kotov, N. A, O'Brien, S, & Murray, C. B. (2006) Structural diversity in binary nanoparticle superlattices. *Nature* **439**, 55–59.

[42] Lukatsky, D. B, Mulder, B. M, & Frenkel, D. (2006) Designing ordered DNA-linked nanoparticle assemblies. *Journal of Physics: Condensed Matter* **18**, S567–S580.

[43] Largo, J, Starr, F. W, & Sciortino, F. (2007) Self-assembling DNA dendrimers: a numerical study. *Langmuir* **23**, 5896–5905.

[44] Lee, O.-S & Schatz, G. C. (2009) Molecular dynamics simulation of DNA-functionalized gold nanoparticles. *The Journal of Physical Chemistry C* **113**, 2316–2321.

[45] Padovan-Merhar, O, Lara, F. V, & Starr, F. W. (2011) Stability of DNA-linked nanoparticle crystals: effect of number of strands, core size, and rigidity of strand attachment. *The Journal of Chemical Physics* **134**, 244701.

[46] Martinez-Veracoechea, F. J, Mladek, B. M, Tkachenko, A. V, & Frenkel, D. (2011) Design rule for colloidal crystals of DNA-functionalized particles. *Physical Review Letters* **107**, 045902.

[47] Knorowski, C, Burleigh, S, & Travesset, A. (2011) Dynamics and statics of DNA-programmable nanoparticle self-assembly and crystallization. *Physical Review Letters* **106**, 215501.

[48] Fan, J. A, He, Y, Bao, K, Wu, C, Bao, J, Schade, N. B, Manoharan, V. N, Shvets, G, Nordlander, P, Liu, D. R, & Capasso, F. (2011) DNA-enabled self-assembly of plasmonic nanoclusters. *Nano Letters* **11**, 4859–4864.

[49] Li, T. I. N. G, Sknepnek, R, Macfarlane, R. J, Mirkin, C. A, & Olvera de la Cruz, M. (2012) Modeling the crystallization of spherical nucleic acid nanoparticle conjugates with molecular dynamics simulations. *Nano Letters* **12**, 2509–2514.

[50] Starr, F. W & Sciortino, F. (2006) Model for assembly and gelation of four-armed dna dendrimers. *Journal of Physics: Condensed Matter* **18**, L347.

[51] Jones, M. R, Macfarlane, R. J, Lee, B, Zhang, J, Young, K. L, Senesi, A. J, & Mirkin, C. A. (2010) DNA-nanoparticle superlattices formed from anisotropic building blocks. *Nature Materials* **9**, 913–917.

[52] Travesset, A. (2011) Self-assembly enters the design era. *Science* **334**, 183–184.

[53] Tan, S. J, Campolongo, M. J, Luo, D, & Cheng, W. (2011) Building plasmonic nanostructures with DNA. *Nature Nanotechnology* **6**, 268–276.

[54] Sun, S, Murray, C. B, Weller, D, Folks, L, & Moser, A. (2000) Monodisperse fept nanoparticles and ferromagnetic fept nanocrystal superlattices. *Science* **287**, 1989–1992.

[55] Urban, J. J, Talapin, D. V, Shevchenko, E. V, Kagan, C. R, & Murray, C. B. (2007) Synergism in binary nanocrystal superlattices leads to enhanced p-type conductivity in self-assembled pbte/ag2te thin films. *Nature materials* **6**, 115–121.

[56] Park, D. J, Zhang, C, Ku, J. C, Zhou, Y, Schatz, G. C, & Mirkin, C. A. (2015) Plasmonic photonic crystals realized through dna-programmable assembly. *Proceedings of the National Academy of Sciences* **112**, 977–981.

[57] Hellstrom, S. L, Kim, Y, Fakonas, J. S, Senesi, A. J, Macfarlane, R. J, Mirkin, C. A, & Atwater, H. A. (2013) Epitaxial growth of DNA-assembled nanoparticle superlattices on patterned substrates. *Nano Letters* **13**, 6084–6090.

[58] Li, T. I. N. G, Sknepnek, R, & Olvera de la Cruz, M. (2013) Thermally active hybridization drives the crystallization of DNA-functionalized nanoparticles. *Journal of the American Chemical Society* **135**, 8535–8541.

[59] (2015) Hoomd-blue web page: http://codeblue.umich.edu/hoomd-blue.

[60] Anderson, J. A, Lorenz, C. D, & Travesset, A. (2008) General purpose molecular dynamics simulations fully implemented on graphics processing units. *Journal of Computational Physics* **227**, 5342–5359.

[61] Glaser, J, Nguyen, T. D, Anderson, J. A, Lui, P, Spiga, F, Millan, J. A, Morse, D. C, & Glotzer, S. C. (2015) Strong scaling of general-purpose molecular dynamics simulations on GPUs. *Computer Physics Communications* **192**, 97 – 107.

[62] Nguyen, T. D, Phillips, C. L, Anderson, J. A, & Glotzer, S. C. (2011) Rigid body constraints realized in massively-parallel molecular dynamics on graphics processing units. *Computer Physics Communications* **182**, 2307–2313.

[63] Nosé, S. (1984) A molecular dynamics method for simulations in the canonical ensemble. *Molecular Physics* **52**, 255–268.

[64] Hoover, W. G. (1985) Canonical dynamics: Equilibrium phase-space distributions. *Physical Review A* **31**, 1695.

[65] Grzelczak, M, Vermant, J, Furst, E. M, & Liz-Marzn, L. M. (2010) Directed self-assembly of nanoparticles. *ACS Nano* **4**, 3591–3605. PMID: 20568710.

[66] Mittal, M & Furst, E. M. (2009) Electric field-directed convective assembly of ellipsoidal colloidal particles to create optically and mechanically anisotropic thin films. *Advanced Functional Materials* **19**, 3271–3278.

[67] Rogers, W. B, Shih, W. M, & Manoharan, V. N. (2016) Using dna to program the self-assembly of colloidal nanoparticles and microparticles. *Nature Reviews Materials* **1**, 16008.

[68] Knorowski, C, Burleigh, S, & Travesset, A. (2011) Dynamics and Statics of DNA-Programmable Nanoparticle Self-Assembly and Crystallization. *Physical Review Letters* **106**, 215501.

[69] Auyeung, E, Li, T. I. N. G, Senesi, A. J, Schmucker, A. L, Pals, B. C, de la Cruz, M. O, & Mirkin, C. A. (2014) DNA-mediated nanoparticle crystallization into Wulff polyhedra. *Nature* **505**, 73–77.

[70] Tkachenko, A. V. (2002) Morphological Diversity of DNA-Colloidal Self-Assembly. *Physical Review Letters* **89**, 148303.

[71] Senesi, A. J, Eichelsdoerfer, D. J, Brown, K. A, Lee, B, Auyeung, E, Choi, C. H. J, Macfarlane, R. J, Young, K. L, & Mirkin, C. A. (2014) Oligonucleotide Flexibility Dictates Crystal Quality in DNA-Programmable Nanoparticle Superlattices. *Advanced Materials* **26**, 7235–7240.

[72] Macfarlane, R. J, Lee, B, Jones, M. R, Harris, N, Schatz, G. C, & Mirkin, C. A. (2011) Nanoparticle Superlattice Engineering with DNA. *Science* **334**, 204–208.

[73] Kershner, R. J, Bozano, L. D, Micheel, C. M, Hung, A. M, Fornof, A. R, Cha, J. N, Rettner, C. T, Bersani, M, Frommer, J, Rothemund, P. W. K, & Wallraff, G. M. (2009) Placement and orientation of individual DNA shapes on lithographically patterned surfaces. *Nature Nanotechnology* **4**, 557–561.

[74] Padovan-Merhar, O, Lara, F. V, & Starr, F. W. (2011) Stability of DNA-linked nanoparticle crystals: effect of number of strands, core size, and rigidity of strand attachment. *The Journal of chemical physics* **134**, 244701.

[75] Li, T. I, Sknepnek, R, Macfarlane, R. J, Mirkin, C. A, & Olvera de la Cruz, M. (2012) Modeling the Crystallization of Spherical Nucleic Acid Nanoparticle Conjugates with Molecular Dynamics Simulations. *Nano Letters* **12**, 2509–2514.

[76] Macfarlane, R, Jones, M, Senesi, A, Young, K, Lee, B, Wu, J, & Mirkin, C. (2010) Establishing the Design Rules for DNA-Mediated Programmable Colloidal Crystallization. *Angewandte Chemie International Edition* **49**, 4589–4592.

[77] Macfarlane, R. J, O'Brien, M. N, Petrosko, S. H, & Mirkin, C. A. (2013) Nucleic acid-modified nanostructures as programmable atom equivalents: forging a new "table of elements". *Angewandte Chemie International Edition* **52**, 5688–5698.

[78] Terrones, H, Lv, R, Terrones, M, & Dresselhaus, M. S. (2012) The role of defects and doping in 2d graphene sheets and 1d nanoribbons. *Reports on Progress in Physics* **75**, 062501.

[79] Pan, S, Li, T, & Olvera de la Cruz, M. (2016) Molecular dynamics simulation of dna-directed assembly of nanoparticle superlattices using patterned templates. *Journal of Polymer Science Part B: Polymer Physics* **54**, 1687–1692.

[80] Cates, M. E & Candau, S. J. (1990) Statics and dynamics of worm-like surfactant micelles. *Journal of Physics: Condensed Matter* **2**, 6869.

[81] Jones, M. R, Macfarlane, R. J, Lee, B, Zhang, J, Young, K. L, Senesi, A. J, & Mirkin, C. A. (2010) DNA-nanoparticle superlattices formed from anisotropic building blocks. *Nature Materials* **9**, 913–917.

[82] O'Brien, M. N, Jones, M. R, Lee, B, & Mirkin, C. A. (2015) Anisotropic nanoparticle complementarity in DNA-mediated co-crystallization. *Nature Materials* **14**, 833–839.

[83] Lin, H, Lee, S, Sun, L, Spellings, M, Engel, M, Glotzer, S. C, & Mirkin, C. A. (2017) Clathrate colloidal crystals. *Science* **355**, 931–935.

[84] O'Brien, M. N, Girard, M, Lin, H.-X, Millan, J. A, De La Cruz, M. O, Lee, B, & Mirkin, C. A. (2016) Exploring the zone of anisotropy and broken symmetries in dna-mediated nanoparticle crystallization. *Proceedings of the National Academy of Sciences* **113**, 10485–10490.

[85] Haji-Akbari, A, Engel, M, & Glotzer, S. C. (2011) Phase diagram of hard tetrahedra. *The Journal of Chemical Physics* **135**, 194101.

[86] Fang, F, Kovacs, J, Sadler, G, & Irwin, K. (2013) An Icosahedral Quasicrystal as a Packing of Regular Tetrahedra. *arXiv:1311.3994 [cond-mat]*. arXiv: 1311.3994.

[87] Haji-Akbari, A, Engel, M, Keys, A. S, Zheng, X, Petschek, R. G, Palffy-Muhoray, P, & Glotzer, S. C. (2009) Disordered, quasicrystalline and crystalline phases of densely packed tetrahedra. *Nature* **462**, 773–777.

[88] Jaoshvili, A, Esakia, A, Porrati, M, & Chaikin, P. (2010) Experiments on the random packing of tetrahedral dice. *Physical Review Letters* **c**, 1–4.

[89] Anderson, J. A, Irrgang, M. E, & Glotzer, S. C. (2016) Scalable metropolis monte carlo for simulation of hard shapes. *Computer Physics Communications* **204**, 21–30.

[90] Weeks, J. D, Chandler, D, & Andersen, H. C. (1971) Role of repulsive forces in determining the equilibrium structure of simple liquids. *The Journal of Chemical Physics* **54**, 5237.

[91] Steinhardt, P. J, Nelson, D. R, & Ronchetti, M. (1983) Bond-orientational order in liquids and glasses. *Physical Review B* **28**, 784.

[92] Mann, S. (2008) Life as a nanoscale phenomenon. *Angewandte Chemie International Edition* **47**, 5306–5320.

# APPENDIX A

# Source Code Files for the Thesis

## A.1. Python source code for analyzing DNA linkers

### A.1.1. NADCDRead class for parsing DCD files

```python
from struct import *


"""Class NADCDRead is used to process CHARMM DCD files as produced
by HOOMD molecular dynamics package.
"""

class NADCDRead:
    def __init__(self, file_name):
        """Constructor. Opens DCD file, reads in its header and
        stores it into the class members. No reading of the coordinates it
        perforemd at this stage. Actual reading of the timesteps is handled
        by separate methods."""
        self.file_name = file_name
        self.dcd = open(file_name, 'rb')
        data = self.dcd.read(2*4)           # skip first 8 = 2*4 bytes
        data = self.dcd.read(4)             # read in number of frames
        self.nframes = unpack('I',data)[0]  # and unpack it
        data = self.dcd.read(4)             # starting frame
        self.tstart = unpack('I',data)[0]   # and unpack it
        data = self.dcd.read(4)             # time between frames
        self.tperiod = unpack('I',data)[0]  # and unpack it
        data = self.dcd.read(4)             # total number of steps
        self.nsteps = unpack('I',data)[0]   # and unpack it
        data = self.dcd.read(19*4)          # skip next 76 = 19*4 bytes
        data = self.dcd.read(80)            # Read title string
        self.title = data.strip()           # and get rid of the junk
        data = self.dcd.read(80)            # Read remarks string
        self.remark = data.strip()          # and get rid of the junk
        data = self.dcd.read(2*4)           # Skip 8 = 2*4 bytes
        data = self.dcd.read(4)             # Read number of particles
        self.nbeads = unpack('I',data)[0]   # and unpack it
        data = self.dcd.read(4)             # skip 4 bytes


    def __read_frame_header(self):
        """Read header for each frame."""
        try:
          data = self.dcd.read(4)           # skip 4 bytes
        except:
          return None
        data = self.dcd.read(12*4)          # read in box size 48=12*4
 ↪  bytes
        try:
          box = unpack('6d', data)          # unpack it
        except:
          return None
        data = self.dcd.read(4)             # skip 4 bytes
        return box

    def __read_frame_data(self):
        """Read frame coordinates."""
        data = self.dcd.read(4)             # this is 4 times nparticle -->
 ↪  x
        data_size = unpack('I',data)        # and unpack it
        data = self.dcd.read(data_size[0])  # read in x coordinates
        x = list(unpack(str(self.nbeads)+'f',data))     # an store them into
 ↪  x
        data = self.dcd.read(4)             # skip 4 bytes
        data = self.dcd.read(4)             # this is 4 times nparticle -->
 ↪  y
        data_size = unpack('I',data)        # and unpack it
        data = self.dcd.read(data_size[0])  # read in y coordinates
        y = list(unpack(str(self.nbeads)+'f',data))     # an store them into
 ↪  y
        data = self.dcd.read(4)             # skip 4 bytes
        data = self.dcd.read(4)             # this is 4 times nparticle -->
 ↪  z
        data_size = unpack('I',data)        # and unpack it
        data = self.dcd.read(data_size[0])  # read in z coordinates
        z = list(unpack(str(self.nbeads)+'f',data))     # an store them into
↪   z
        data = self.dcd.read(4)               # skip 4 bytes
        return [x, y, z]

    def box_size(self):
        """Get box size."""
        self.box = self.__read_frame_header()
        if self.box == None:
          print "Ooops!"
          return None
        print 'box size =', self.box
        box = [self.box[0], self.box[2], self.box[5]]
        print 'box = ', box
        return box

    def read_frame(self):
        """Read one time frame."""
        self.box = self.__read_frame_header()
        x, y, z = self.__read_frame_data()
        self.r = []
        for (xx,yy,zz) in zip(x,y,z):
            self.r.append((xx,yy,zz))
        return self.r

    def skip_frame(self, frameNumber):
        """Skip defined number of frames."""
        for i in range(frameNumber):
            temp = self.read_frame()
        print "%d frames skipped!" % frameNumber
```

### A.1.2. An example showing how to use NADCDRead class

```
#!/usr/bin/python
#title           :analyze_dcd_file.py
#description     :get position information from dcd files
#author                 :Saijie Pan
#date            :20140801
#version         :1.0.0
```

```
#usage           :python analyze_dcd_file.py
#notes           :
#python_version  :2.6.6
#================================================================================

# Import the modules to run this script
from NADCDRead2014 import NADCDRead
import sys
import numpy
import os

def read_dcd(filename):
    """ function to get information from one dcd file"""

    # Get input dcd file
    inputFile = filename
    dcdRead = NADCDRead(inputFile)

    # Print basic information about the dcd file
    print 'Frames: ', dcdRead.nframes
    print 'Beads: ', dcdRead.nbeads

    # Skip the first few frames only keep last 200 frames
    dcdRead.skip_frame(dcdRead.nframes-200)

    # create a list of list, which contains trajectories of each particle
    trajectory_list = [[] for particle in range(dcdRead.nbeads)]
    for i in range(200):
        one_frame = dcdRead.read_frame()
        for index in range(dcdRead.nbeads):
            coordinate = one_frame[index]
            # adjust coordinate according to boundary condition
            if coordinate[0] >= 40.0:
                x = coordinate[0] - 93.0
            else:
                x = coordinate[0]
            if coordinate[1] >= 40.0:
                y = coordinate[1] - 93.0
            else:
                y = coordinate[1]
            z = coordinate[2]
            coordinate = (x, y, z)
```

```python

                trajectory_list[index].append(coordinate)

    attached_particle_index = []
    for index in range(dcdRead.nbeads):
        one_trajectory = numpy.array(trajectory_list[index])
        coordinate_average = numpy.mean(one_trajectory, axis=0)
        coordinate_std = numpy.std(one_trajectory, axis=0)
        z_std = coordinate_std[2]

        if z_std <= 2.0:
            attached_particle_index.append(i)
            print coordinate_average, coordinate_std
    print "%d particles attached." % len(attached_particle_index)

def main():
    """ main function """
    dcdFiles = [file for file in os.listdir('.') if file.endswith('.dcd')]
    for file in dcdFiles:
        print '-----------------------------------------------'
        print file
        read_dcd(file)


if __name__ == '__main__':
    main()
```

## A.2. C source code for Monte Carlo simulations of epitaxial growth on a BCC(100) lattice

```c
#include <gtk/gtk.h>
#include <unistd.h>
#include <pthread.h>
#include <signal.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdlib.h>
```

```
#include <math.h>
#include <string.h>
#include <time.h>
#define PI 3.141592653589793
#define GRIDSIZE 200
double COUPLING = 1;
#define WINDOW_SIZE 800
#define MICROSLEEP 0
#define ENERGY0 @E0     //-5
#define ENERGY1 @E1     //-4.8
#define ENERGY2 0
#define T       1.0      //k = 1; temperature


double DIAMETER=0.01;

double total_energy=0;
short int spins[GRIDSIZE*GRIDSIZE];
double boltzmann_factors[5];

#include "mt19937ar.c"
/*
    Mersenne Twister -- A very fast random number generator
    Download at

 ↪  http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/MT2002/CODES/mt19937ar.c
*/

//compile with
//gcc ising.c -lm -o ising `pkg-config --cflags --libs cairo gtk+-2.0` -O3


int pid=0;
int isparent=0;
double *drawparams;

static GdkPixmap *pixmap = NULL;
gboolean on_window_configure_event(GtkWidget * da,
        GdkEventConfigure * event, gpointer user_data){
    static int oldw = 0;
    static int oldh = 0;
    //make our selves a properly sized pixmap if our window has been resized
    if (oldw != event->width || oldh != event->height){
        //create our new pixmap with the correct size.
        GdkPixmap *tmppixmap = gdk_pixmap_new(da->window, event->width,
                event->height, -1);
        //copy the contents of the old pixmap to the new pixmap.
        //This keeps ugly uninitialized
        //pixmaps from being painted upon resize
        int minw = oldw, minh = oldh;
        if( event->width < minw ){ minw =  event->width; }
        if( event->height < minh ){ minh =  event->height; }
        gdk_draw_drawable(tmppixmap, da->style->fg_gc[GTK_WIDGET_STATE(da)],
                pixmap, 0, 0, 0, 0, minw, minh);
        //we're done with our old pixmap,
        //so we can get rid of it and replace it with our properly-sized one.
        g_object_unref(pixmap);
        pixmap = tmppixmap;
    }
    oldw = event->width;
    oldh = event->height;
    return TRUE;
}
gboolean on_window_expose_event(GtkWidget * da, GdkEventExpose * event,
        gpointer user_data){
    gdk_draw_drawable(da->window,
        da->style->fg_gc[GTK_WIDGET_STATE(da)], pixmap,
        // Only copy the area that was exposed.
        event->area.x, event->area.y,
        event->area.x, event->area.y,
        event->area.width, event->area.height);
    return TRUE;
}
static int currently_drawing = 0;
void *do_draw(void *ptr){

    //prepare to trap our SIGALRM so we can draw when we recieve it!
    static int no_update=0;
    siginfo_t info;
    sigset_t sigset;

    sigemptyset(&sigset);
    sigaddset(&sigset, SIGALRM);


        //wait for our SIGALRM. Upon receipt, draw our stuff.
            //Then, do it again!
        while (sigwaitinfo(&sigset, &info) > 0) {

            currently_drawing = 1;

            int width, height;
            gdk_threads_enter();
            gdk_drawable_get_size(pixmap, &width, &height);
            //printf("width = %d\n",width);
            gdk_threads_leave();

            //create a gtk-independant surface to draw on
            cairo_surface_t *cst = cairo_image_surface_create(
                    CAIRO_FORMAT_ARGB32, width, height);
            cairo_t *cr = cairo_create(cst);

            //do some time-consuming drawing
            static int i = 0;
            ++i; i = i % 300;   //give a little movement to our animation
            cairo_set_source_rgb (cr, 0.9,0.9,0.9);
            cairo_paint(cr);

                double *spin  =&drawparams[0];
            double *redraw = &drawparams[GRIDSIZE*GRIDSIZE+2];
            double size = width;
            if (width>height) size = height;
            double dx = (width-size)/2;
            double dy = (height-size)/2;
            int j,k;
            for (j =0;j<GRIDSIZE;j++)
            for (k =0;k<GRIDSIZE;k++)
            {
                if (drawparams[j*GRIDSIZE+k]!=1)
                {
                    if ((j%2==1)&&(k%2==1))
```

```
                        {
                            cairo_arc(cr,dx+size/GRIDSIZE*(j+0.5),
                            dy+size/GRIDSIZE*(k+0.5),
                            size/GRIDSIZE*0.2,0,2 * M_PI);
                            //size changed from 0.2
                            cairo_set_source_rgb (cr, .2, .2, .2);
                            cairo_fill(cr);
                        }
                    }

                }
                for (j =0;j<GRIDSIZE;j++)
                for (k =0;k<GRIDSIZE;k++)
                {
                    if (drawparams[j*GRIDSIZE+k]==1)
                    {
                        cairo_arc(cr,dx+size/GRIDSIZE*(j+0.5),
                        dy+size/GRIDSIZE*(k+0.5),size/GRIDSIZE*0.9,0,2 * M_PI);
                        // size changed from 0.9
                        if ((j%2==0)&&(k%2==0))
                                cairo_set_source_rgb (cr, .2, .8, .2);
                        if ((j%2==1)&&(k%2==0))
                                cairo_set_source_rgb (cr, .8, .2, .2);
                        if ((j%2==0)&&(k%2==1))
                                cairo_set_source_rgb (cr, .8, .2, .2);
                        if ((j%2==1)&&(k%2==1))
                                cairo_set_source_rgb (cr, .2, .2, .8);
                        cairo_fill(cr);
                    }

                }
                cairo_destroy(cr);
                gdk_threads_enter();
                cairo_t *cr_pixmap = gdk_cairo_create(pixmap);
                cairo_set_source_surface (cr_pixmap, cst, 0, 0);
                cairo_paint(cr_pixmap);
                cairo_destroy(cr_pixmap);

                gdk_threads_leave();

                cairo_surface_destroy(cst);

                currently_drawing = 0;
                if (*redraw==1) no_update+=1;
                if (no_update==3000) {exit(0);}
                *redraw=1;
            }
}

gboolean timer_exe(GtkWidget * window){

            static int first_time = 1;

            //use a safe function to get the value of currently_drawing so
            //we don't run into the usual multithreading issues

            //if this is the first time, create the drawing thread
            static pthread_t thread_info;
            if(first_time == 1){
                int  iret;
                iret = pthread_create( &thread_info, NULL, do_draw, NULL);
            }
        do
        {
            int drawing_status = g_atomic_int_get(&currently_drawing);
            //if we are not currently drawing anything, send a SIGALRM signal
            //to our thread and tell it to update our pixmap
            if(drawing_status == 0){
                pthread_kill(thread_info, SIGALRM);
            }

            //tell our window it is time to draw our animation.
            int width, height;
            gdk_drawable_get_size(pixmap, &width, &height);
            gtk_widget_queue_draw_area(window, 0, 0, width, height);


            first_time = 0;

        } while(1==0);

        return TRUE;

}
```

```

static gboolean clicked(GtkWidget * widget, GdkEventButton *event,
        gpointer user_data)
{
        double *redraw = &drawparams[GRIDSIZE*GRIDSIZE+2];
        if (event->button ==1) *redraw=2;
        if (event->button ==2) exit(0);
        if (event->button ==3) *redraw=3;

}

void create_output_window(int argc, char *argv[])
{
  //Block SIGALRM in the main thread
    sigset_t sigset;
    sigemptyset(&sigset);
    sigaddset(&sigset, SIGALRM);
    pthread_sigmask(SIG_BLOCK, &sigset, NULL);

    //we need to initialize all these functions so that gtk knows
    //to be thread-aware
    if (!g_thread_supported ()){ g_thread_init(NULL); }
    gdk_threads_init();
    gdk_threads_enter();

    gtk_init(&argc,&argv);

    GtkWidget *window = gtk_window_new (GTK_WINDOW_TOPLEVEL);
    gtk_widget_add_events(window,GDK_BUTTON_PRESS_MASK);
    g_signal_connect(G_OBJECT(window), "destroy",
            G_CALLBACK(gtk_main_quit), NULL);
    g_signal_connect(G_OBJECT(window), "expose_event",
            G_CALLBACK(on_window_expose_event), NULL);
    g_signal_connect(G_OBJECT(window), "configure_event",
            G_CALLBACK(on_window_configure_event), NULL);
    g_signal_connect(G_OBJECT(window), "button-press-event",
            G_CALLBACK(clicked), NULL);


    //this must be done before we define our pixmap so that it can
↪   reference
    //the colour depth and such
```

```
     gtk_window_set_position(GTK_WINDOW(window), GTK_WIN_POS_CENTER);
   gtk_window_set_default_size(GTK_WINDOW(window),
↪  WINDOW_SIZE,WINDOW_SIZE);

    gtk_widget_show_all(window);

    //set up our pixmap so it is ready for drawing
    pixmap = gdk_pixmap_new(window->window,WINDOW_SIZE,WINDOW_SIZE,-1);
    //because we will be painting our pixmap manually during expose events
    //we can turn off gtk's automatic painting and double buffering
↪  routines
    gtk_widget_set_app_paintable(window, TRUE);
    gtk_widget_set_double_buffered(window, FALSE);

    (void)g_timeout_add(33, (GSourceFunc)timer_exe, window);
      gtk_window_set_position(GTK_WINDOW(window), GTK_WIN_POS_CENTER);


    gtk_main();
    gdk_threads_leave();

}
void multiprocess_start(int argc, char *argv[])
{
    int i;
    int shmid;
    double *shm_ptr;
    isparent = 1;
    pid = getpid();
    key_t key=12345+pid;

    shmid=shmget(key,(2*GRIDSIZE*GRIDSIZE)*sizeof(double),0666|IPC_CREAT);

    if (shmid < 0)
    {
        perror("shmget");
        exit(1);
    }

    shm_ptr=(double *)shmat(shmid,(void *)0,0);

    if (shm_ptr == (double *)(-1))
    {
        perror("shmat:shm_ptr");
        exit(1);
    }
    drawparams =(shm_ptr);
    //*groupline = 1;

    pid = fork();

        if (pid == 0)
        {
            pid = getpid();
            printf("child created: %d\n",pid);
            isparent = 0;
            create_output_window(argc, argv);
            exit(0);
        }


}
void write_locations()
{
    int i;
    double tt =0.5;
    double *spincolors  =&drawparams[0];
    double *measurements = &drawparams[GRIDSIZE*GRIDSIZE];
    measurements[0] =total_energy;
    for (i=0;i<GRIDSIZE*GRIDSIZE;i++) spincolors[i] = spins[i];


}
double energy_stat =0;

void init()
{
    int i;
    //energy of center-bound sites
    boltzmann_factors[0] = exp(1.*ENERGY0 *COUPLING /T);
    //energy of edge-bound sites
    boltzmann_factors[1] = exp(1.*ENERGY1 *COUPLING /T);
    //energy of corner-bound sites
    boltzmann_factors[2] = exp(1.*ENERGY2 *COUPLING /T);
    for (i=0;i<GRIDSIZE*GRIDSIZE;i++) spins[i]=-1;
        //for (i=0;i<GRIDSIZE*48;i++) spins[i*2] = 1;
}

void combination()  //include both diffuse and absorption and disorption
{
    double bf, bf2;
    int k = genrand_real2()*GRIDSIZE*GRIDSIZE;
    int i = k/GRIDSIZE;
    int j = k-i*GRIDSIZE;
    int di, dj;
    int spin = spins[k];
    inline int get_state(int i, int j)
    {
        return spins[((i+GRIDSIZE)%GRIDSIZE)*GRIDSIZE
                +((j+GRIDSIZE)%GRIDSIZE)];
    }

    if (spin == -1)
    //spin == -1 : site unoccupied or spin == 1 : site occupied
    {
        if ((get_state(i-1,j-1)==1)||(get_state(i-1,j)==1)||
                (get_state(i-1,j+1)==1)||
            (get_state(i+1,j-1)==1)||(get_state(i+1,j)==1)||
            (get_state(i+1,j+1)==1)||
            (get_state(i,j-1)==1)||(get_state(i,j+1)==1))
            return; // neighbouring occupied, stay unoccupied due to block

        spins[k]=1;     //else switch to occupied
                return;
    }
    else
    {
        //change current site to unoccupied, if no move, change back
        spins[k] = -1;
        if ((i%2==0)&&(j%2==0)) bf = boltzmann_factors[0];   //center sites
        else if ((i%2==0)&&(j%2==1)) bf = boltzmann_factors[1]; //edge
        else if ((i%2==1)&&(j%2==0)) bf = boltzmann_factors[1];
        else if ((i%2==1)&&(j%2==1)) bf = boltzmann_factors[2]; //corner
                double rand_0_1 = genrand_real2();
                if (rand_0_1 < bf)
```

```
                    //particle disorption
                    //spins[k]=-1 is already set, so just return
                    return;
                //no disorption, what about diffusion
        int direction = genrand_real2()*4;
        di = i;
        dj = j;
        if (direction == 0) di = i+1;
        else if (direction == 1) di = i-1;
        else if (direction == 2) dj = j+1;
        else if (direction == 3) dj = j-1;
        inline int get_state(int i, int j)
        {
            return spins[((i+GRIDSIZE)%GRIDSIZE)*GRIDSIZE
                    +((j+GRIDSIZE)%GRIDSIZE)] ;
        }
        if ((get_state(di-1,dj-1)==1)||(get_state(di-1,dj)==1)||
                (get_state(di-1,dj+1)==1)||
            (get_state(di+1,dj-1)==1)||(get_state(di+1,dj)==1)||
            (get_state(di+1,dj+1)==1)||
            (get_state(di,dj-1)==1)||(get_state(di,dj+1)==1))
            {spins[k] = 1; return;}
            // destination not permitted, stay unchanged
        else
        {
            if ((di%2==0)&&(dj%2==0)) bf2 = boltzmann_factors[0];
            //center sites
            else if ((di%2==0)&&(dj%2==1)) bf2 = boltzmann_factors[1];
            //edge
            else if ((di%2==1)&&(dj%2==0)) bf2 = boltzmann_factors[1];
            else if ((di%2==1)&&(dj%2==1)) bf2 = boltzmann_factors[2];
            //corner
                        if (rand_0_1<bf/bf2)
                {
                    di = ((di+GRIDSIZE)%GRIDSIZE);
                    dj = ((dj+GRIDSIZE)%GRIDSIZE);
                    spins[di*GRIDSIZE + dj] = 1;
                    return;
                }
            else spins[k] = 1;
            //If no diffuse or disorption, stay unchanged
        }
```

```
    }
}


void absorption()
{
    double bf;
    int k = genrand_real2()*GRIDSIZE*GRIDSIZE;
    int i = k/GRIDSIZE;
    int j = k-i*GRIDSIZE;
    int spin = spins[k];
    inline int get_state(int i, int j)
    {
        return spins[((i+GRIDSIZE)%GRIDSIZE)*GRIDSIZE
                ((j+GRIDSIZE)%GRIDSIZE)];
    }
    if (spin == -1)
    //spin == -1 : site unoccupied or spin == 1 : site occupied
    {
        if ((get_state(i-1,j-1)==1)||(get_state(i-1,j)==1)||
                (get_state(i-1,j+1)==1)||
            (get_state(i+1,j-1)==1)||(get_state(i+1,j)==1)||
            (get_state(i+1,j+1)==1)||
            (get_state(i,j-1)==1)||(get_state(i,j+1)==1))
            return;   //neighbouring occupied, stay unoccupied due to block
        spins[k]=1;               //else switch to occupied
    }   else
    {
        if ((i%2==0)&&(j%2==0)) bf = boltzmann_factors[0];  //center sites
        if ((i%2==0)&&(j%2==1)) bf = boltzmann_factors[1];  //edge sites
        if ((i%2==1)&&(j%2==0)) bf = boltzmann_factors[1];
        if ((i%2==1)&&(j%2==1)) bf = boltzmann_factors[2];  //corner sites
        if (genrand_real2()<bf) spins[k]=-1;
    }
}

void diffusion() // move to neighbouring sites
{
    double bf, bf2;
    int k = genrand_real2()*GRIDSIZE*GRIDSIZE;
    int i = k/GRIDSIZE;
    int j = k-i*GRIDSIZE;
    int di, dj;
    int spin = spins[k];
    inline int get_state(int i, int j)
    {
        return spins[((i+GRIDSIZE)%GRIDSIZE)*GRIDSIZE
                +((j+GRIDSIZE)%GRIDSIZE)];
    }

    if (spin == -1)
            //spin == -1 : site unoccupied or spin == 1 : site occupied
        return;
    else
    {
        spins[k] = -1;
        //change current site to unoccupied, if no move, change back
        int direction = genrand_real2()*4;
        di = i;
        dj = j;
        if (direction == 0) di = i+1;
        else if (direction == 1) di = i-1;
        else if (direction == 2) dj = j+1;
        else if (direction == 3) dj = j-1;
        inline int get_state(int i, int j)
        {
            return spins[((i+GRIDSIZE)%GRIDSIZE)*GRIDSIZE
                    +((j+GRIDSIZE)%GRIDSIZE)];
        }
        if ((get_state(di-1,dj-1)==1)||(get_state(di-1,dj)==1)||
                (get_state(di-1,dj+1)==1)||
            (get_state(di+1,dj-1)==1)||(get_state(di+1,dj)==1)||
            (get_state(di+1,dj+1)==1)||
            (get_state(di,dj-1)==1)||(get_state(di,dj+1)==1))
            {spins[k] = 1; return;}
            // destination not permitted, stay unchanged
        else
        {
            if ((i%2==0)&&(j%2==0)) bf = boltzmann_factors[0];//center
↪   sites
            else if ((i%2==0)&&(j%2==1)) bf = boltzmann_factors[1];
            //edge
            else if ((i%2==1)&&(j%2==0)) bf = boltzmann_factors[1];
            else if ((i%2==1)&&(j%2==1)) bf = boltzmann_factors[2];
```

```
                //corner
                if ((di%2==0)&&(dj%2==0)) bf2 = boltzmann_factors[0];
                //center
                else if ((di%2==0)&&(dj%2==1)) bf2 = boltzmann_factors[1];
                //edge
                else if ((di%2==1)&&(dj%2==0)) bf2 = boltzmann_factors[1];
                else if ((di%2==1)&&(dj%2==1)) bf2 = boltzmann_factors[2];
                //corner
                if (genrand_real2()<bf/bf2)
                    {
                        di = ((di+GRIDSIZE)%GRIDSIZE);
                        dj = ((dj+GRIDSIZE)%GRIDSIZE);
                        spins[di*GRIDSIZE + dj] = 1;
                        return;
                    }
                else spins[k] = 1;          //If no move, stay unchanged
            }
        }
}

double* print_data()
{
        int sites[4] = { 0 };
        //number of different sites: center, edge, corner and vacancy
        int total_site = (GRIDSIZE/2)*(GRIDSIZE/2);
        //double sites_percent[4];
        double *sites_percent = malloc(sizeof(double) * 4);
        int i,j,k;
        int spin;
    inline int get_state(int i, int j)
    {
        return spins[((i+GRIDSIZE)%GRIDSIZE)*GRIDSIZE
                +((j+GRIDSIZE)%GRIDSIZE)];
    }
        for (i=0; i<GRIDSIZE; i++)
                for (j=0; j<GRIDSIZE; j++)
                {
                        spin = spins[GRIDSIZE*i+j];
                        if (spin == 1)
                        {
                                if ((i%2==0)&&(j%2==0)) sites[0] += 1;
                                else if ((i%2==0)&&(j%2==1)) sites[1] += 1;
                                else if ((i%2==1)&&(j%2==0)) sites[1] += 1;
                                else if ((i%2==1)&&(j%2==1)) sites[2] += 1;
                            }
                            else
                        {
                            continue;
                            //vacancy not correct;
                            //first the total is not always 100% in theory;
                            //second, count more vacancy when a line of open
   ↪  sites
                    if ((get_state(i-1,j-1)==1)||(get_state(i-1,j)==1)||
                            (get_state(i-1,j+1)==1)||
                        (get_state(i+1,j-1)==1)||(get_state(i+1,j)==1)||
                        (get_state(i+1,j+1)==1)||
                        (get_state(i,j-1)==1)||(get_state(i,j+1)==1)) continue;
                            sites[3] += 1;
                            }
                    }
         for (k=0; k<4; k++)
                    sites_percent[k] = sites[k] / (double)(total_site);
         return sites_percent;
}

void edge_defect_length(double *len, double *orientation)
{
        int count = 0;
        int i, j, k;
        int flag1;
        int num_edge_site;
        int num1 = 0, num2 = 0;
    inline int get_state(int i, int j)
    {
        return spins[((i+GRIDSIZE)%GRIDSIZE)*GRIDSIZE
                +((j+GRIDSIZE)%GRIDSIZE)];
    }
        for (j=0; j<GRIDSIZE; j+=2)
        {
            flag1 = 1;
            if (get_state(1, j)==1 && get_state(2*(GRIDSIZE/2)-1, j)==1)
                    count -= 1;
            for (i=1; i<GRIDSIZE; i+=2)
            {
                if (flag1 == 1 && get_state(i, j)== 1)
                {
                    flag1 = 0;
                    count += 1;
                }
                else if (flag1 == 0 && get_state(i, j)== -1)
                    flag1 = 1;
            }
        }
        num1 = count;
        count = 0;
        for (i=0; i<GRIDSIZE; i+=2)
        {
            flag1 = 1;
            if (get_state(i, 1)==1 && get_state(i, 2*(GRIDSIZE/2)-1)==1)
                    count -= 1;
            for (j=1; j<GRIDSIZE; j+=2)
            {
                if (flag1 == 1 && get_state(i, j)== 1)
                {
                    flag1 = 0;
                    count += 1;
                }
                else if (flag1 == 0 && get_state(i, j)== -1)
                    flag1 = 1;
            }
        }
        num2 = count;
        count = num1 + num2;
        if (count == 0)
            *orientation = 0;
        else
            *orientation = abs(num1 - num2)/ (double)count;
        num_edge_site = 0;
        for (i=0; i<GRIDSIZE; i++)
                for (j=0; j<GRIDSIZE; j++)
                {
                        if (spins[GRIDSIZE*i+j] == 1)
                        {
                            if (((i%2==0)&&(j%2==1))||((i%2==1)&&(j%2==0)))
                                    num_edge_site += 1;
                        }
                    }
        if (num_edge_site == 0 && count == 0)
            *len = 0;
        else
            *len = (double)num_edge_site/ count;

}


int main (int argc, char *argv[])
{
    int i,j,k = 0;
    int tt;
        double sum= 0;
        double defect_len, orientation_deg;
    init_genrand(time(0));
    init();
    //multiprocess_start(argc, argv);
    //double *redraw = &drawparams[GRIDSIZE*GRIDSIZE+2];
        FILE *fp = fopen("output3_@E0_@E1_T=1.txt", "w+");
        FILE *fp2 = fopen("output3_@E0_@E1_T=1_all20000_step1000.dat",
 ↪  "wb");
    do
    {
                k += 1;
                edge_defect_length(&defect_len, &orientation_deg);
                fprintf(fp,"%f\t%f\t%f\t%f\t%f\n", print_data()[0],
                        print_data()[1], print_data()[2],
                        defect_len, orientation_deg);
                if (k % 1000 == 0){
                    printf("%d steps finished\n", k);
                    fwrite(spins, 1, sizeof(spins), fp2);
                    //fwrite(spins, sizeof(short int),
                    //sizeof(spins)/sizeof(short int), fp2);
                    }
                if (k >= 20000)
                {
                printf("Output Done!\n");
                fclose(fp);
                fclose(fp2);
                exit(0);
                }
        for (tt=0;tt<1000000;tt++)
        {
                        if (genrand_real2() < 0.5)
                absorption();
                        else
                            diffusion();
                        /*
            if (*redraw==3)
            {
                printf("Coupling:\n");
                j = scanf("%lf", &COUPLING);
            }

            if (*redraw==2) {init();write_locations(); *redraw=0;}
            if (*redraw==1) {write_locations(); *redraw=0;}
                        */
        }
        //usleep(0);

    }   while TRUE;
}
```