# Development and Analysis of a Gravity-Simulated Particle-Packing Algorithm for Modeling Optimized Rocket Propellants

**Mark Stockmyer**

Submitted to the Department of Electrical Engineering and Computer Science and the Faculty of the Graduate School of the University of Kansas in partial fulfillment of the requirements for the degree of Master's of Science

October 2007

Committee Members: **Hossein Saiedian, Ph.D.**
Professor and Thesis Adviser

**Arvin Agah, Ph.D.,**
Associate Professor

**Xue-wen Chen, Ph.D.**
Assistant Professor

**Date Defended: October 5, 2007**

Approved for public release; distribution is unlimited.

The thesis committee for Mark Stockmyer certifies that this is the approved version of the following thesis:

# Development and Analysis of a Gravity-Simulated Particle-Packing Algorithm for Modeling Optimized Rocket Propellants

Committee Members:

**Hossein Saiedian, Ph.D.**
Professor and Thesis Adviser

**Arvin Agah, Ph.D.,**
Associate Professor

**Xue-wen Chen, Ph.D.**
Assistant Professor

**Date Approved:** ____________

# Abstract

The random heterogeneous morphology of modern solid rocket propellant formulations has traditionally been difficult to characterize and quantify. Current computational simulations of these formulations require an accurate description of the packing arrangement in order to correctly model the complex geometric effects that stem from the random morphology. A new and novel computational packing algorithm was invented, implemented, and analyzed using various particle starting arrangements. This was intended to be fast for use in combinatorial chemistry applications and to provide a numerical representation of the material for use with other computational tools, including codes that predict combustion behavior. The packing algorithms were evaluated using homogeneous distributions of spherical particles. Both the Radial Distribution Function (RDF) and the packing fraction were used to evaluate the validity of the invented algorithm.

# Acknowledgements

I dedicate this thesis to my wife, Kristina Stockmyer, without whom I never would have attempted an effort of this magnitude. Her support and cheering from the sidelines was requisite. I never would have finished the program without her.

Also on the list of those to thank is my advisor, Dr. Hossein Saiedian. Without his advising, suggesting, and cajoling, not to mention his many reassurances, this research probably would have been put off indefinitely.

My parents, John and Connie Stockmyer, also bear some responsibility here, as they provided a learning environment up in which to grow. I especially wish to acknowledge my mother whom, despite my repeated protestations that I didn't want to learn "all those languages," kept suggesting that I get into computers.

Special recognition needs to be granted to Dr. Travis Laker. He started this research before I arrived at China Lake, and didn't complain when I took the project in an entirely different direction. Dr. Laker is responsible for the three-dimensional visualization tool.

Financial support for the early stages of this research were provided by the Naval Air Warfare Center Weapons Division (NAWCWD) In-house Laboratory Independent Research (ILIR) program through the Office of Naval Research (ONR). I would like to thank the program for the opportunity to work on such a challenging project.

For all of you who have helped along the way, you have my permission to stop here. Everybody else needs to keep reading.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Rocketry traces its earliest roots to 10th century (A.D.) China where primitive fire-arrows (Ling 1947) were used in fighting between rival factions. In those times, simply moving a weapon through the air from point A to point B was considered a success. The fuel powering one of these projectiles had to have two key properties: it produced the thrust required and it did not destroy the flying object before it reached the intended target. While these properties are still needed in modern propellants, the precision munitions of today must meet more specific tolerances and other optimized properties.

This research aims to find computational means of finding the optimal mix of chemical reactants to form rocket propellants. Properties that can be optimized include: increasing power, optimizing burn rates, reducing unwanted burn products, and reducing shock sensitivity. Specifically, this research will focus on techniques for modeling the **structure** of a propellant, as the arrangement of the individual molecules that make up formulations of propellant have a significant bearing on it performance and characteristics.

## 1.1 Justification

Currently, a research chemist invents a new theoretical mix of chemicals, guessing what may provide superior properties to past mixes. Computer modelers take that fixed set of inputs and try to determine, using computational tools, what the properties of that new mixture would be. Several computer models would be run and, if the research chemists accepted the results of the computations, actual mixes would be produced and tested to verify or contradict the prediction of the model.

Computing power has been constantly increasing and has recently reached the point at which most, but not all, single-iteration fixed input problems can be solved to an acceptable degree. A recent phenomena is the emerging research area known as combinatorial computational techniques (Furka 1995), essentially moving from solving single-input to varied-input problems. Ranges of input can be tested to find an optimal solution. This technique is being used heavily in the pharmaceutical industry to develop new medications.

There are several techniques for modeling a propellant. One method is known as Propellant Equilibrium Programs, or PEP. Various implementations of PEP algorithms exist, but they all act more or less the same. All of the reactants in a propellant mix in specific amounts are entered with their known heats of formation. The reactants are then virtually deconstructed (simulating combustion) and, using the known heats of formation of the possible products, an estimation of the amounts and types of different products formed is determined. PEP programs are widely used but typically only as a starting point for propellant discovery because other important properties are missed by the PEP algorithms.

## 1.2 Significance and Expected Contributions

The speed at which a propellant burns is directly related to its internal structure (Knott, Jackson & Buckmaster 2001). Researchers don't need to model the closest packing structure of the particles in question, although even modeling that in random simulations has proven difficult (Aste & Weaire 2000). Determining the closest packing structure is immaterial as studies show that chemical mixtures do not approach packing perfection. Indeed, what is needed is a way to model realistically, or realistically *enough*, the packing of particles in a system. Kepler's conjecture states that the packing fraction (space of the particles compared to the empty space in a fixed volume) of a system consisting of single diameter hard spheres is maximally $\pi/\sqrt{18} \approx 0.74$ (Hales 2005).

Results shown by Kilgore & Scott (1969) indicate that the highest packing fraction experimentally obtainable is $.6366 \pm .0005$. It is clear that there is a large difference between the theoretical and the practical.

The approaches used most frequently in current research are the kinematic models. In this approach, particles are modeled as hard spheres (Lubachevsky 1991) and thrown into a mirror-sided box. The particles are given an initial velocity in a random direction and the collisions are modeled until the entire system settles into a steady state or a jammed system is detected. These models, while fairly accurate, are more computationally expensive due to modeling of the tremendous number of collisions that must be processed as the model progresses. These approaches are not fast enough for what is currently required by a high-speed combinatorial chemistry system. Many approaches like this require large clusters of computers and complex systems such as those described by Bagrodia, Chandy & Liao (1991). Indeed, a fast algorithm could be run on single processor systems having very little system complexity. Kinematic models also must take into account motion-based issues like many of those summarized in Agarwal, Guibas, Edelsbrunner, Erickson, Isard, Har-Peled, Hershberger, Jensen, Kavraki, Koehl, Lin, Manocha, Metaxas, Mirtich, Mount, Muthukrishnan, Pai, Sacks, Snoeyink, Suri & Wolefson (2002).

The goal of this research is to devise a particle-packing algorithm that is both computationally fast and sufficiently correct in modeling propellant particle systems. The scope of this work will be restricted to mono-modal (particles of only one size) systems, but further work should include more realistic multi-modal particle packs. Also of interest in future research would be verification of the model similar to the methodology described by Balci (2001) in which common metrics are defined and subject matter experts are employed to determine the quality of a particular model.

## 1.3 Research Methodology

The approach taken in this research attempts to develop highly efficient packing models and to avoid the computationally expensive kinematic calculations. Torquato (2002) states that computer models of packing structures are difficult because the results tend to be "protocol dependent." This is true if a unified modeling algorithm is used for packing generic particles. This research's goal is to find a way to model propellants with a relatively small set of candidate particles. All computer modeling to this date, and into the foreseeable future, attempts to reflect reality which falls short of perfection. The aim is to take advantage of this imperfection, because perfection need not exist in this range of applications.

The author has invented a gravity-simulation algorithm that will be exercised in multiple ways. There are two parts to the algorithm. The first involves the pattern in which the particles are suspended above the control volume. Four different arrangements will be tested: Single Column, Small Dense, Large Dense, and Loose Random. Once the particles are in their starting positions, the second part, the gravity simulation, will take effect. It involves moving particles downward in space until they contact the floor, wall, or another particle. If a floor or wall is contacted, that particle is fixed in place and the next particle is selected. If the **falling** particle contacts another particle in the pack, the Spin Gap Move Protocol (SGMP) begins. The SGMP starts by doing a spherical sweep of the falling particle around the contacted particle to see if a non-contact area can be found. If a sufficiently large open space can be found, the falling particle is moved in that direction, and the downward movement begins again. This action simulates the falling and rolling motion of particles onto a complex surface of already packed particles.

## 1.4 Evaluation Criteria

The quality of a packing algorithm is evaluated by measuring several properties: packing density, randomness, average coordination number, speed of algorithm, and scalability. The first three regard the "correctness" of the algorithm whereas the latter two merely reflect how effective it will be in combinatorial chemistry applications.

A “correct” algorithm will generate a particle pack that reflects reality. The packing density is a measure of the total volume of the particles in a chamber divided by the chamber volume. The density can vary depending on the modal distribution of the particles in the pack, but for mono-modal packs, the target density for this research is in the range of 63%; anything less is not realistic enough.

Randomness is a much more difficult value to measure. A relatively easy way to measure randomness utilizes a statistical pairwise correlation function called the Radial Distribution Function, or RDF. The RDF measures the correlation from one particle to all the other particles. At distances close to the particle, the RDF will be relatively high. As the distance from the particle gets larger, the RDF will approach 1, indicating that there is no correlation (no predictable pattern) between particles.

Coordination number is a count of how many other particles any specific particle is in contact with. The average coordination number is simply the average of all of the coordination numbers for each particle in the finished pack. At this time, the SGMP algorithm does not have any method for bringing particles in contact with each other. They will be very close to each other, but there will be a measurable distance between all of the particles. This is a deficiency that will be addressed in future research. In essence, the average coordination number will be zero.

The direct goal of this research is to find a packing algorithm that is fast enough to run thousands to millions of runs in a calendar day. According to unpublished research by Tom Jackson and his team at the Center for Simulation of Advanced Rockets (CSAR), their best kinematic simulation is able to determine a single 100,000 particle pack in around 100 hours on a 64 processor computer system. This speed is a step forward compared to past systems, but it is clearly not fast enough for combinatorial chemistry.

Also of importance is scalability. If 10,000 particle packs are our goal, then having an excellent result at 1,000 or 2,000 particles, but poor results for anything higher, is unacceptable. Each of the algorithms in this research will be analyzed as to their Big(O) complexity. Also, each will be exercised and timed with different numbers of particles to examine how well they scale up to realistic numbers of particles found in complex systems.

## 1.5 Thesis Organization

The thesis will be organized into the following sections:

- **Chapter 1: Introduction** – The background and justification for the work contained in this thesis.
- **Chapter 2: Previous Work** – A detailed account of previous work in packing algorithms.
- **Chapter 3: Problem Domain and Evaluation Criteria** - A look at the details of the testing framework and how the results will be interpreted.

- **Chapter 4: SGMP: The Spin Gap Move Protocol** – A description of the algorithm invented for this research.
- **Chapter 5: SGMP – Single Column Starting Pack** – The SGMP algorithm applied to a single tall column of particles as a starting arrangement.
- **Chapter 6: SGMP – Small Dense Starting Pack** – The SGMP algorithm applied to a small densely packed starting arrangement of particles.
- **Chapter 7: SGMP – Large Dense Starting Pack** – The SGMP algorithm applied to a large densely packed starting arrangement of particles.
- **Chapter 8: SGMP – Loose Random Starting Pack** – The SGMP algorithm applied to a loose randomly packed starting arrangement of particles.
- **Chapter 9: Comparison and Analysis** – A comparison of the different starting arrangements and the performance of the SGMP in general.
- **Chapter 10: Conclusion and Future Work** – A conclusion regarding the results of this research and possible future work or improvements.

# Chapter 2

# Previous Work

Early historical efforts to quantify the nature of random packing arrangements consisted primarily of empirical experiments using large numbers of spherical particles. McGeary (1961) examined the packing of spheres of various sizes in the early sixties, in which the emphasis was on the determination of the packing fraction, or the amount of the packed volume that is occupied by particles. The experiments were conducted by placing steel shot into 100 mL graduated cylinders using various deposition methods. McGeary concluded that the best method was to place a single layer of particles at a time, followed by mechanical vibration of the system. This produced a "stationary filter bed" upon which the next layer would be deposited. McGeary also found that the final density of the pack was "essentially independent" of the amount of time that it took to form the pack. This study also included investigations of bimodal packs and an identification of the optimal packing ratios.

An early attempt with computer-assisted packing of spheres was done by Bennett (1972). His model starts with a ‘seed’ cluster of three particles formed in the shape of an equilateral triangle. The fixed pack (that starts with the three particles) is scanned to find all of the “pockets” where a new particle might be placed to be in contact with the three fixed particles. A pocket is defined simply as “a point lying exactly one particle diameter ($\sigma$) away from each of three particle centers and at least $\sigma$ away from all of the other particle centers in the cluster.” Bennett had two methods for the selection of which pocket to place the particle. The first, called the “global” method, selected the pocket that was closest to the original center of the cluster. The “local” method selected a pocket “at the site having the least distance from the plane of its three nearest neighbors” (Bennett 1972); in other words, the ‘deepest’ pocket on the fixed cluster. The local method created an egg-shaped pack at a packing density of .57 and was discarded in favor of the global method. The global method created pack had a .61 packing fraction, short of .6366, and its Radial Distribution graph showed nearly as much pattern (non-randomness) as an experimental close pack of ball bearings similar to the work of McGeary (1961).

More recent experimental investigations also have been performed using actual energetic material. Miller (1982) created a number of carefully quantified packs as part of an investigation into the ballistic performance of reduced smoke propellants. These data have been used in a variety of computational efforts as a means to validate and quantify the predicted packing arrangements by Knott et al. (2001) and Wang, Jackson & Massa (2004). Such quantification experiments have also been extended to non-spherical particles (Zou & Yu 1996).

Another attempt at computational simulation of random media was made by Clarke & Wiley (1987). They created a collective rearrangement algorithm that addressed the atomic structure in amorphous metals. It started with randomly distributed particles that were initially allowed to overlap. As the simulation progressed, the particles were systematically rearranged until a dense packing arrangement was achieved. To avoid a “lockup” state prematurely, they introduced a vibration step, in which each particle is moved a short random distance. Interestingly, changing the vibration parameters, like frequency and amplitude, had no effect on the final outcome, only on how long it took to get to the final state. Clarke and Wiley reported packing fractions of 0.64-0.68 for their algorithm. It is this range that is generally accepted as the random packing limit for uniform hard spheres.

Lubachevsky & Stillinger (1990), initially interested in 2-dimensional disks, proposed another simple algorithm that has been used successfully in a variety of applications. Rather than the traditionally sequential algorithm, they proposed an “event-driven” dynamic simulation where hard disks and spherical particles were allowed to interact with each other and collide. Notable elements of their algorithm were: 1) a particle growth function in which the initially randomly seeded particles were allowed to grow at fixed rates, 2) the definition of periodic boundary conditions for the particle simulation domain (ensuring no initial overlap of the disks or spheres and that particle interaction with the boundaries were neglected), and 3) assignment of random velocity vectors that allowed the particles to move through the domain. Contact between particles was determined using a simple functional relationship. Rather than progress through fixed time increments, Lubachevsky (1991) suggested that substantial gains in computational performance could be achieved if the simulation moved from one collision to the next future collision, regardless of the amount of time between collisions. Computational results for the packing fractions were reported to be within the range of 0.63-0.65. The model results in two types of packings: a completely jammed system and a system with a rattler. A rattler is a particle that is completely caged by other particles yet has some degree of freedom to move.

Davis & Carter (1990) performed one of the first applications of random packing simulations to heterogeneous propellants. They used a reduced dimensional algorithm to create their propellant packs. This approach involves particles of different sizes and densities. Particles are randomly selected from a list of particles in the amounts that they would naturally occur in a studied propellant. As a particle is selected, it is randomly placed to be in contact with a vertical line in three-dimensional space. The particle is moved downward until it hits the floor (the first particle) or comes into contact with the previous particle on the line. The discovery made using this technique is the “realization that an arbitrary straight line drawn through a uniformly random pack spends the same fraction of its length inside solid particles as the packing fraction” (Davis & Carter 1990). They would create many of these “strings” of particles and use a perturbation method to bring them together in order to create random packs.

Researchers at the French aerospace laboratory ONERA also generated random propellant packings for use in their propellant combustion models (Groult & Bizot 2004). Groult and Bizot randomly position the spheres’ centers and allow them to grow. When collisions occur between particles, they are allowed to move and reorient themselves. Periodic boundary conditions are used to simplify the computational domain. A secondary growing phase is also performed where non-spherical shape characteristics can be added.

A significant amount of propellant packing and combustion research has been performed by the Center for Simulation of Advanced Rockets (CSAR) at the University of Illinois at Urbana-Champaign (Knott et al. 2001) and (Wang et al. 2004). Researchers at CSAR developed a fully dynamic, kinetics-based particle packing code. The computational packing results show good agreement with the experimental data generated by Miller (1982). Although effective, the algorithm tends to consume a large amount of computational resources for large amounts of time.

The most recent attempt at computerized random packing is a novel concept invented by Shi & Zhang (2006). Their approach involves dropping spheres, one at a time, into a control volume. When a sphere contacts a particle, they have “rolling rules” that simulate the actual physical process of one particle hitting another and being pulled down over the stationary particle by gravity. When the falling particle hits a second stationary particle, different rules take effect and the falling particle rolls down over the two particles as if by gravity. The particle stops and is fixed in place when it comes into contact with three fixed particles and is determined to be in a stable position. Periodic boundaries are used to eliminate edge effects. The results obtained by this method are a packing fraction in the range of .56 to .58, depending on various particle sizing parameters.

# Chapter 3

# Problem Domain and Evaluation Criteria

## 3.1 Particle Testbed

In order to simulate the packing of particles (or, in this case, perfect spheres), a simulated chamber is needed. To simplify calculations later, a three-axis chamber with a unit volume of one is used and will be known as the control volume. The chamber is visualized as a simplified particle system like that described by Ebert (1996). The control volume is the smaller (inside) cube shown in Figure 3.1. Each side of the chamber has a length of one with the origin of each axis being at the center of each side. This puts the origin (0,0,0) of the X, Y, Z coordinate system at the center of the chamber. The outer cube in Figure 3.1 is the expanded volume and is 1.25 units per side. Its use will be discussed in Section 3.2. As seen in the following figures, the X-Axis is left-to-right (Figure 3.2), the Z-Axis is front-to-back (Figure 3.4), and the Y-Axis is top-to-bottom (Figure 3.3).

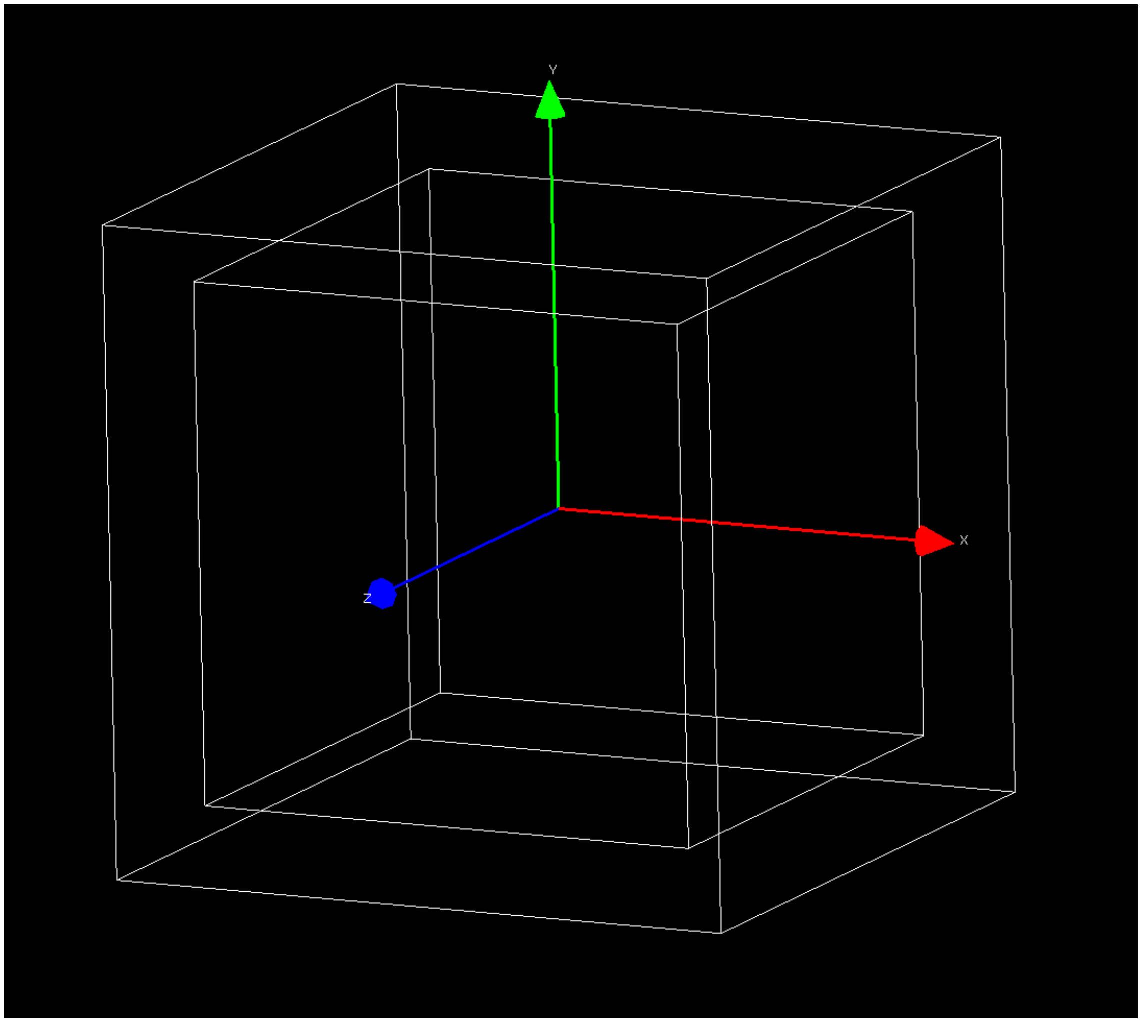


Figure 3.1: Control Volume and Expanded Volume Bounding Boxes

Each particle has three properties: center position, diameter, and color. The color is for visual differentiation and has no bearing on the particle itself.

The algorithm requires that the particles not overlap in space, so there needs to be a method of detecting if a particle is overlapping; in other words, if one particle has collided with another particle. In simple terms, a particle is said to

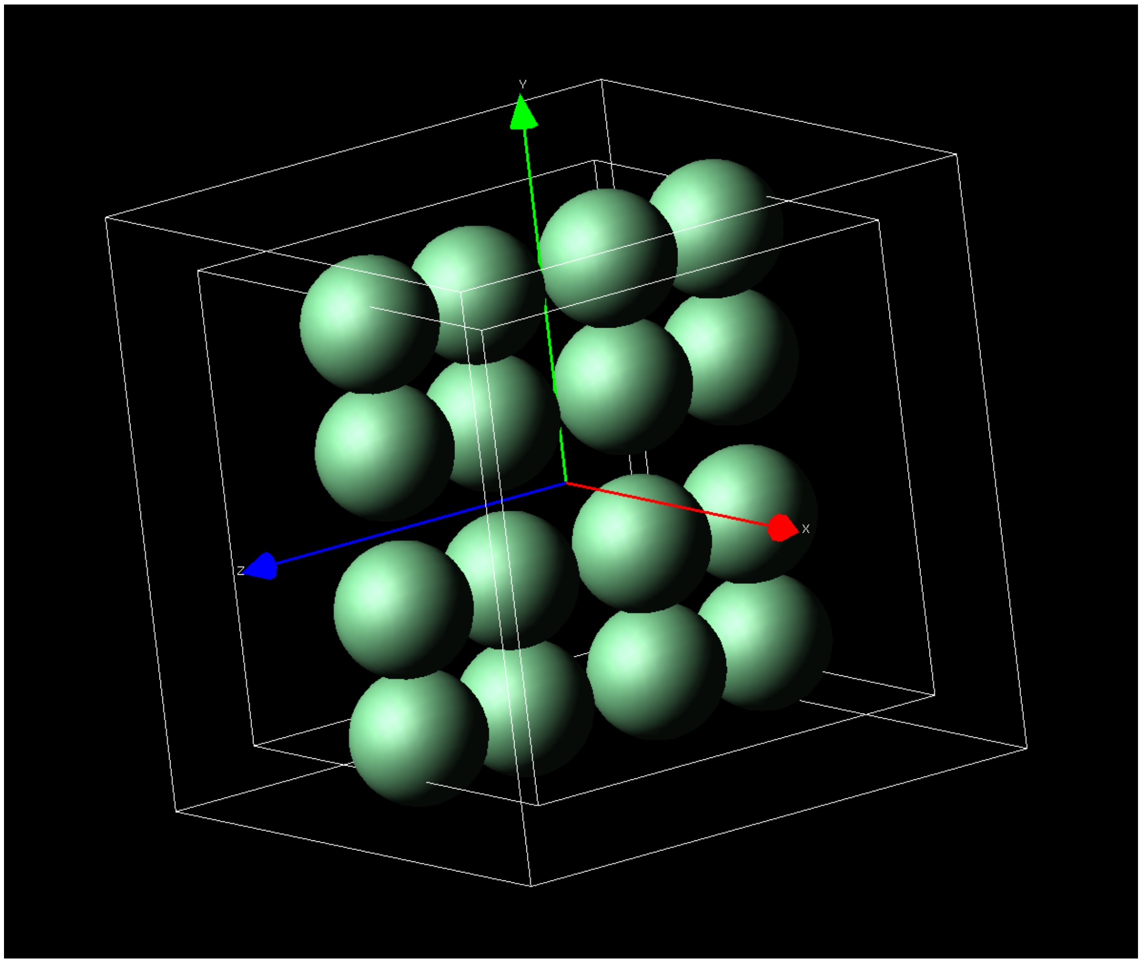


Figure 3.2: Particles Along X-Axis (X = 0)

have collided with another if Equation 3.1 is true. The sum of their radii is less than the distance between the particles' centers.

$$r_1 + r_2 < Distance_{r_1,r_2} \tag{3.1}$$

The sum of the particles' radii is easily calculable. The distance between the particles' centers can be found with Equation 3.2, a three-dimensional

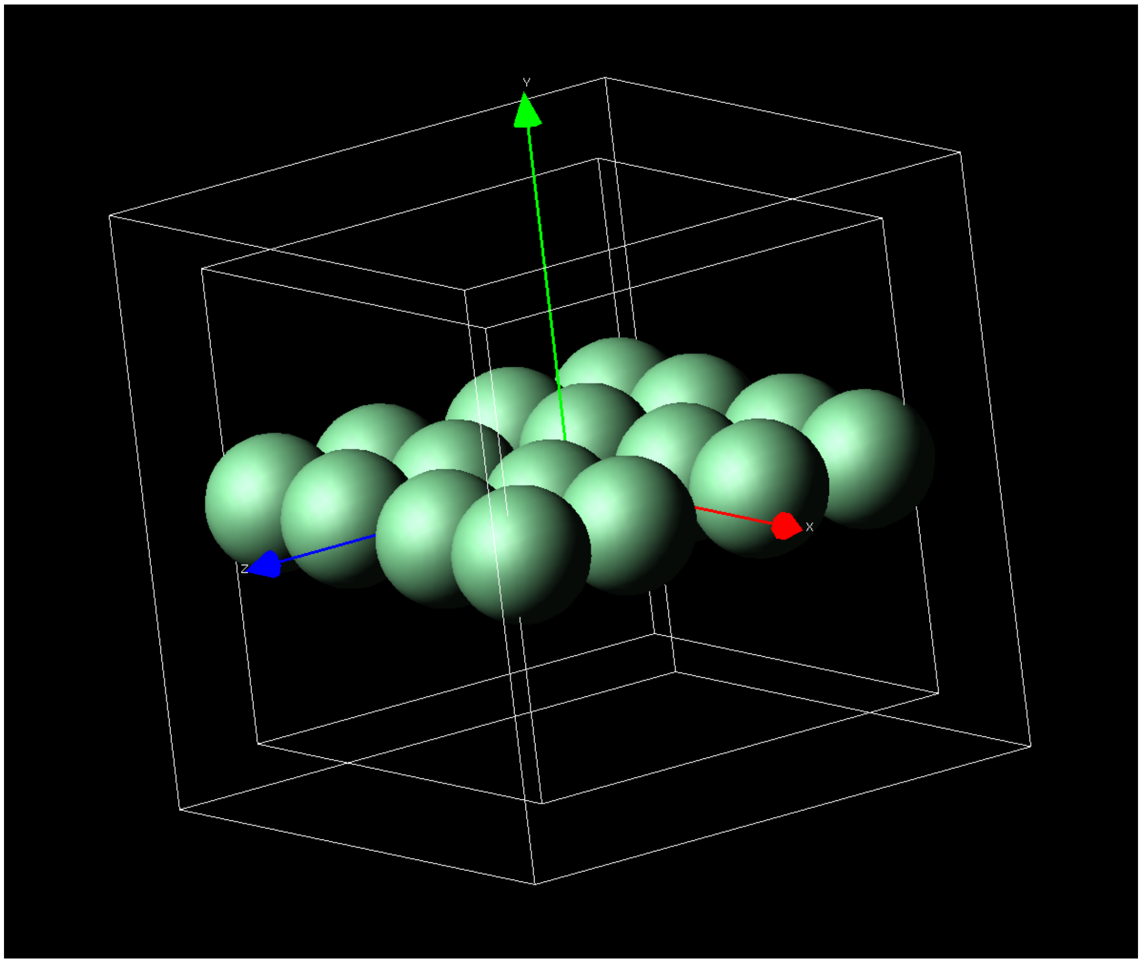


Figure 3.3: Particles Along Y-Axis (Y = 0)

Pythagorean theorem-type formula.

$$Distance_{r_1,r_2} = \sqrt{(x_1+x_2)*(y_1+y_2)*(z_1+z_2)} \tag{3.2}$$

If there's a collision between two particles, Equation 3.3 will be true.

$$r_1+r_2 < \sqrt{(x_1+x_2)*(y_1+y_2)*(z_1+z_2)} \tag{3.3}$$

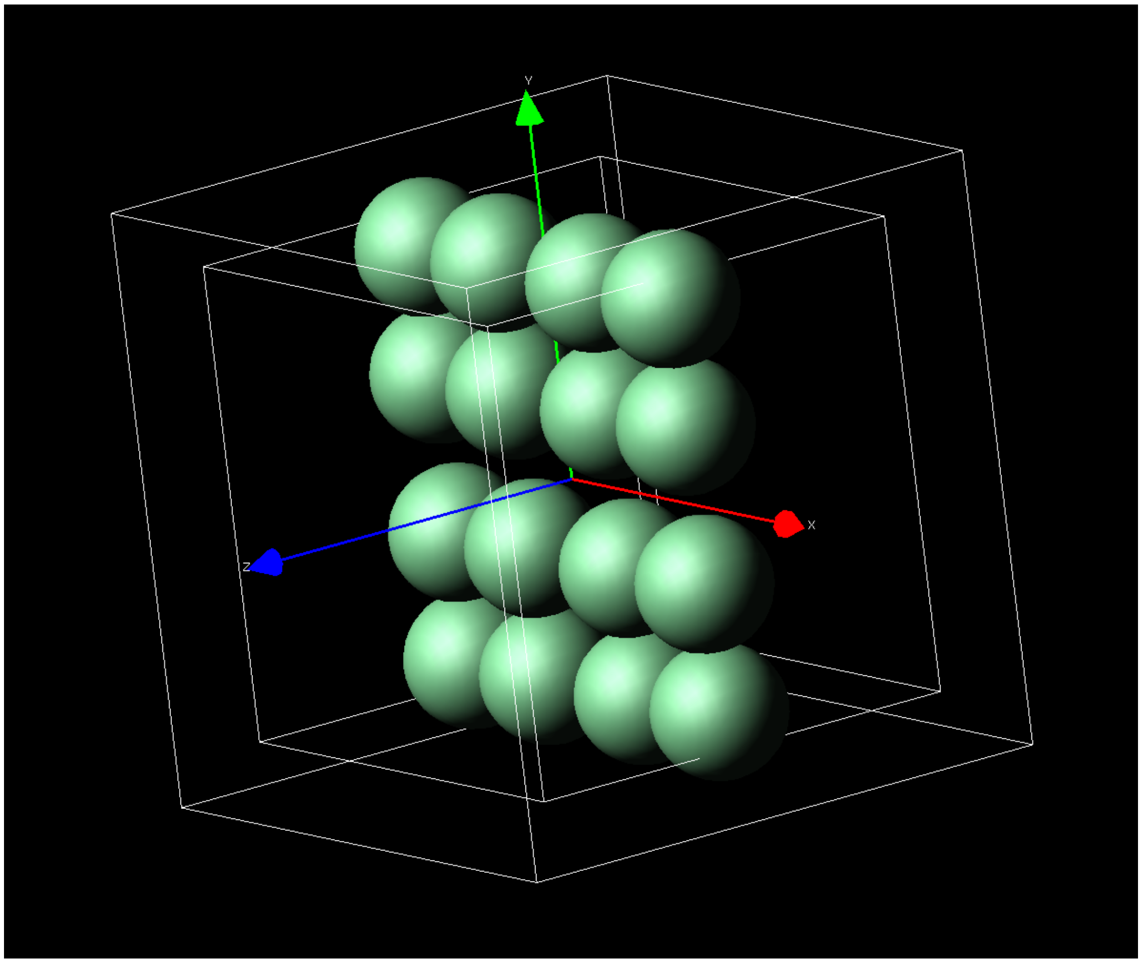


Figure 3.4: Particles Along Z-Axis (Z = 0)

Since the collision detection calculation will be done more than any other in the simulation, it should be as efficient as possible. The square root function, as described by Soderquist & Leeser (1996), is computationally expensive – much more so than a simple floating point multiply. Therefore, Equation 3.4 is calculated instead.

$$(r_1 + r_2) * (r_1 + r_2) < (x_1 + x_2) * (y_1 + y_2) * (z_1 + z_2) \tag{3.4}$$

The particle system is intended to contain a certain number of particles in the chamber at a given packing fraction. Specifically, based on a given packing fraction (.6300 in all cases), the particles are scaled in size to fit $1/3$ of them in the control volume. The other $2/3$ of the particles will fall outside of the control volume. The $1/3$, $2/3$ split is done to eliminate edge effects that can cause non-random patterns in the chamber.

All data from models run for this research was calculated on a MacBook Pro running Windows XP with a 2.0 GHz Intel T2500 (Core Duo) CPU and 2 GB of memory. While the processor is a dual-core model, every program implementation is single threaded, and each program run was restricted to one CPU core.

Four sets of runs using this algorithm are presented in this research. Each set of data differs only in the arrangement of the particles before the algorithm begins. The first will be a single particle-width column above the center of the control volume. The second and third will be densely packed rows of particles. The second will span only the control volume and the third will span the expanded volume. It's worth noting that the first three starting arrangements are not random in any way, other than the color of the particles.

The fourth starting arrangement will be a random pattern. Particles will be positioned in the control volume on the floor (the minimum Y-Axis value) with a random X- and Z-Coordinate. Particles will continue to be placed until there is a collision with a previously fixed particle. The Y-Axis value will be increased until the collision is resolved, so there is no more collision. Particles will be randomly initialized with a new Y-Axis value until there is another collision. This will be repeated until there are no particles left.

Ten colors are used in each particle pack and distributed as evenly as possible in each run. Particles start in virtual ’bins’ where each bin is assigned a color. For each starting arrangement, when particles are placed into the testing framework, they are selected one by one from random bins until all bins are empty. The differing colors are used only to aid in visualization and give a general view of randomness.

## 3.2 Evaluation Criteria

There will be four metrics used to evaluate the model with different starting positions: packing density, randomness, speed of algorithm, and scalability. Packing density and randomness involve the “correctness” of the algorithm. The latter two show the efficacy of use in combinatorial chemistry applications.

Typically, the density of a particle pack is determined via a simple volume comparison between the control volume and the total volume of the particles contained within the volume. This simple approach requires every particle to be entirely contained within the control volume. The problem with this approach is that to maintain the particles inside the boundaries of the control volume, edge effects are introduced. Edge effects are characterized by patterns that are formed in the particle pack that are caused by a pattern (flat walls, for example) in the edge of the container. Space that could be filled partially by a particle is left empty, thereby reducing the measurable packing density. Sometimes, a Monte Carlo method is used to find the density. Random points in the control volume are selected and tested to determine if they fall within a particle or in the empty space of the volume. This can be time consuming and have a low rate of precision.

The method used in this research is a hybrid between the simple and the Monte Carlo. The volume of the control volume is known; as a unit cube, its volume is one. The volume taken by the particles in the control volume will be calculated in two parts: the volume of the particles totally contained in the control volume and the volume of the particles that are only partially in the control volume. The volume of particles totally in the control volume will be calculated by Equation 3.5, a simple sphere volume formula.

$$v_{sphere} = \left(\frac{4}{3}\right)\pi R_3 \tag{3.5}$$

Particles that are only partially in the control volume will have their volume determined with the Monte Carlo method. The two particle volumes are then added together and the packing density $\rho$ is calculated with Equation 3.6.

$$\rho = \frac{v_{particles}}{v_{control}} \tag{3.6}$$

Randomness is a much more difficult value to measure. In one sense, randomness is the absence of a predictable pattern. For example, suppose you had a random number generator and it gave you the number eight. Is that number random? You can't determine randomness simply by looking at one number in a series. You would have to examine many numbers generated before and after the target number in order to find its randomness. Determining randomness in particle packs is more complex. The easiest way to measure randomness among groups of particles is by using a statistical pairwise correlation function called the Radial Distribution Function, or RDF. The RDF measures the correlation from one particle to all of the other particles. At distances close to the particle, the RDF will be relatively high. As the distance from the particle gets larger, the RDF will approach 1, indicating that there is no correlation (no predictable pattern) between particles.

The RDF, in general, is an indicator of **where** all of the other particles are in a pack in relation to one particle specifically. Starting at the center of the particle selected, the chamber is broken up into circular shells and a count is taken of how many particle centers there are in each shell.

Here's a simple example in two dimensions. Figure 3.5 shows a shell centered around .8 diameters from the center of the object particle. It is easy to see that it is physically impossible for there to be any particle centers in this shell. Figure 3.6 shows the shell centered around a particle diameter of 1. There are many particle centers in this shell. In a reasonably dense particle pack, there will be many particles in the shell around a diameter of 1.

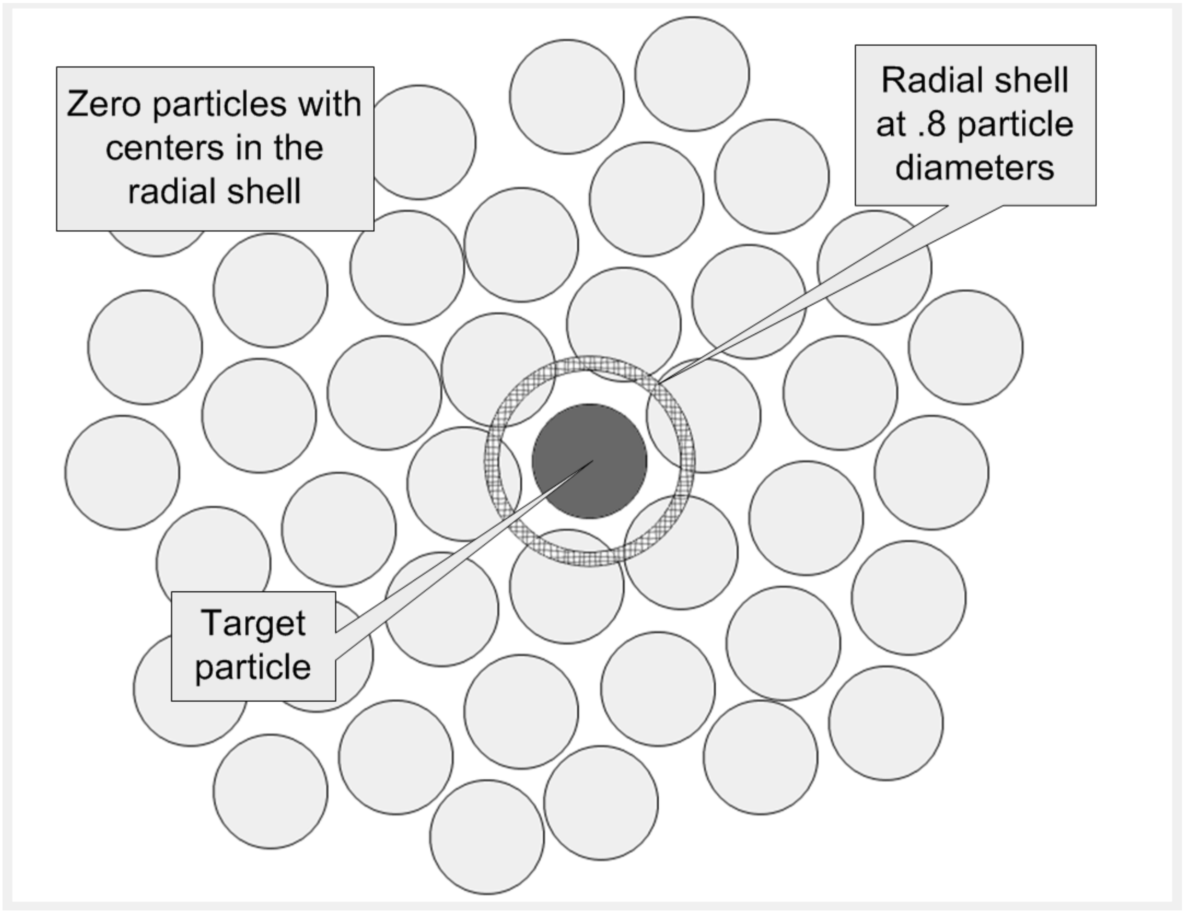

Zero particles with centers in the radial shell
Radial shell at .8 particle diameters
Target particle

Figure 3.5: Radial Shell at .80 Diameters

Figure 3.7 shows the shell centered at a particle diameter of 1.5. Again, as

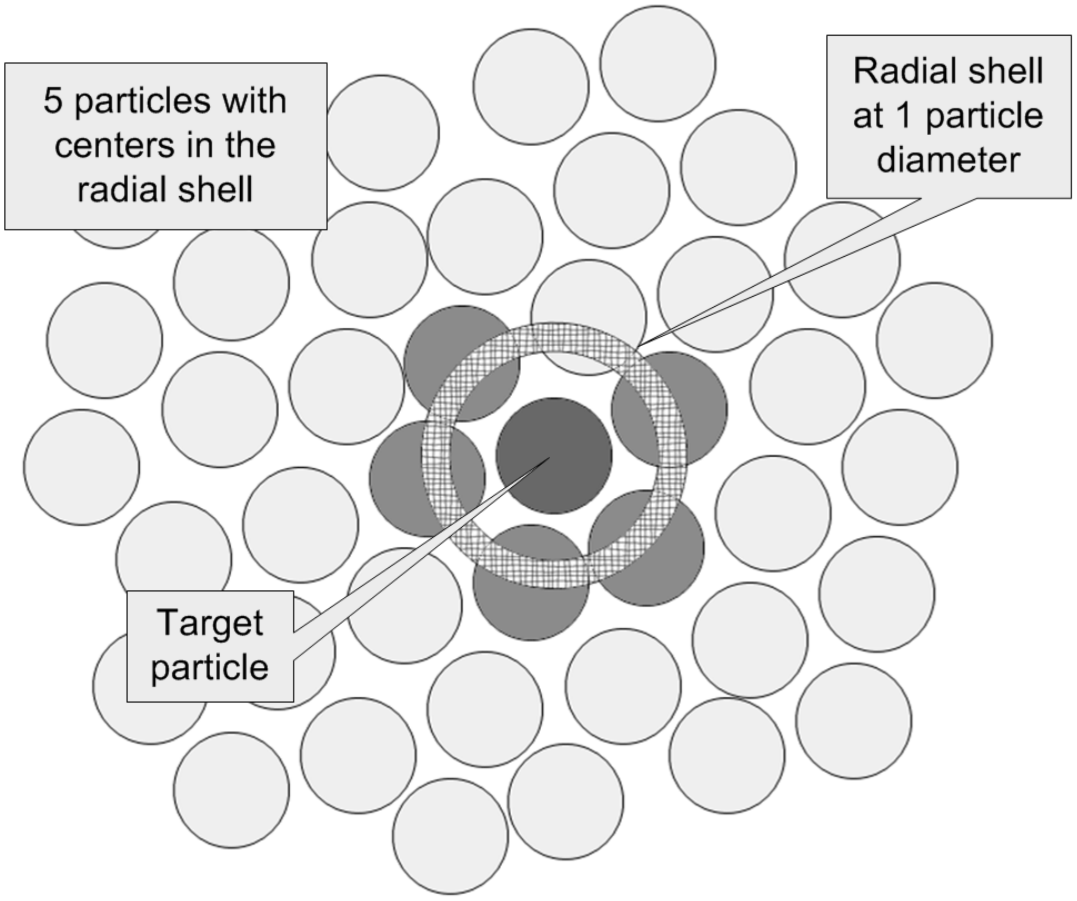

5 particles with centers in the radial shell
Radial shell at 1 particle diameter
Target particle

Figure 3.6: Radial Shell at 1 Diameter

seen in the .8 diameter shell, there aren't any particles.

Figure 3.8 shows the shell centered at a particle diameter of 2.0. There are particle centers found in this shell, but not at the same percentage possible if the particles were perfectly (tightly) packed. It can be seen that because of the looseness of this pack, there are particles that fall just inside and just outside of the shell.

Using the counts from these shells, an RDF graph of particle arrangement

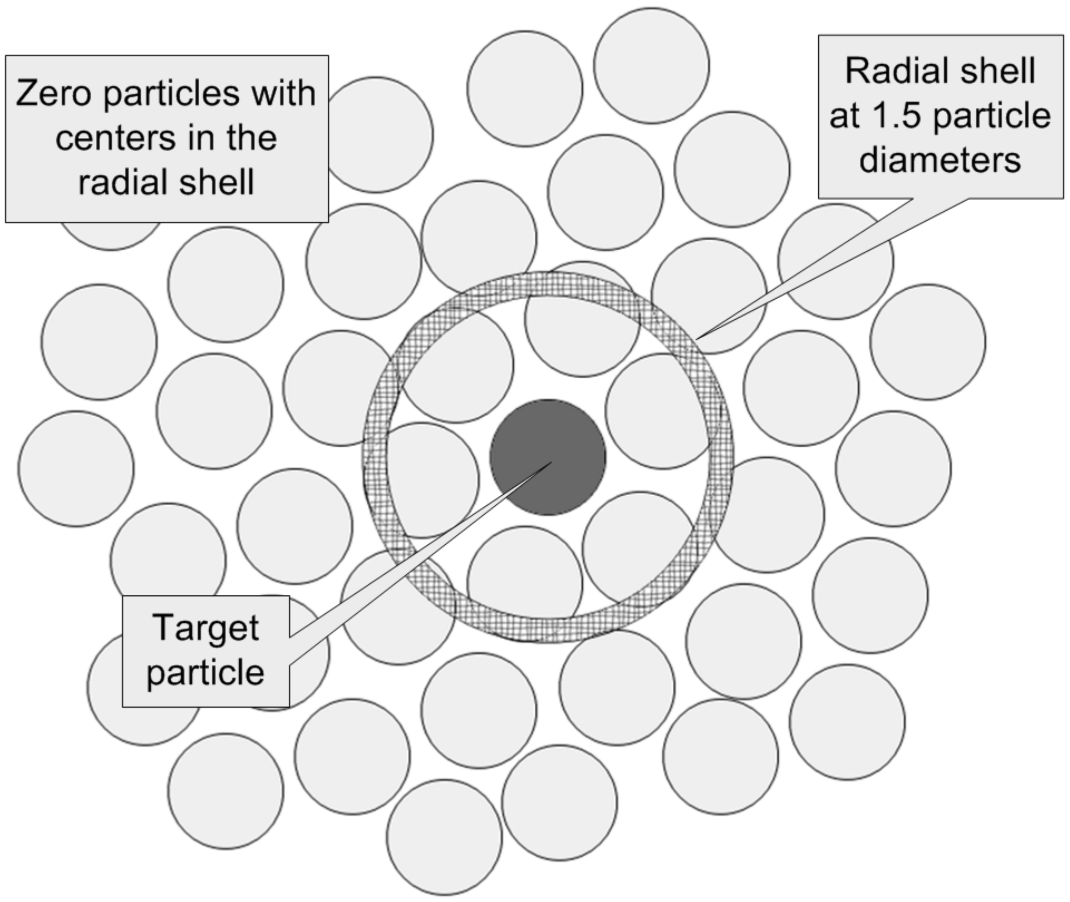

Zero particles with centers in the radial shell
Radial shell at 1.5 particle diameters
Target particle

Figure 3.7: Radial Shell at 1.5 Diameters

can be made. According to Torquato (2002), the RDF is defined by Equation 3.7:

$$g_2(R) = \frac{n(R)}{\rho v_{shell}(R)} \tag{3.7}$$

Here, $g_2(R)$ represents the RDF as a function of radial distance. $n(R)$ is an estimate of the number of particles at a given radial location. $\rho$ is an estimate of the number density of the system. The most simplistic estimate of this pa-

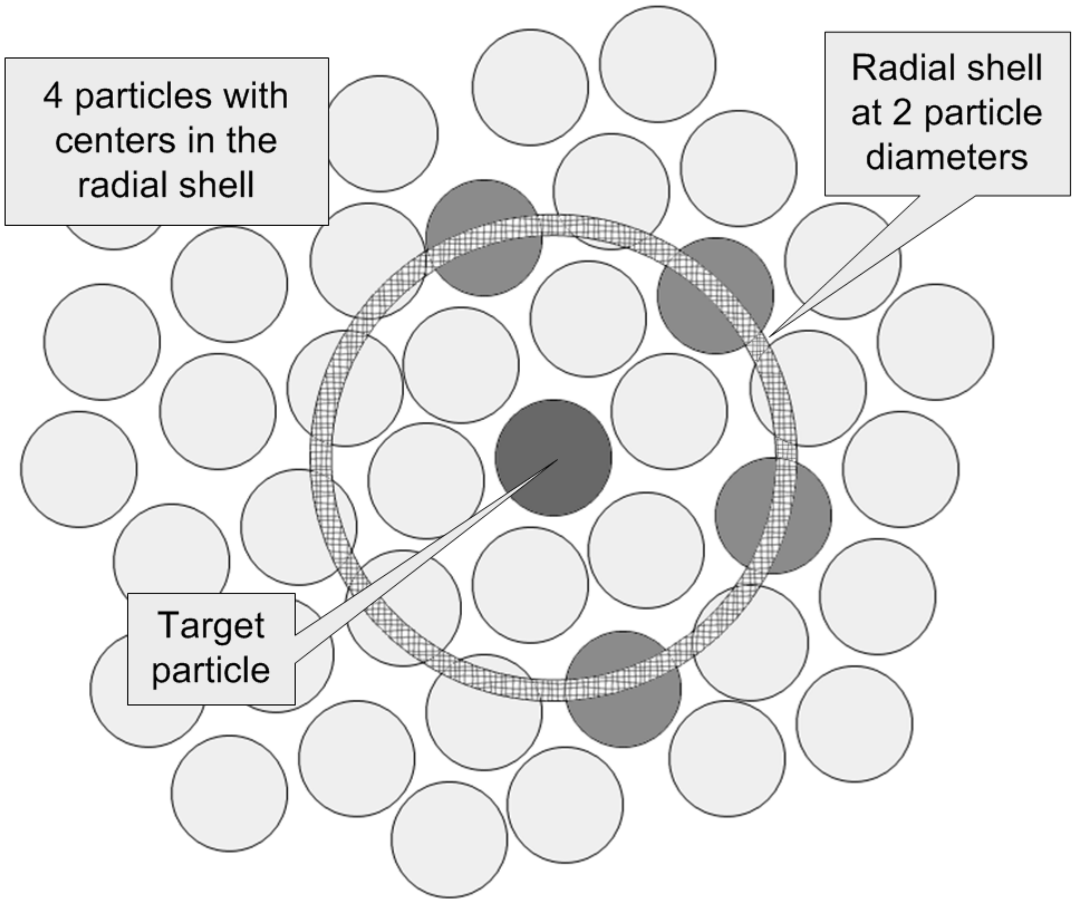

4 particles with centers in the radial shell
Radial shell at 2 particle diameters
Target particle

Figure 3.8: Radial Shell at 2 Diameters

rameter is the packing fraction, or ratio of sphere volume to control volume as in Equation 3.8:

$$\rho = \frac{\sum_{i=1}^{N} v_{Sphere,i}}{v_{Control}} \tag{3.8}$$

$v_{Shell}(R)$ is the volume of the radial shell being considered. In this analysis, a control particle near the center of the control volume is selected at random, and the RDF is determined using the remainder of the particles within the control volume of the simulation.

To compute $n(R)$, particles at various radial locations from the control particles are placed in radial bins and summed, effectively creating a histogram of distance from the control particle. The number in each bin, $n_k(R)$, is then divided by the total number of particles to create a ratio for a particular bin, $k$, according to Equation 3.9:

$$n(R) = \frac{n_k(R)}{N} \tag{3.9}$$

where k is the bin number and $N$ is the total number of particles. The radial shell volume is determined by a simple geometrical relationship in Equation 3.10:

$$v_{Shell}(R) = \frac{4\pi R^3}{3} \left[ \frac{\left(R + \frac{\Delta R}{2}\right)^3 - \left(R - \frac{\Delta R}{2}\right)^3}{R^3} \right] \tag{3.10}$$

where $\Delta R$ is the radial increment used for the bin, $k$. There is some dependence on the radial increment chosen in this method, so the radial increment is typically reduced until convergence in the numerical values is observed. Since the RDF is defined only for homogeneous distributions of spheres, all comparisons using the RDF were performed on mono-modal particle distributions.

To be useful in combinatorial chemistry applications, any modeling algorithm must be fast and scalable. Speed is simply a measure of how fast a particular model runs, while scalability relates to how the packing fraction and RDF change as the number of particles in the model increase.

# Chapter 4

# SGMP: The Spin Gap Move Protocol

## 4.1 Model Algorithm

The Spin Gap Move Protocol (SGMP), which this research will be using, is defined by four repeating phases: downward movement, circular sweep, gap selection, and the movement of the particle into a gap.

Before phase one begins, spin offset tables are calculated that will be used in the circular sweep phase. These are done only once, because sine and cosine functions are used, which are computationally expensive. The X- and Z-Axis values for a small circle are calculated with Equations 4.1 and 4.2.

$$x_{offset,\theta} = \sin\theta * (d_{particle}/Factor_{circle}) \tag{4.1}$$

$$z_{offset,\theta} = \cos\theta * (d_{particle}/Factor_{circle}) \tag{4.2}$$

The circle factor is a value that is used to decrease the size of the circular sweep and is set to 50 in all of these experiments.

The first phase is downward movement. A particle not already in the completed pack is selected, which will be called Particle A. Then, a lower columnar neighbor list is created, containing the closest 10 particles that are directly below the selected particle. Particle A is then moved downward (in the negative Y-Axis direction) in a very small increment. After it is moved downward, a check for collisions with its lower columnar neighbors is done. If there is no collision, the particle is moved downward again. If there is a collision, shown in figure 4.1, phase two starts.

At the beginning of phase two, the circular sweep, the neighbor list is recalculated with particles that are near Particle A, not only the particles that are below it to reduce the number of collision calculations that need to be made. Particle A is moved in a circle around the Y-Axis using the spin offset tables calculated before phase one. At each position in the circle, a check for collisions with the neighbor list is made. Each non-collision position in the sweep is stored. Figures 4.2 through 4.13 show several positions in the sweep. When the circular sweep is complete, phase three begins.

In phase three, gap selection, the points where Particle A didn't collide with any other particles are merged to form non-collision arcs. In the example shown in Figure 4.14, there is only one non-collision arc and it is $38^\circ$. If there are mul-

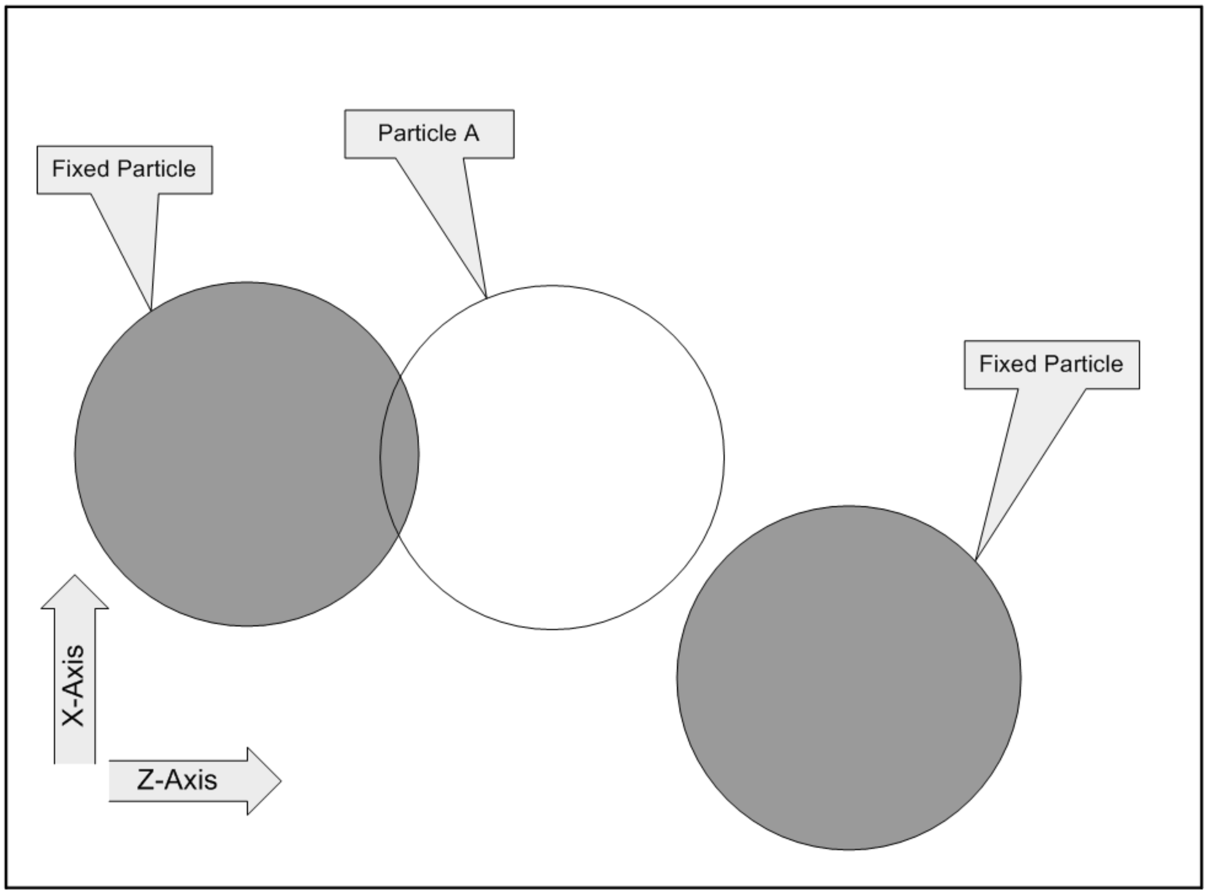


Figure 4.1: Y-Axis Cross Section, Falling Particle Initial Collision

tiple non-collision arcs, the largest non-collision arc is selected and bisected.

In phase four, movement, shown in Figure 4.15, the particle is moved in the direction of the point of bisection.

## 4.2 Improvements on Previous Work

Two general approaches are taken to solve packing problems, the assembly approach and the kinematic approach. The SGMP invented by this author is an

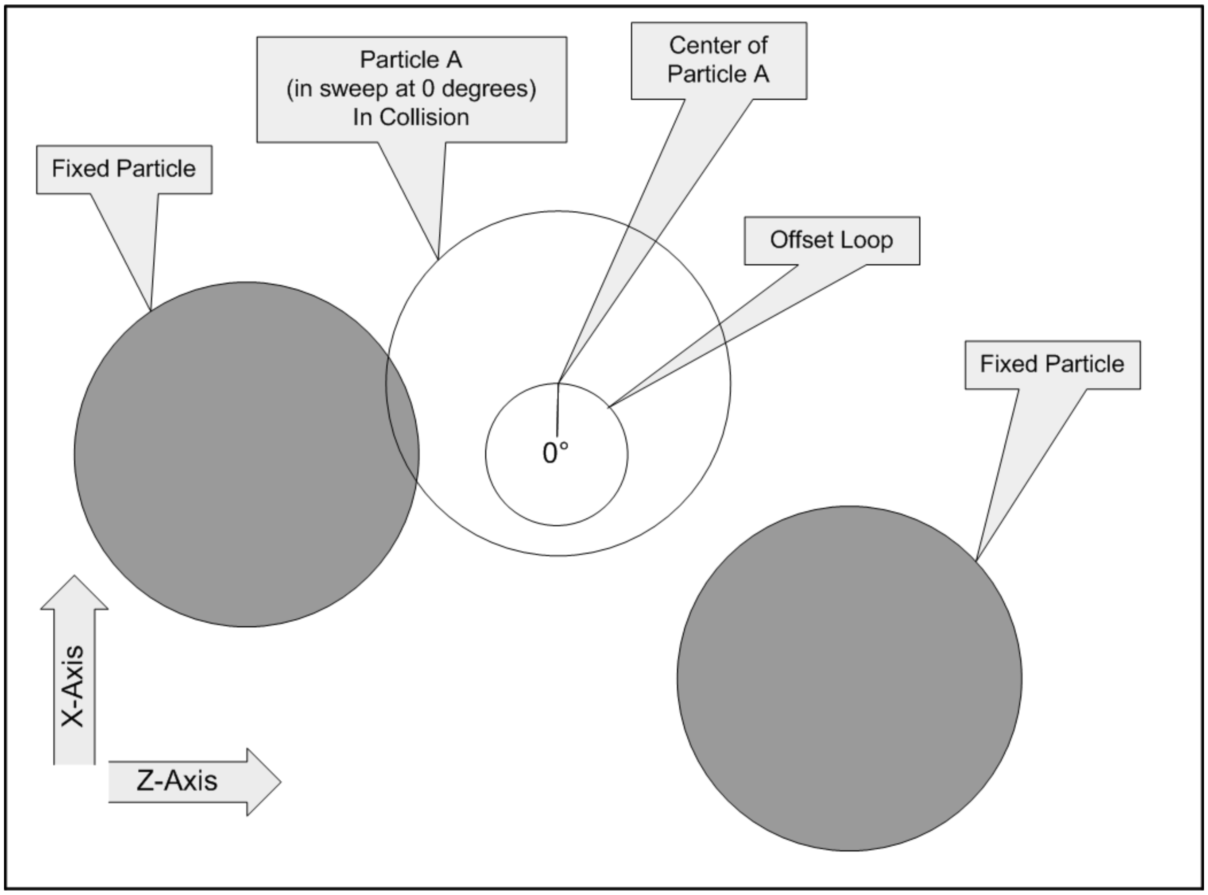


Figure 4.2: Y-Axis Cross Section, Circular Sweep: Zero degrees

assembly approach. Assembly models were largely abandoned by researchers after the 1970's but some new work has recently begun.

There are several techniques that are new inventions in this thesis, either in the algorithm or in its analysis. The first technique is to use very few computationally expensive sine and cosine operations. There is a sine/cosine pair for each degree in a 360 degree circle. This is a constant factor no matter how many particles are involved.

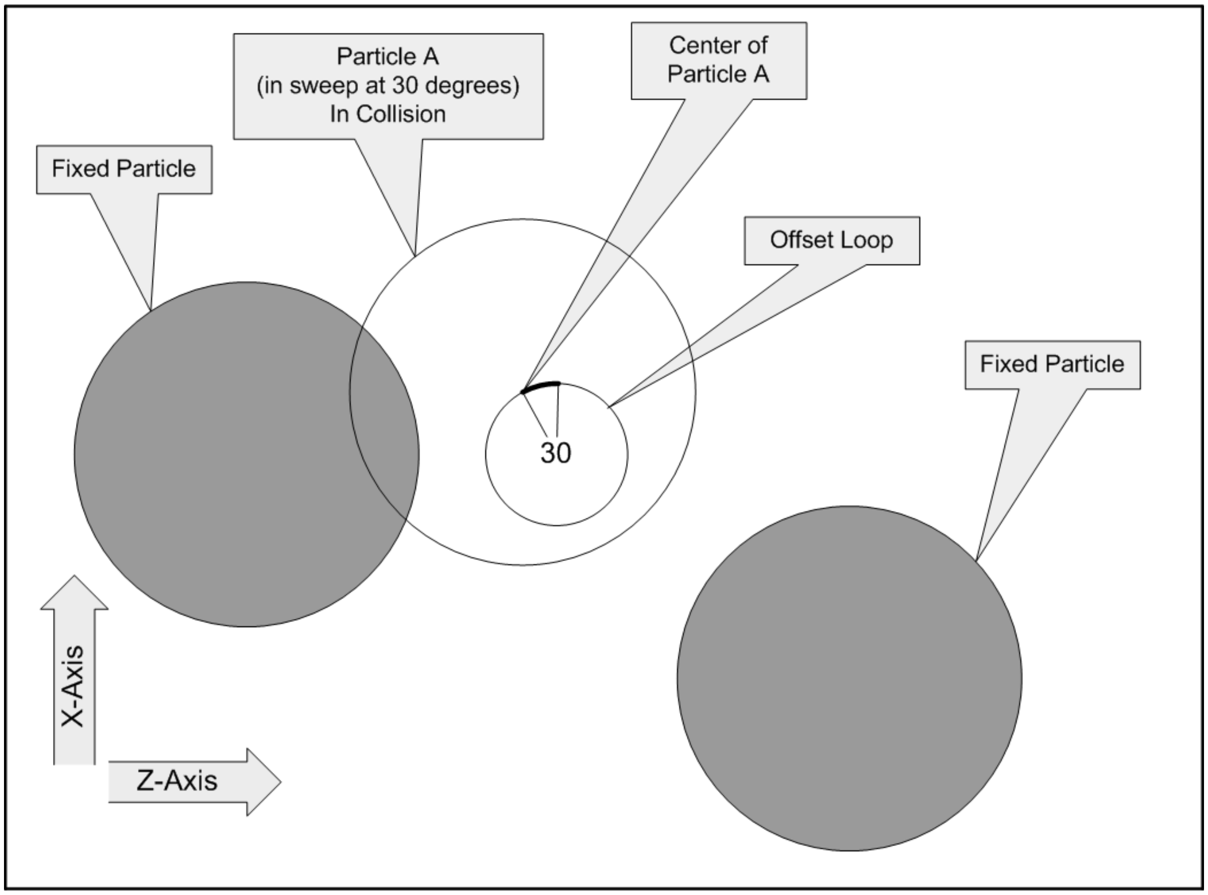


Figure 4.3: Y-Axis Cross Section, Circular Sweep: Thirty degrees

Also in this newly created algorithm is the use of expanded boundaries. This reduces the overhead of having periodic boundaries like most modern algorithms. However, there is extra overhead in having extra particles not taking part in the measurable pack, but this author believes that it pays for itself.

Where Shi & Zhang (2006) used a model in which a particle would ‘hinge’ downward after coming in contact with two other particles, the SGMP uses the circular sweep technique. The Shi & Zhang (2006) method stops when a

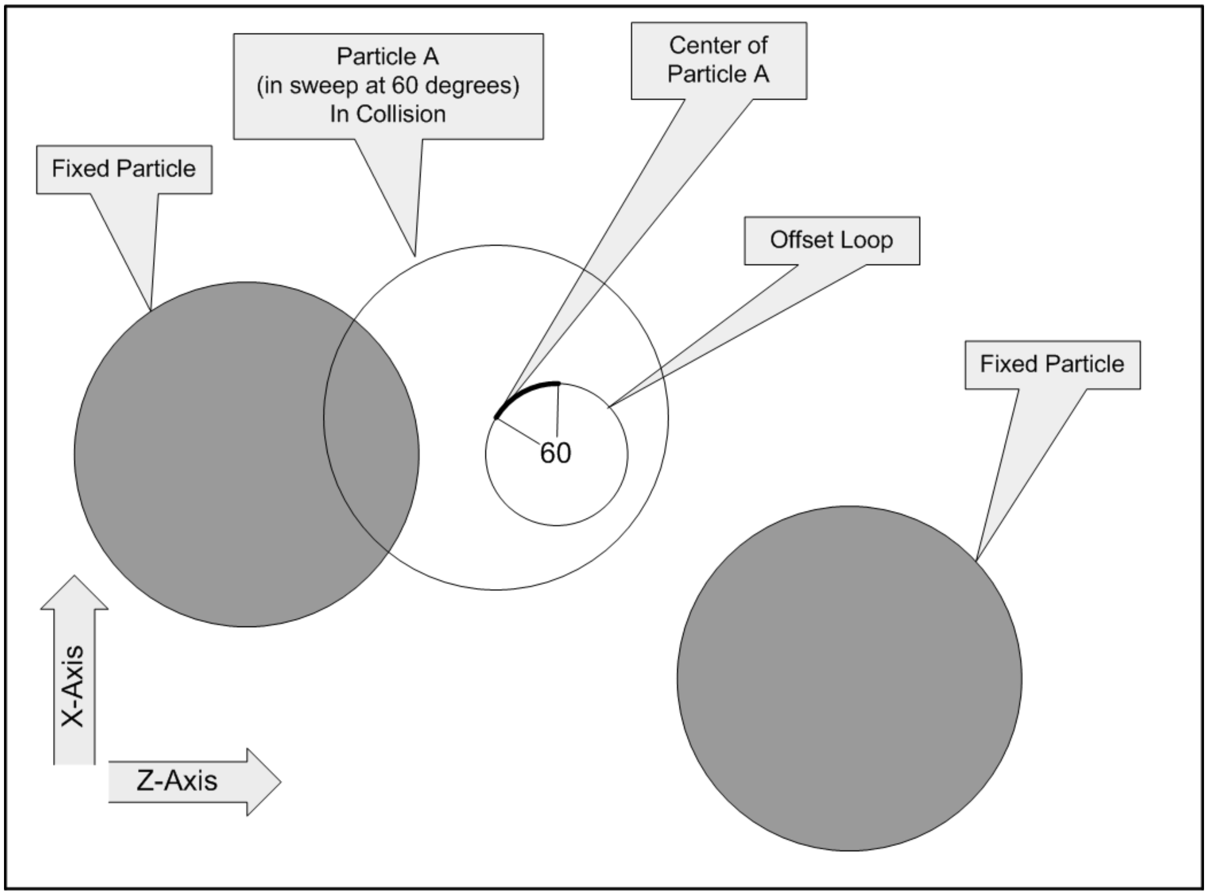


Figure 4.4: Y-Axis Cross Section, Circular Sweep: Sixty degrees

particle comes into contact with three others. The SGMP will allow further compaction if three-particle contact occurs, allowing for tighter packs.

Neighbor lists are summarized by Torquato (2002) as a way of reducing the computational overhead of collision checking; therefore, the focus is on the probability of the existence of particles close to a specific particle in all dimensions. To take advantage of that, during the downward movement phase of the algorithm, only particles that are directly below the moving particle are consid-

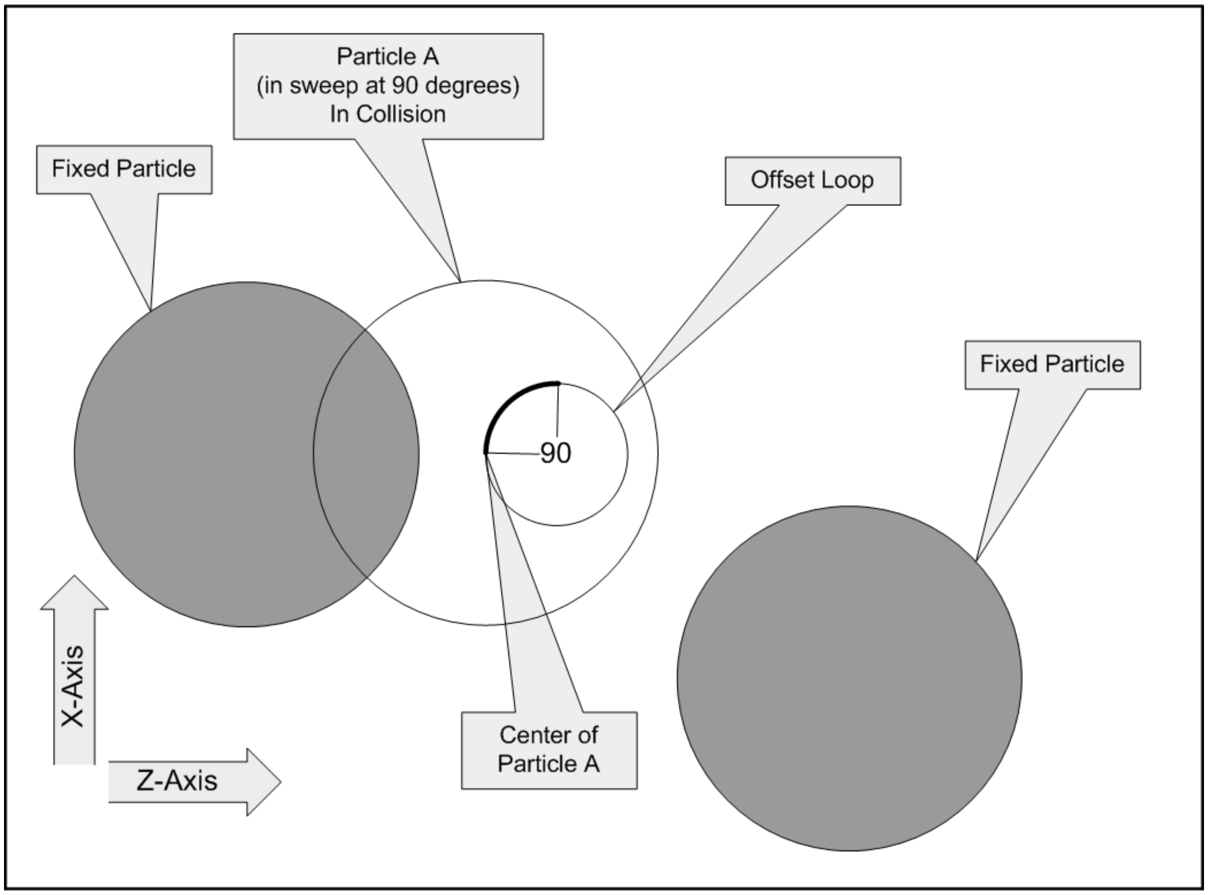


Figure 4.5: Y-Axis Cross Section, Circular Sweep: Ninety degrees

ered in the neighbor list. This reduces the number of collision calculations by half.

To avoid the complication and complexity of a collision with a wall boundary, when a particle touches one of the boundaries, it simply stops there. This can also be considered a drawback, as pack 'looseness' might be higher at the edges of the pack. This should be overcome by the fact that the extended boundaries are not considered in the packing fraction calculation or the RDF

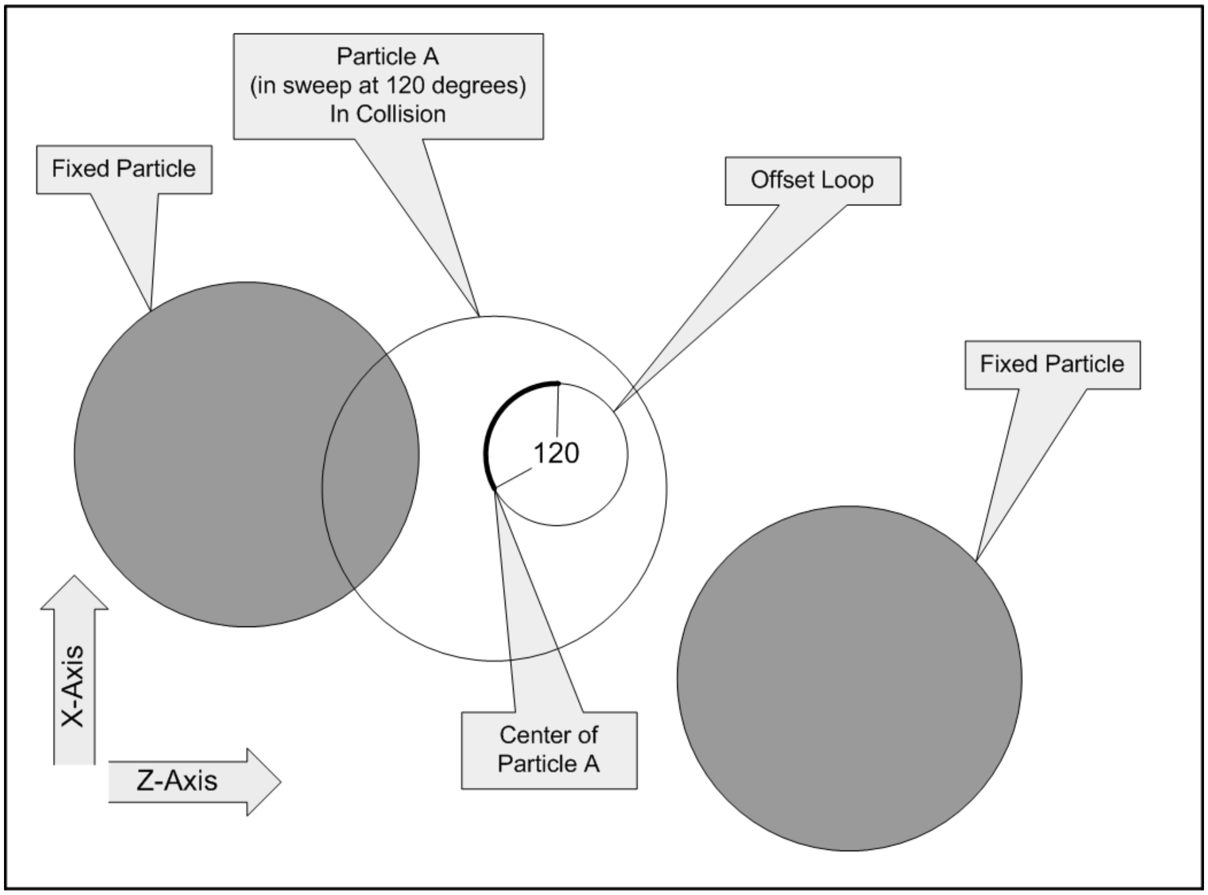


Figure 4.6: Y-Axis Cross Section, Circular Sweep: One Hundred and Twenty degrees

calculation.

One area that improves that packing fraction but may be too unrealistic is that the largest gap, after a circular sweep, is simply bisected and the particle moved in the direction of the bisection point. If the SGMP is intended to be a gravity simulation, this step disregards any momentum that might be built up as a result of previous movements. However, its intent is to form a tighter particle pack by selecting the **best** part of a gap. Further research should include

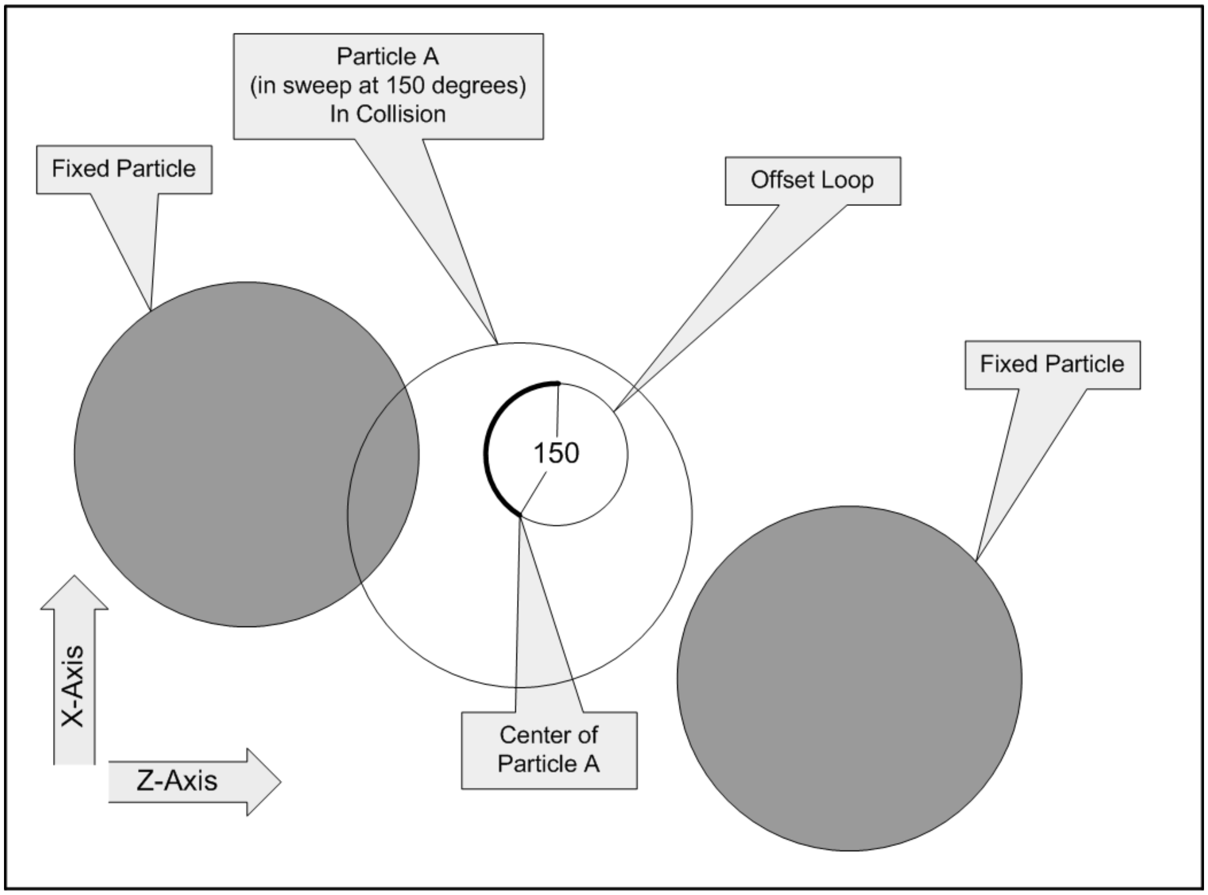


Figure 4.7: Y-Axis Cross Section, Circular Sweep: One Hundred and Fifty degrees

random gap selection and particle momentum.

Because of the extended boundary condition to remove edge effects, a new method of calculating the packing fraction was needed. Monte Carlo techniques in the entire pack would work, but to get both accurate and precise results, very high numbers of test points would be required. This was the original approach selected by this researcher; however, the large number of test points took almost as long as the packing model itself. A hybrid concept was

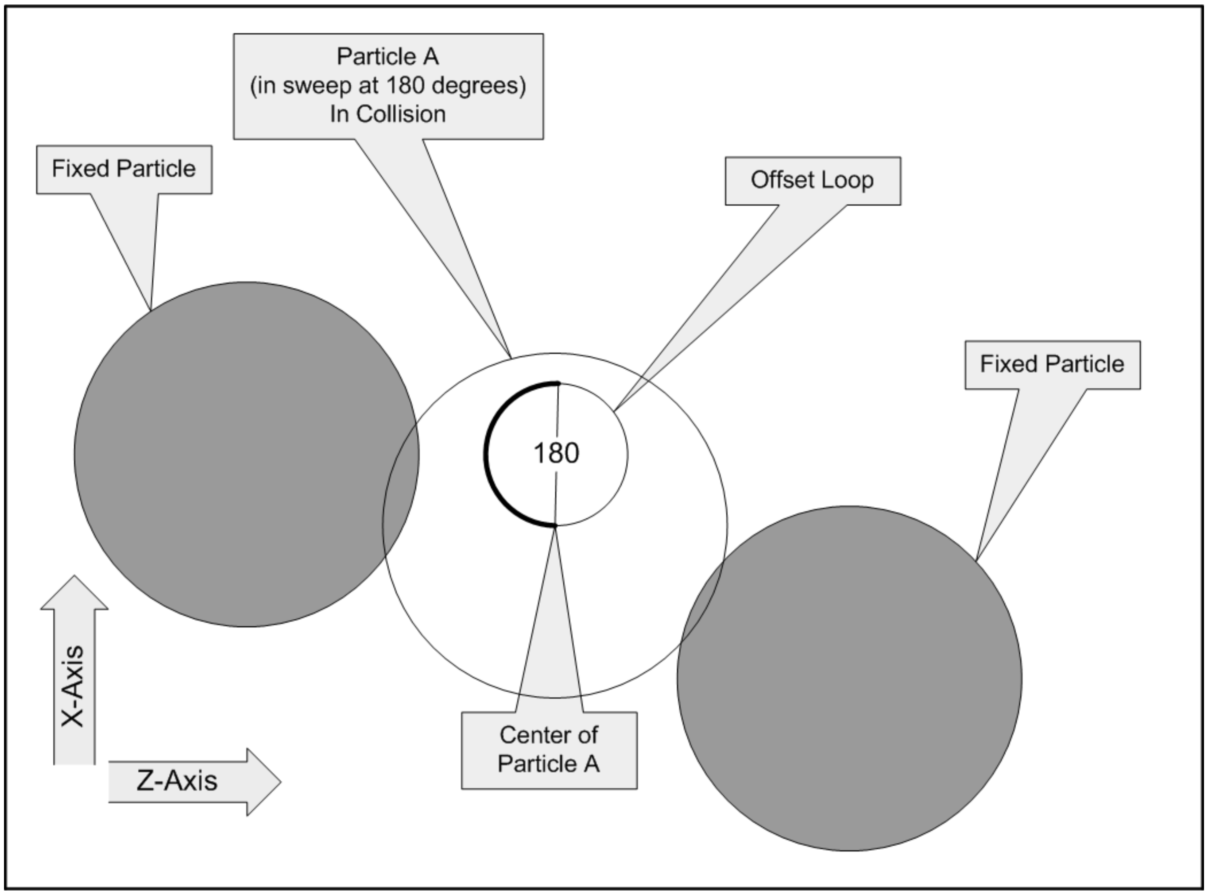


Figure 4.8: Y-Axis Cross Section, Circular Sweep: One Hundred and Eighty degrees

invented by this author in which only particles on a boundary used a Monte Carlo method.

## 4.3 Order of Operations

A complexity analysis of the SGMP is simpler than those of the kinematic variety. Due to its assembly nature, a complex analysis like that done by Krantz (1996) is unnecessary as the steps of the SGMP are well bounded in terms of the

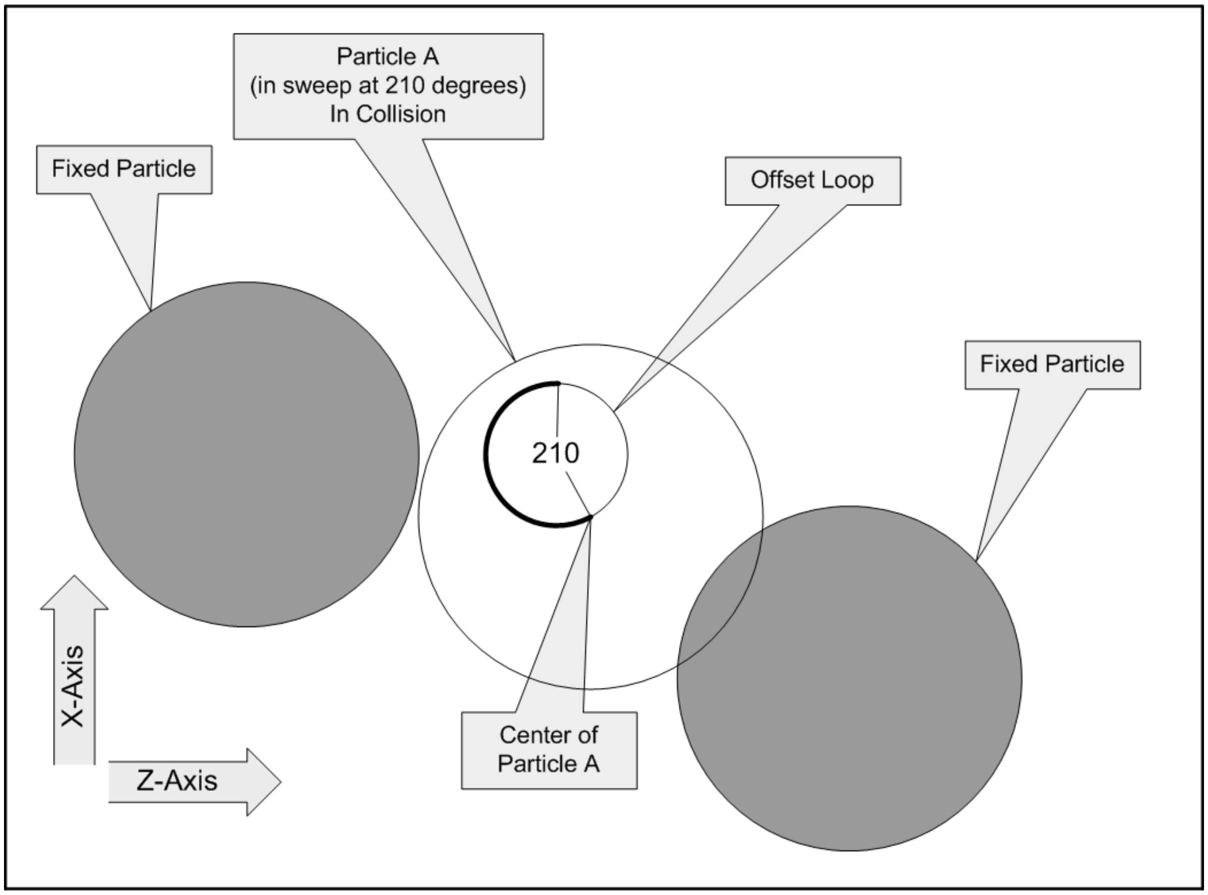


Figure 4.9: Y-Axis Cross Section, Circular Sweep: Two Hundred and Ten degrees

number operations required to generate a finished pack. Most of the operations in the SGMP are, in terms of operational complexity, linear. Only two steps, individually, have an operational complexity of $O(n)$. These are both of the nearest neighbor calculations. Each of the operations, except for initial spin offset calculation, is executed more than once. Table 4.1 summarizes the steps and their relative complexity.

The last four steps have an added complexity of $O(n\log n)$ when taken

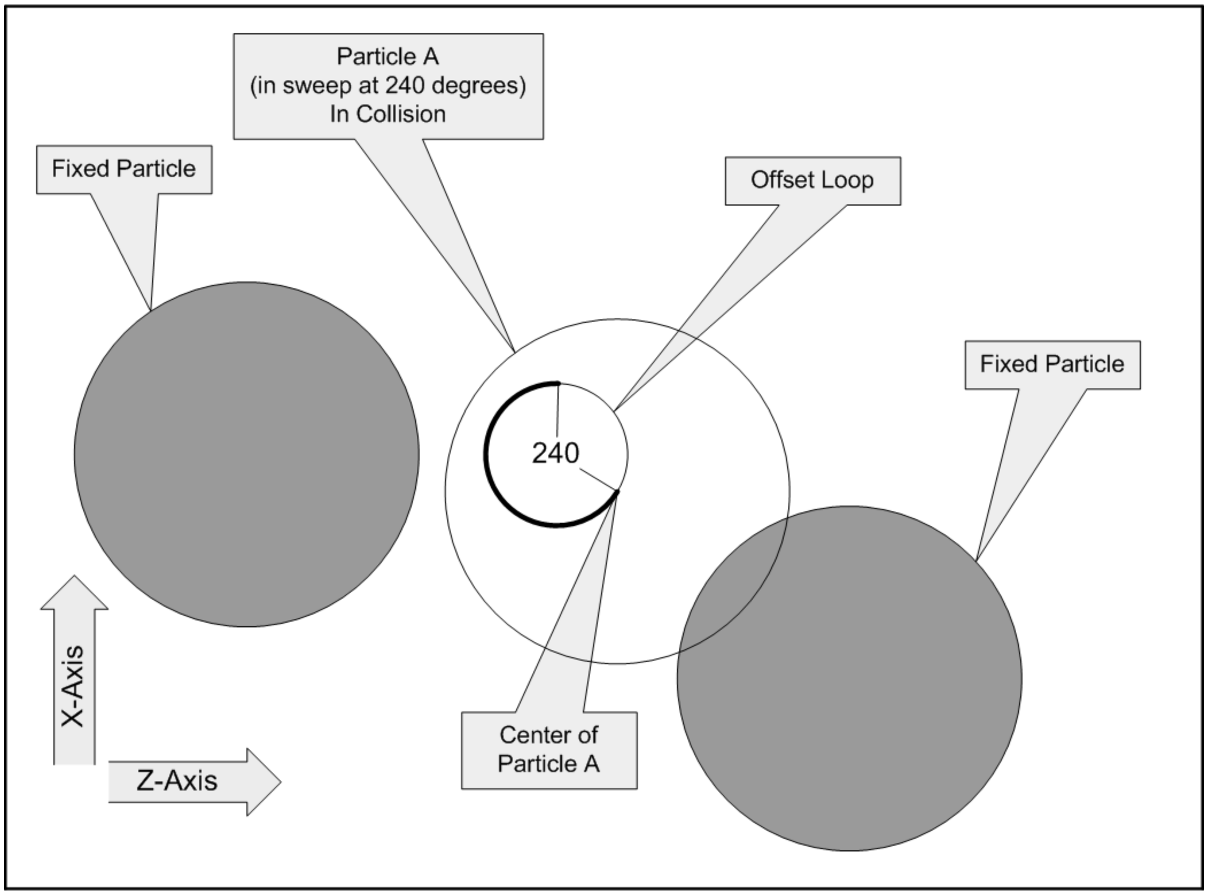


Figure 4.10: Y-Axis Cross Section, Circular Sweep: Two Hundred and Forty degrees

across the entire pack. The $O(n)$ portion comes from the operation being done to each particle in the pack. The $O(logn)$ portion requires more explanation. Each step in three through seven is processed when a falling particle first comes into contact with the previously processed fixed bed of particles. There is a reduced portion of the entire pack that the particle in question could possibly contact to find its final position. For example, in the 1002 particle run seen in Figure 4.16, only approximately eight particles can fit from one edge of the

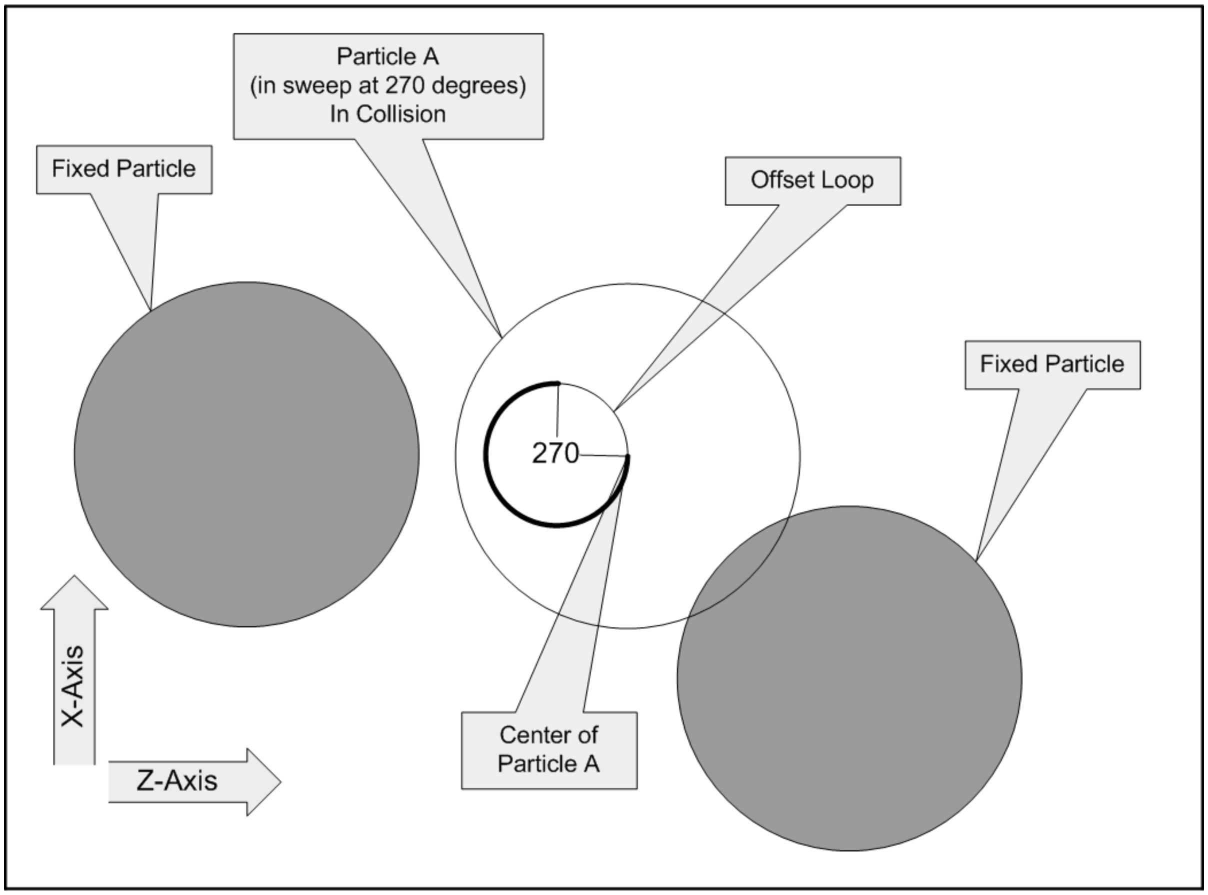


Figure 4.11: Y-Axis Cross Section, Circular Sweep: Two Hundred and Seventy degrees

packing space to the other. Taken across two axes, there are only around 60-100 particles that the falling particle could potentially contact.

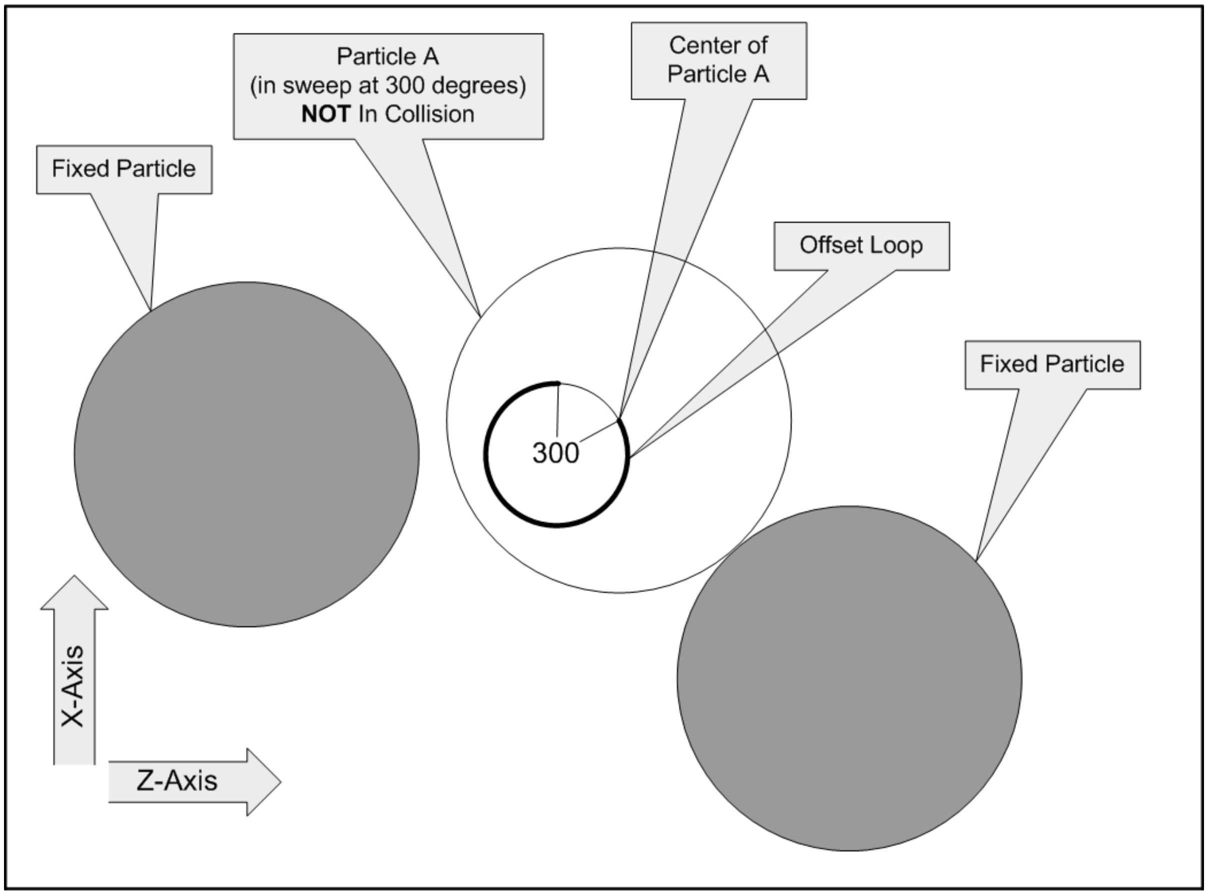


Figure 4.12: Y-Axis Cross Section, Circular Sweep: Three Hundred degrees

| Step Number | Step Description | One Time Complexity | Total Complexity |
|---|---|---|---|
| 1 | Calculate Spin Offsets | $O(1)$ | $O(1)$ |
| 2 | Generate Lower Neighbor List | $O(n)$ | $O\left(n^2\right)$ |
| 3 | Downward Drop | $O(1)$ | $O(n)$ |
| 4 | Circular Sweep | $O(1)$ | $O(n\log n)$ |
| 5 | Find the Largest Gaps | $O(1)$ | $O(n\log n)$ |
| 6 | Move Into Gap | $O(1)$ | $O(n\log n)$ |
| 7 | Generate Normal Neighbor List | $O(n)$ | $O\left(n^2\log n\right)$ |

Table 4.1: Operational Complexity for Steps of the SGMP

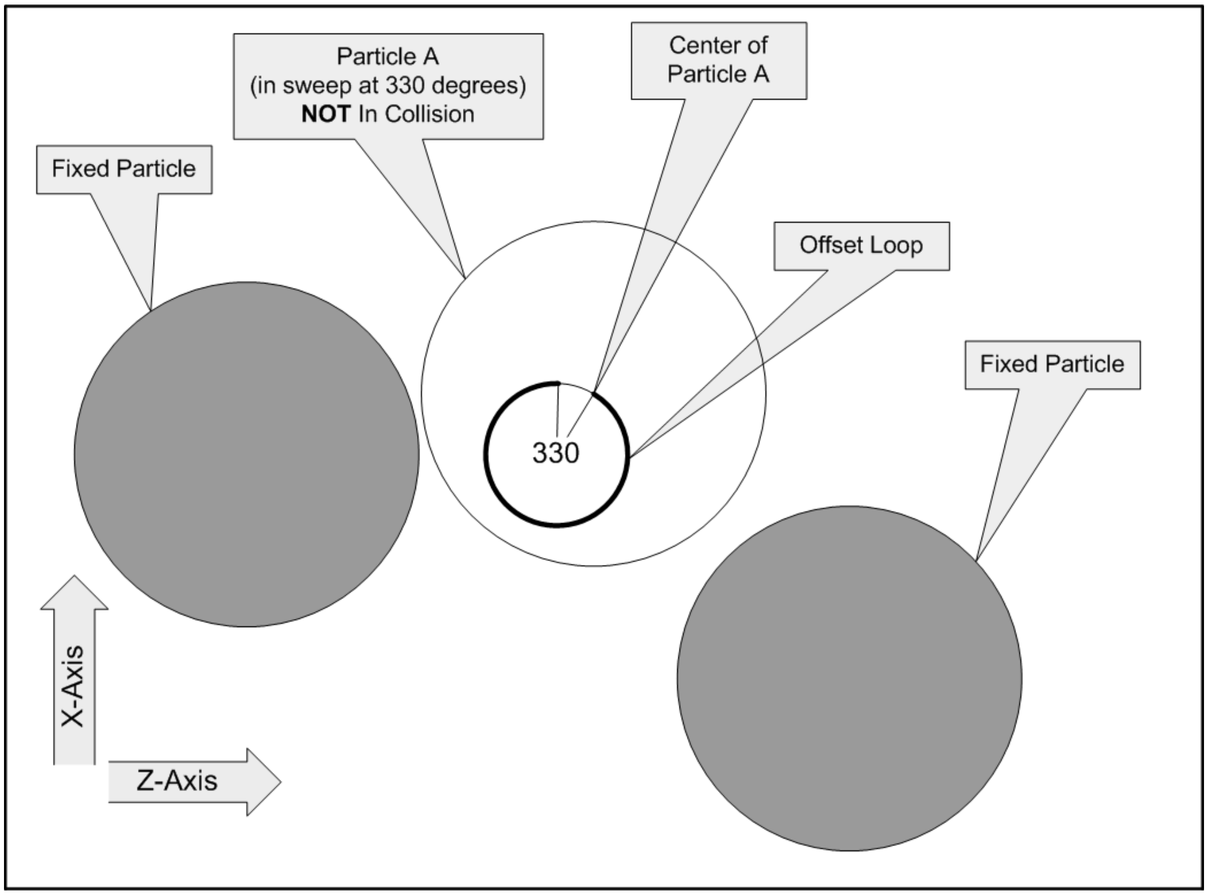


Figure 4.13: Y-Axis Cross Section, Circular Sweep: Three Hundred and Thirty degrees

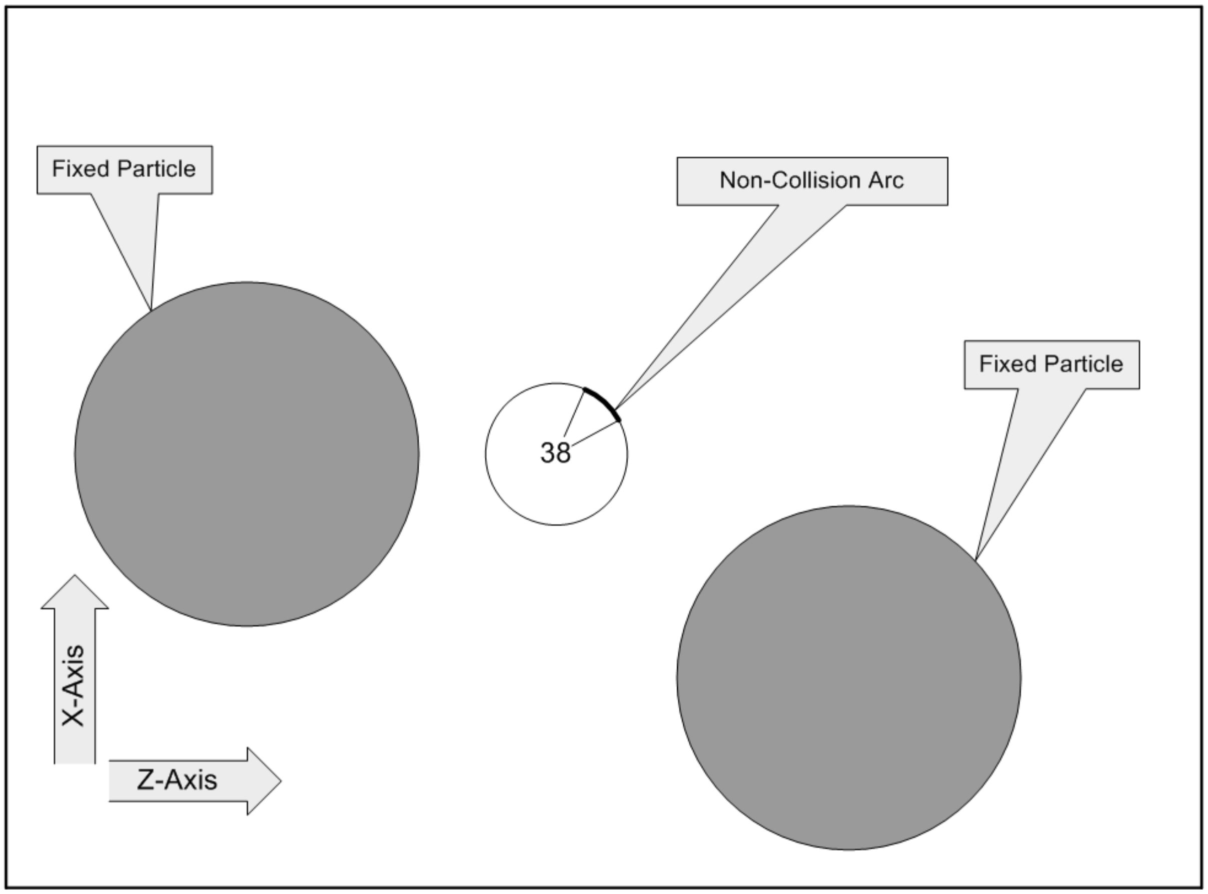


Figure 4.14: Y-Axis Cross Section, Gap: 38 degree Non-Collision Arc

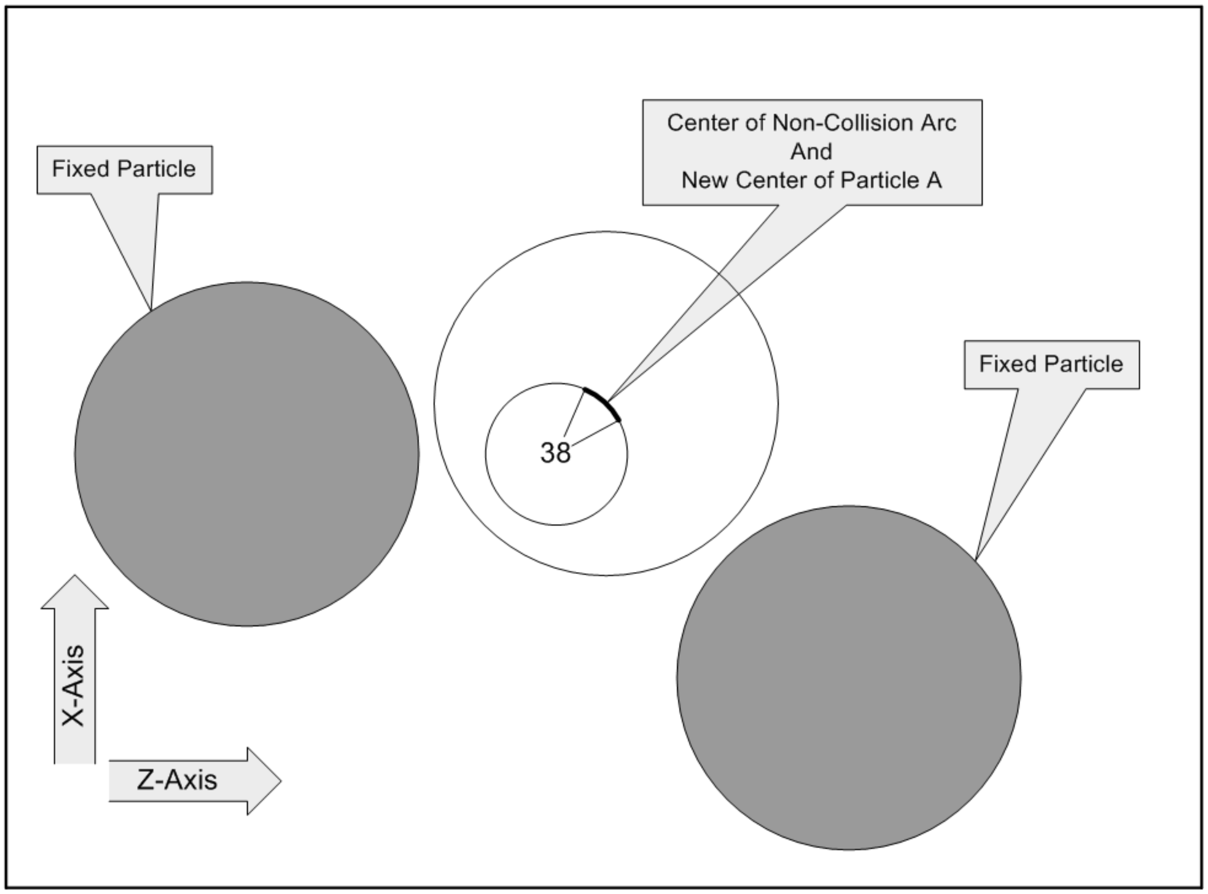


Figure 4.15: Y-Axis Cross Section, Move: New Position, Bisected Non-Collision Arc

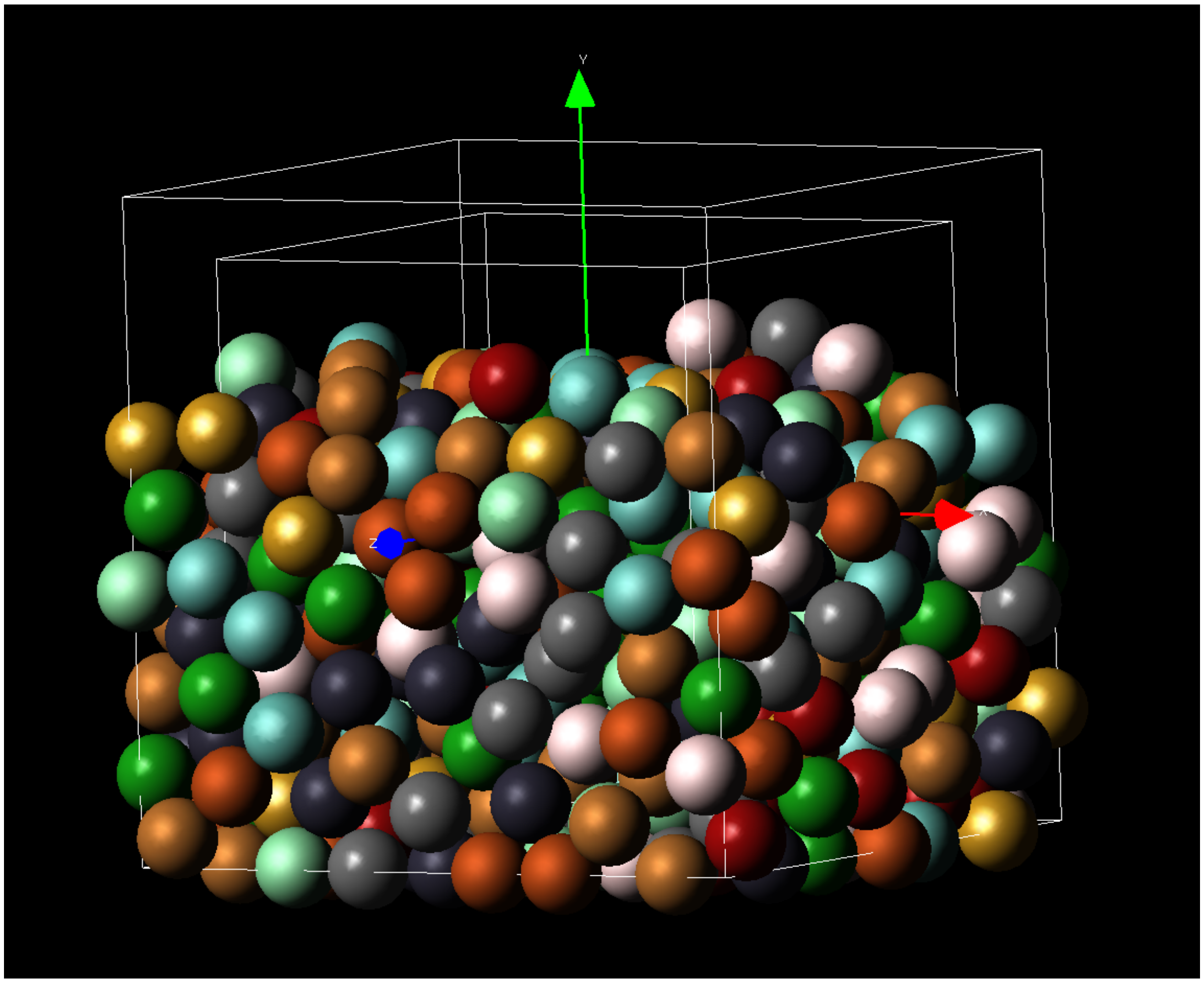


Figure 4.16: 1002 Particle Run, 500 Particles Fallen

# Chapter 5

# SGMP – Single Column Starting Pack

## 5.1 Model Setup

The initial starting pattern is the single column. Particles are randomly selected from the ten color bins and placed in the control volume at the center of the X and Z axes, with X and Z both being equal to zero. The Y-Axis value starts at -.625, the floor of the expanded volume. Each subsequent particle is placed above it (keeping X- and Z-Axis values the same) by adding 1.2 times the diameter of particle to the Y value of the previous particle. The model was run eight times with different numbers of particles. For illustration, Figures 5.1 through 5.3 show the model at three different points. All three images are taken at the same position and zoom level. Table 5.1 shows the summary of the runtime and final packing fraction with various numbers of particles.

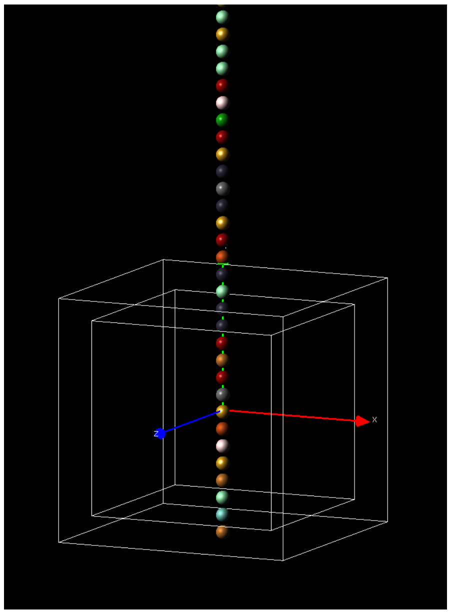


Figure 5.1: 9000 Particle Pack, 0 Particles Fallen

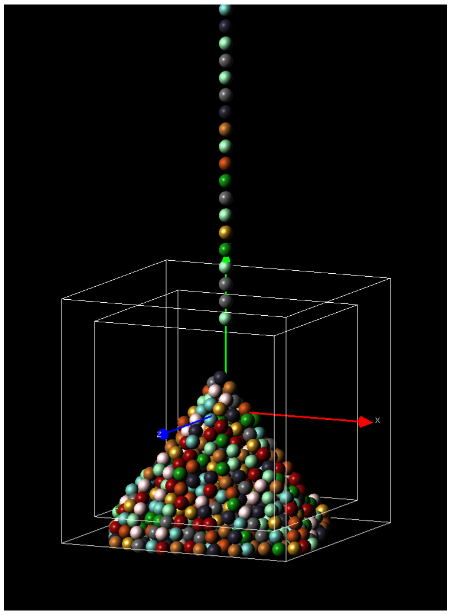


Figure 5.2: 9000 Particle Pack, 1000 Particles Fallen

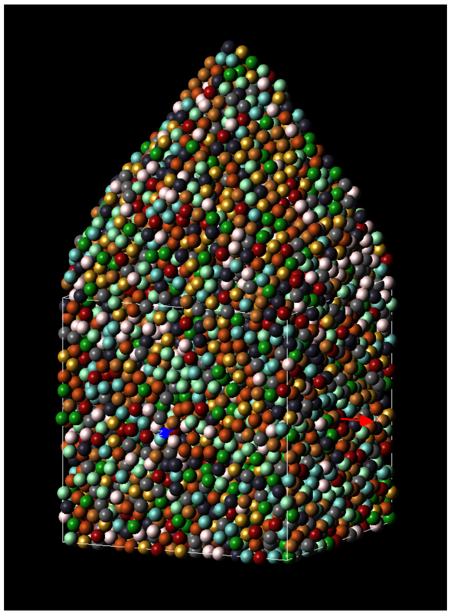


Figure 5.3: 9000 Particle Pack, All Particles Fallen

| **Number of Particles** | **Model Runtime (s)** | **Packing Fraction** |
|---|---|---|
| 150 | 138 | .6160 |
| 300 | 392 | .6221 |
| 750 | 1348 | .6179 |
| 1002 | 2140 | .6160 |
| 2001 | 5164 | .6083 |
| 3000 | 8965 | .6053 |
| 6000 | 23520 | .6061 |
| 9000 | 41424 | .6020 |

Table 5.1: Single Column Result Summary

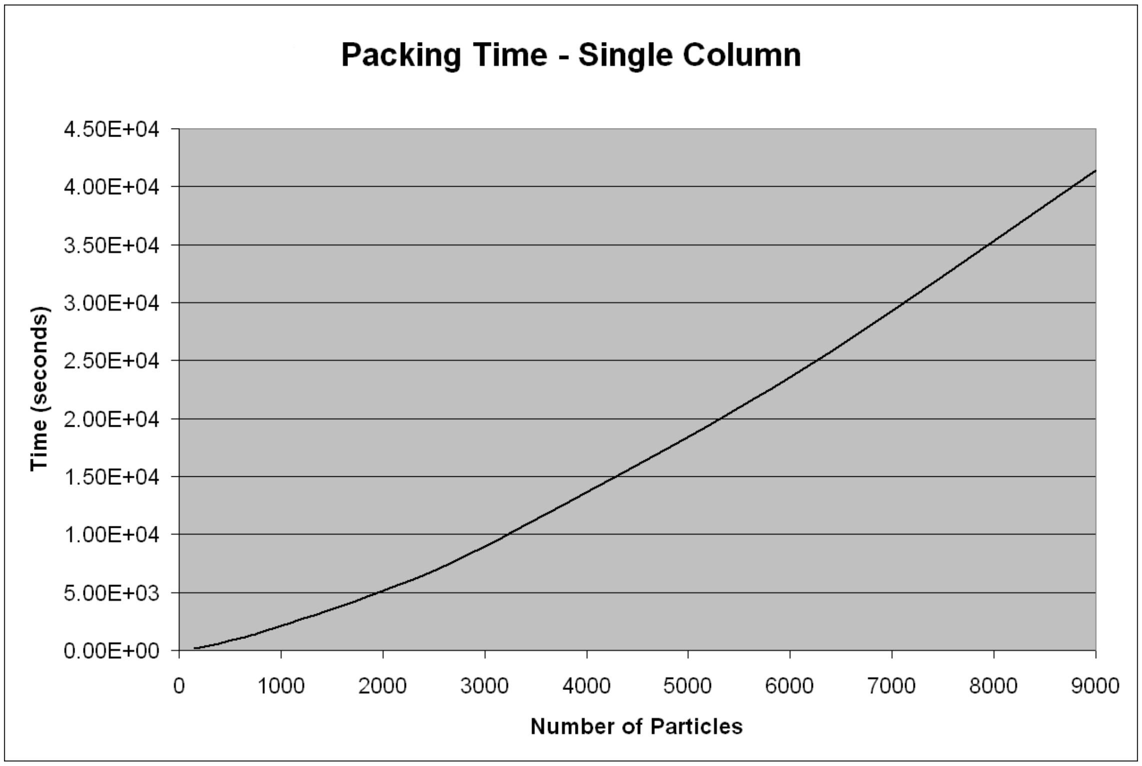


Figure 5.4: Single Column – Model Speed

## 5.2 Speed

The slope of time vs. number of particles graph, in Figure 5.4, follows a $n^2 * log(n)$ trajectory. The line is very smooth with no outlier points. Even with the regularity, and presumed predictability, of time it takes to make a packing run, the total processing time is disappointing. The 9000 particle pack is around 41,000 seconds, approximately half a day. This is much too slow for combinatorial chemistry experiments.

## 5.3 Packing Fraction

The packing fraction (Figure 5.5) varies from $\approx .62$ to $\approx .60$. Changes in the packing fraction are expected. The packing fraction will vary as the relationship between the width of the boundaries and the size of the particles changes. For any particular size of particle, there are several sizes of volumes (exact multiples of the particles' diameters) that will give a tighter packing. For example, even with 'perfectly tight' packings, there can be differences in the packing fraction. Among the types of close, non-random packing is Body-Centered Cubic (BCC) and Face-Centered Cubic (FCC). An example of the particle relationship in BCC packing is shown in Figure 5.6. A BCC pack gives a packing fraction of $\approx .68$. An FCC pack, shown in Figure 5.7, gives a packing fraction of $\approx .74$.

The packing fraction bounces up and down as expected. The graph line is still moving too much from 3,000 to 6,000 to 9,000 particles, so it is difficult to approximate where the final packing fraction would be as the number of particles reaches $\infty$. Most likely, it will be far less than the .63 target that was the goal.

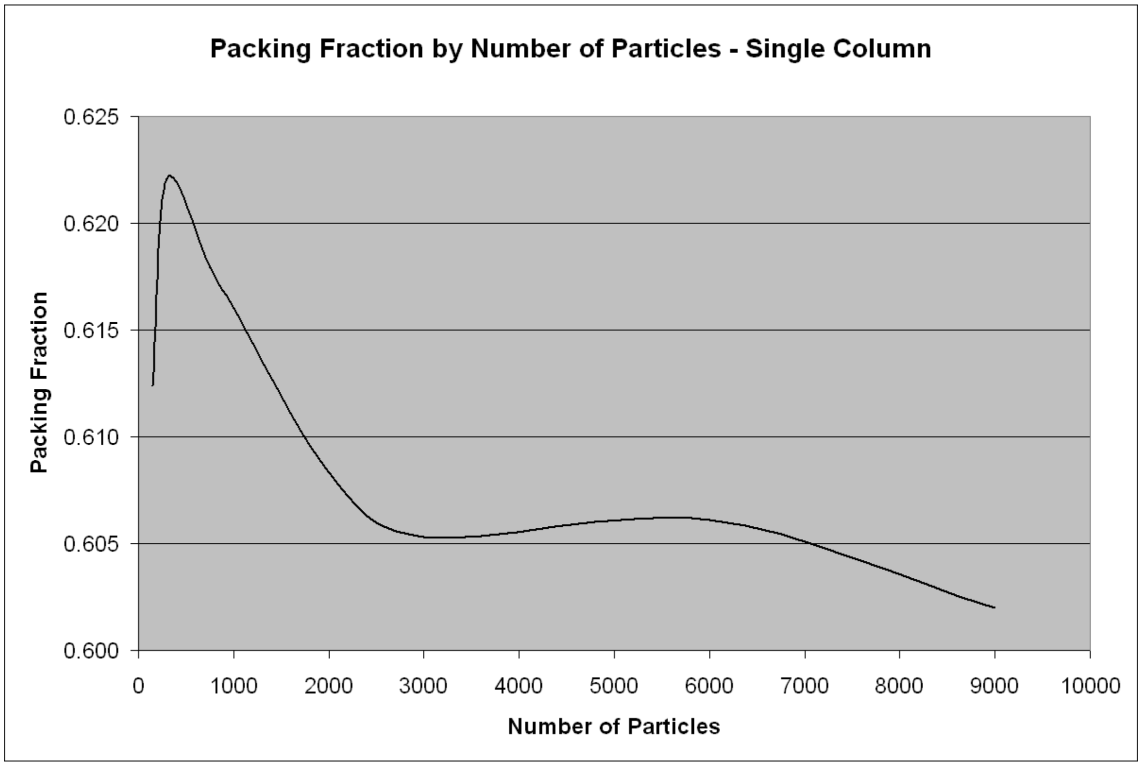


Figure 5.5: Single Column – Packing Fraction

## 5.4 Radial Distribution Function

The RDF graph for this model, Figure 5.8, looks decent at first glance, but upon closer inspection, there are two disturbing features about it. It definitely has a string peak at diameter one, and smaller peaks at two and three, but the peaks at two and three are advanced or early. Also, there's another anomaly. There are two very small peaks at 0.6 and 0.75. Theoretically, this is impossible. If there are peaks before one, it means that there are particles whose centers are closer together than the sum of their radii. In other words, they are interfering in each other's space. This means that there's an unresolved collision, in this

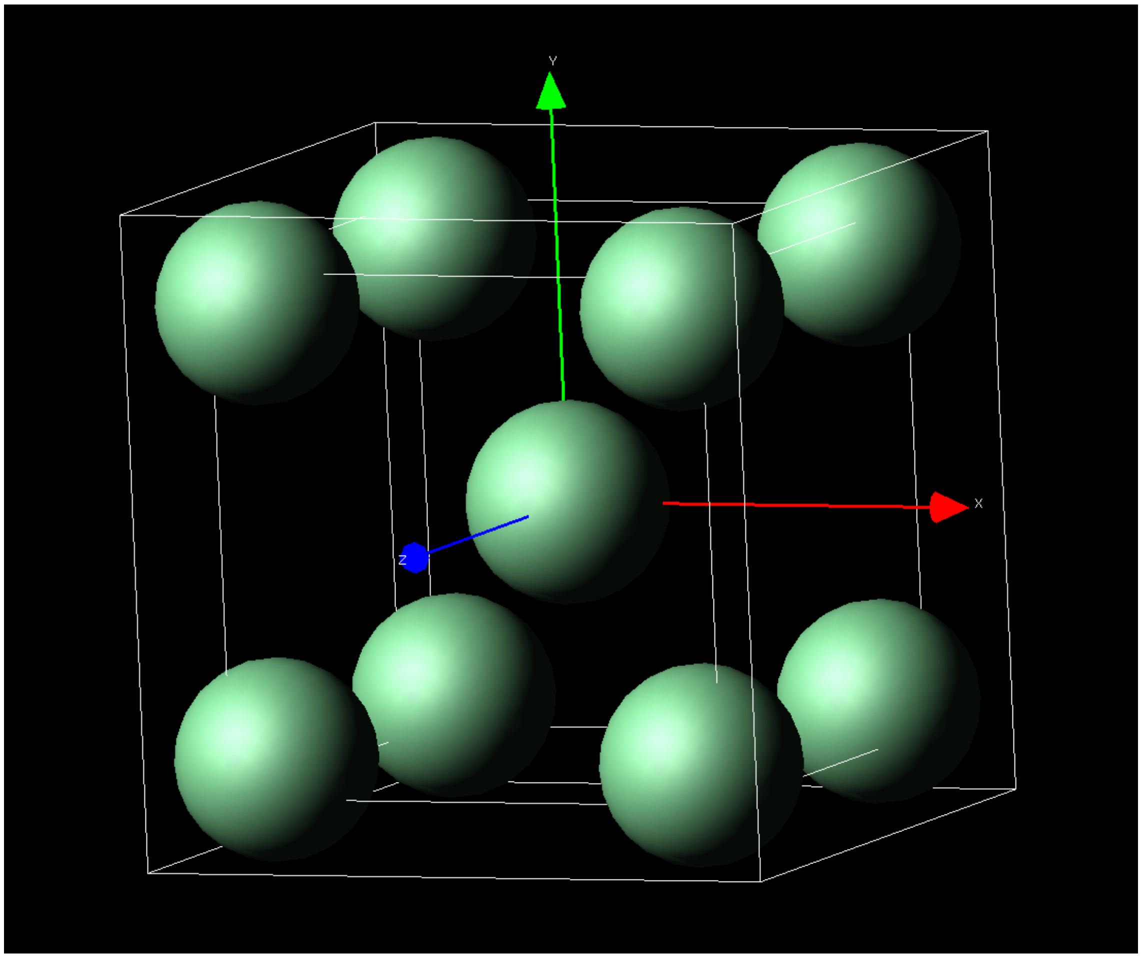


Figure 5.6: Body Centered Cubic Packing Example

case, two, because there are two peaks before one.

Looking into it further, as an example, it looks like particles number 347 and 463 are in an unresolved collision state. Figure 5.9 shows the completed 9,000 particle pack with only the two collided particles. The white dots represent the other particles in the pack. Figure 5.10 shows the offending particles much closer, so the collision can be clearly seen.

The first question at this point is, “What does this mean for the validity

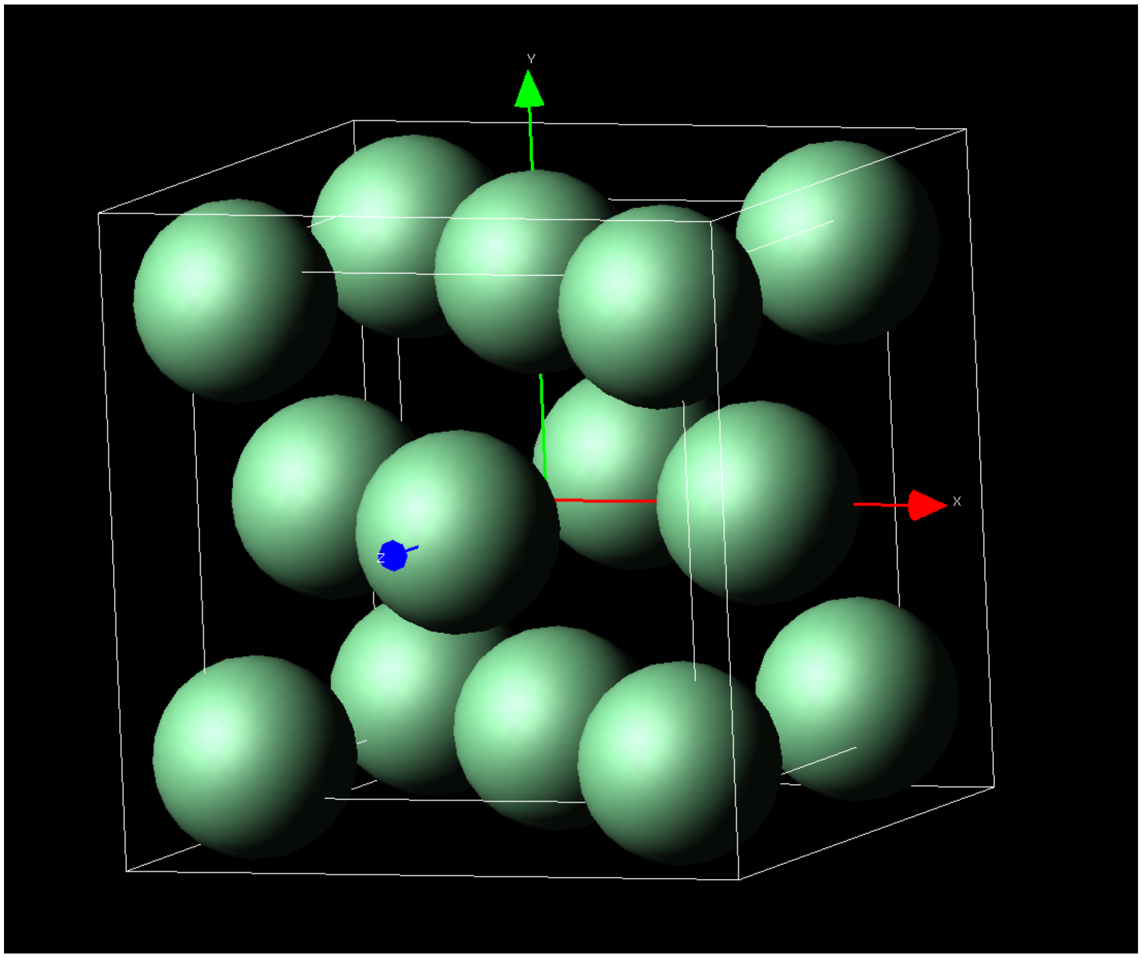


Figure 5.7: Face Centered Cubic Packing Example

of the rest of the data?" It means that the packing fraction is slightly over-calculated; specifically, the way that the packing fraction is calculated, the volume each particle contributes to the total particle volume inside the control volume. Therefore, the volume of intersection of any collided particles is counted twice. In the estimation of this researcher, it probably doesn't over-contribute very much, maybe not even much more than the order of .01 packing fraction's worth.

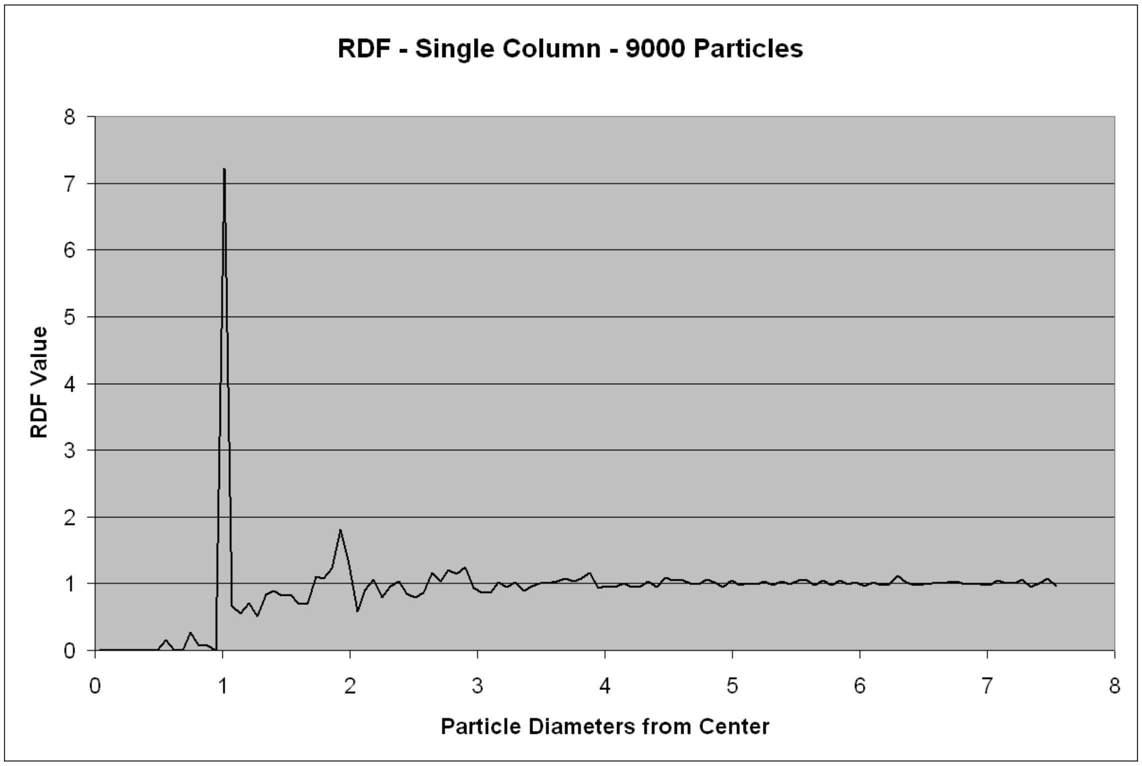


Figure 5.8: Single Column – Radial Distribution Function

As far as the speed of the calculation is concerned, there's probably an even greater effect, but that has to do with what the true problem is. It's most likely the neighbor list calculation. Remember that the neighbor list calculation is of $O\left(n^2 \log n\right)$ complexity. Because of this, the neighbor list calculation is done as infrequently as possible. In this case, it may have been done too infrequently, and a particle near 'Particle A' was not counted as a neighbor and, therefore, not checked for a collision in the circular sweep. In effect, if the neighbor list needs to capture more particles (causing more collision checks per sweep) or be done more often, the speed will be adversely affected. Almost certainly, the time it takes to do a run would increase. Only a lengthy, thorough investigation will discover the true culprit and the final effect on the results. At any rate, the comparison from one starting arrangement to another is valid as they have the same apparently faulty algorithm implementation.

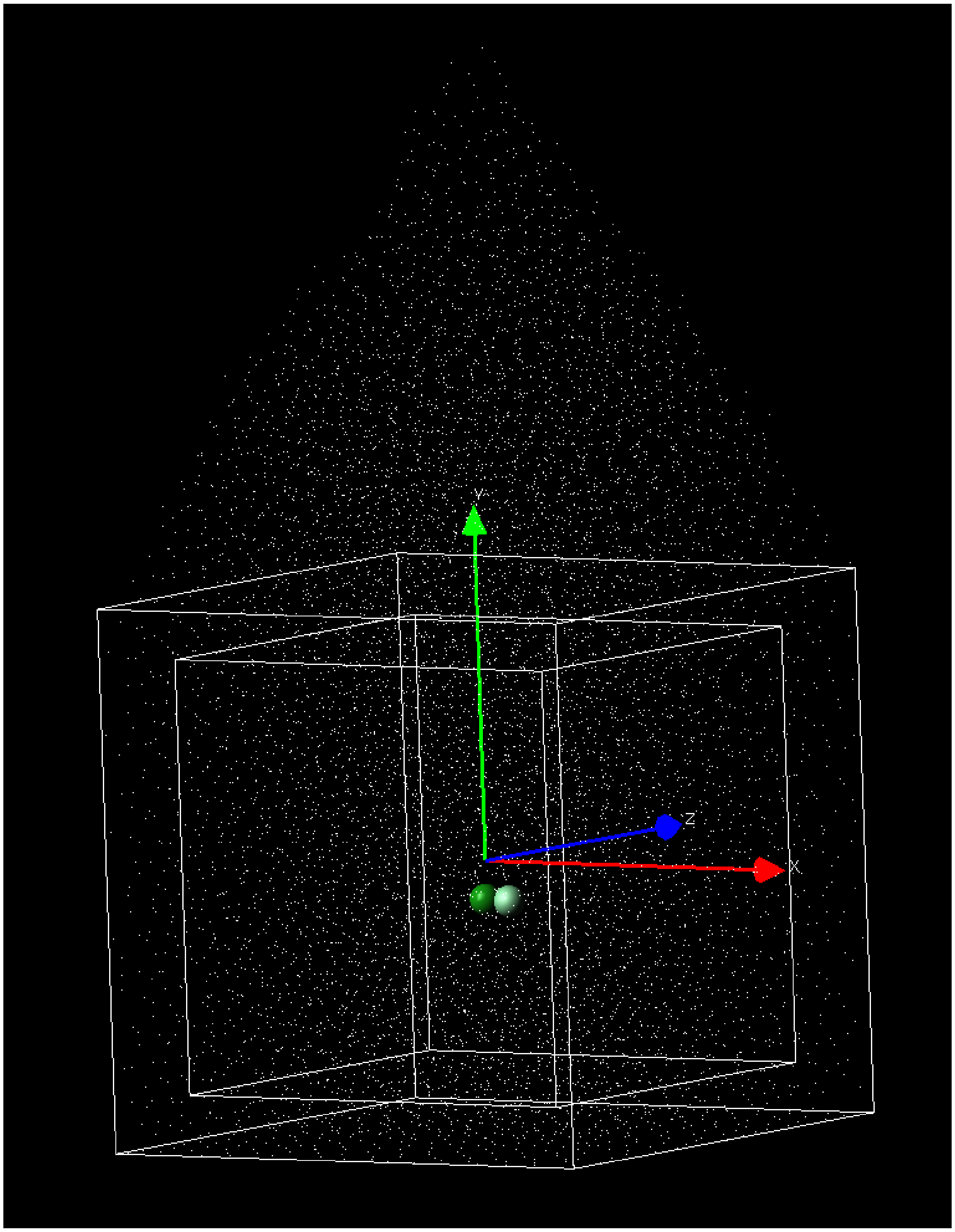


Figure 5.9: Single Column, 9000 Particles, Particle Number 347 and 463

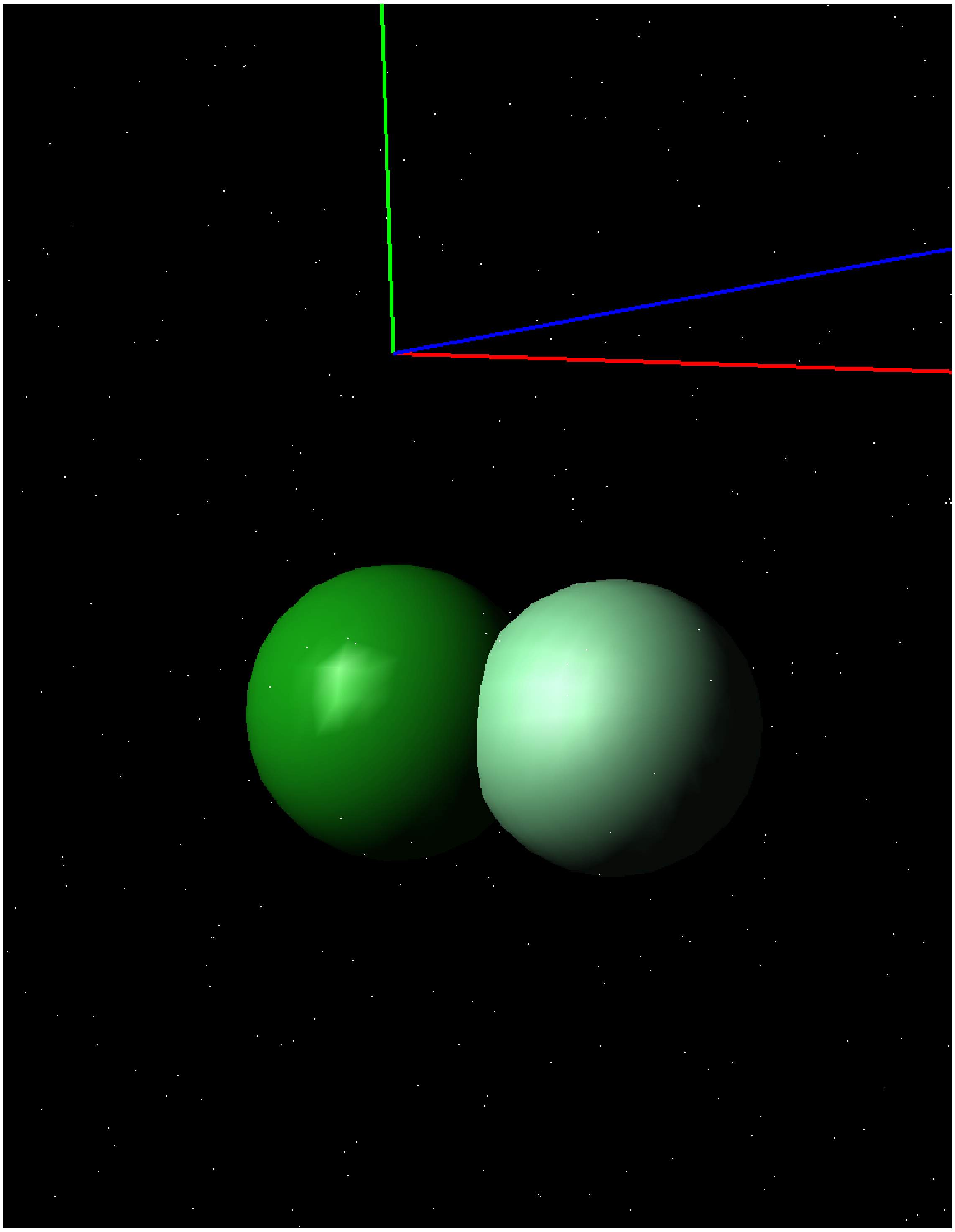


Figure 5.10: Single Column, 9000 Particles, Particle 347 and 463

# Chapter 6

# SGMP – Small Dense Starting Pack

## 6.1 Model Setup

The second starting pattern will be the Small Dense pack. Particles are randomly selected from the ten color bins and placed in the control volume in a repeating pattern. First, a calculation is made to determine how many particles (placed end-to-end) will fit inside the control volume. Then, the particles are placed in flat (fixed Y-Axis values) layers extending from one edge of the unit cube to the other. When a layer has been filled, the Y-Axis value is incremented and the next layer begins. The model was run eight times with different numbers of particles. For illustration, Figures 6.1 through 6.3 show the model at three different points. All three images are taken at the same position and zoom level. Table 6.1 shows the summary of the runtime and final packing fraction with various numbers of particles.

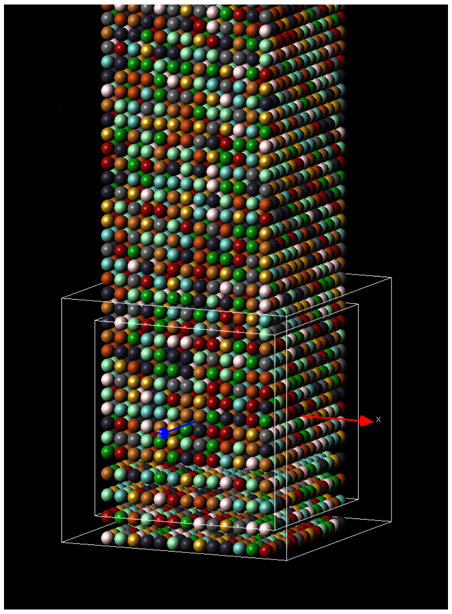


Figure 6.1: Small Dense Pack, 9000 Particles, 0 Particles Fallen

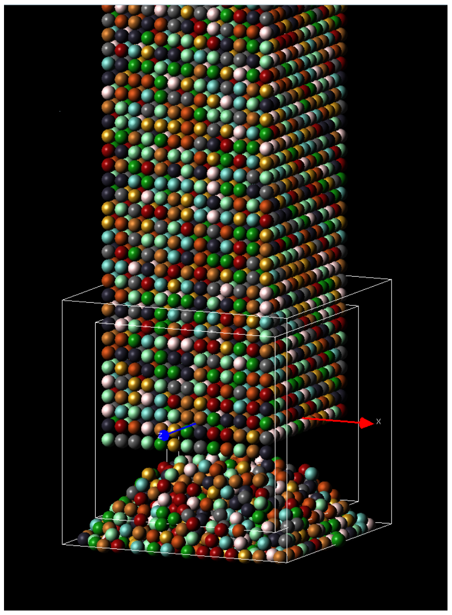


Figure 6.2: Small Dense Pack, 9000 Particles, 1000 Particles Fallen

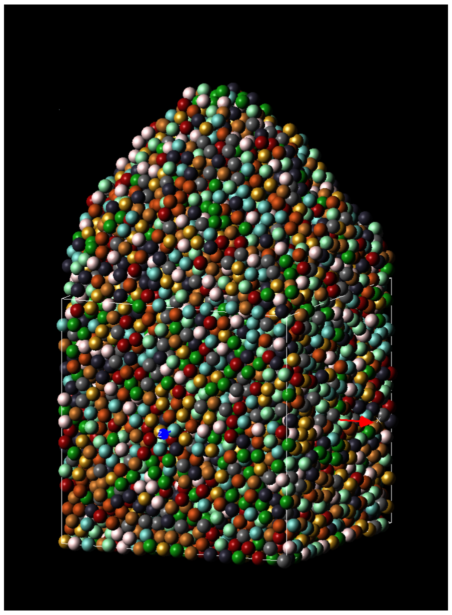


Figure 6.3: Small Dense Pack, 9000 Particles, All Particles Fallen

| **Number of Particles** | **Model Runtime (s)** | **Packing Fraction** |
|---|---|---|
| 150 | 264 | .5851 |
| 300 | 601 | .6067 |
| 750 | 2826 | .6092 |
| 1002 | 3565 | .5985 |
| 2001 | 7519 | .6027 |
| 3000 | 12746 | .5987 |
| 6000 | 37388 | .6038 |
| 9000 | 56337 | .6023 |

Table 6.1: Small Dense Result Summary

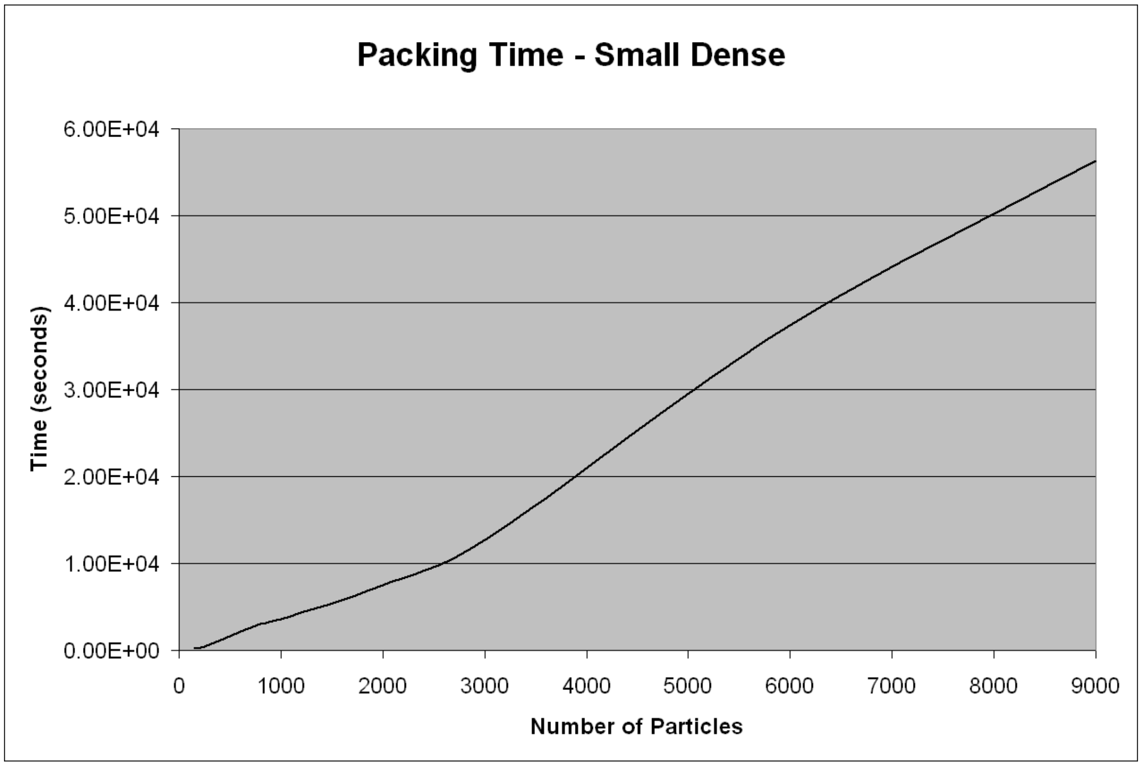


Figure 6.4: Small Dense – Speed

## 6.2 Speed

In general, the slope of the time vs. the number of particles graph line follows a $n^2 * log(n)$ trajectory. However, there are two points on the curve where the line is skewed upwards compared to the other points. Specifically, those are at 750 and 6000.

Looking at the bottom profile of the pack before the model starts, it can be seen that the 750 pack and the 6000 pack have a good deal of extra space inside the control volume. There is extra space, but not quite enough to fit an extra row and column of particles. Figure 6.5 shows each pack's starting position

| **Number of Particles** | **Grid Size** | **Height of Starting Pack** |
|---|---|---|
| 150 | 3x3 | 4.56 |
| 300 | 4x4 | 3.95 |
| 750 | 5x5 | 4.61 |
| 1002 | 6x6 | 3.82 |
| 2001 | 8x8 | 3.40 |
| 3000 | 9x9 | 3.53 |
| 6000 | 11x11 | 3.69 |
| 9000 | 13x13 | 3.40 |

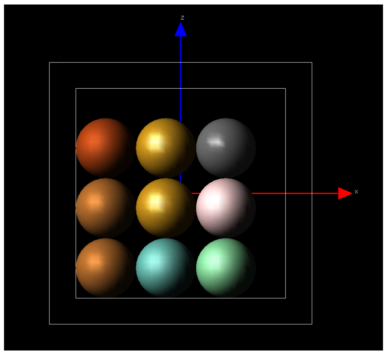
Table 6.2: Small Dense Grid Size and Starting Pack Height Summary

from below so the extra space can be seen.

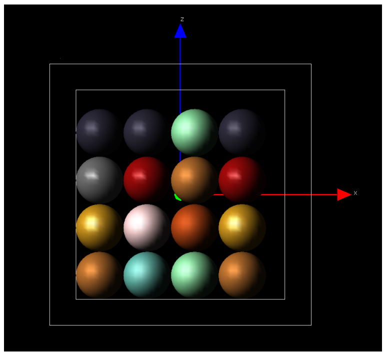
Extra space in the starting grid causes the starting pack to be higher; therefore, the particles in total, will have farther to fall. Figure 6.2 shows the grid size and starting height of each pack.

From these details, this author would conclude that the model is very "starting-height dependent." The effect is compounded as the number of particles increases. The starting height difference between 2001, 3000, 6000, and 9000 is small, but as the number of particles gets relatively large, the effect is significant.

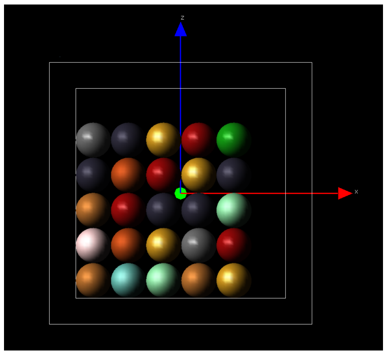
Generally, the speed in general is decent, but not near where we need to be in order to process many, many runs for combinatorial chemistry applications. The 9000 particle run was on the order of 55,000 seconds or close to .65 days. This is far too slow for realistic use.



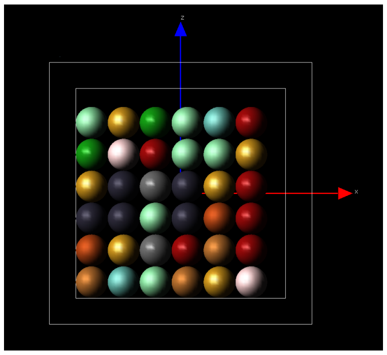
(a) 150 particles



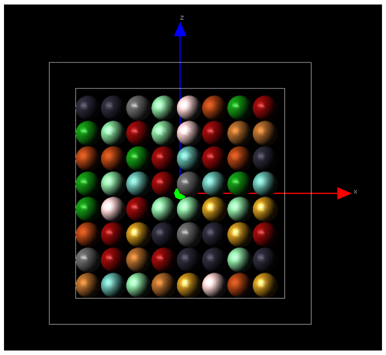
(b) 300 particles



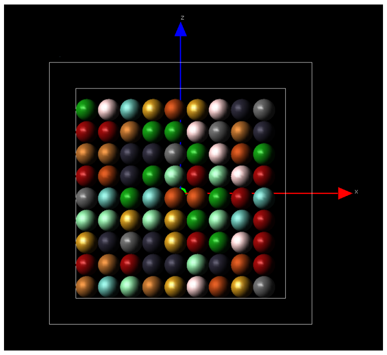
(c) 750 particles



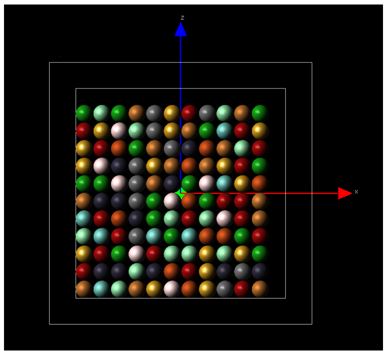
(d) 1002 particles



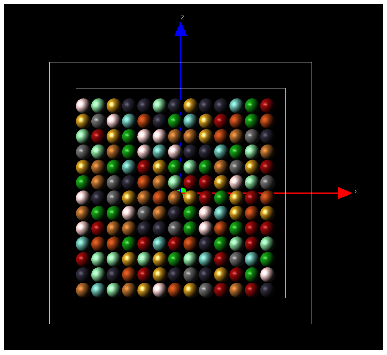
(e) 2001 particles



(f) 3000 particles



(g) 6000 particles



(h) 9000 particles

Figure 6.5: Small Dense Pack, From Below

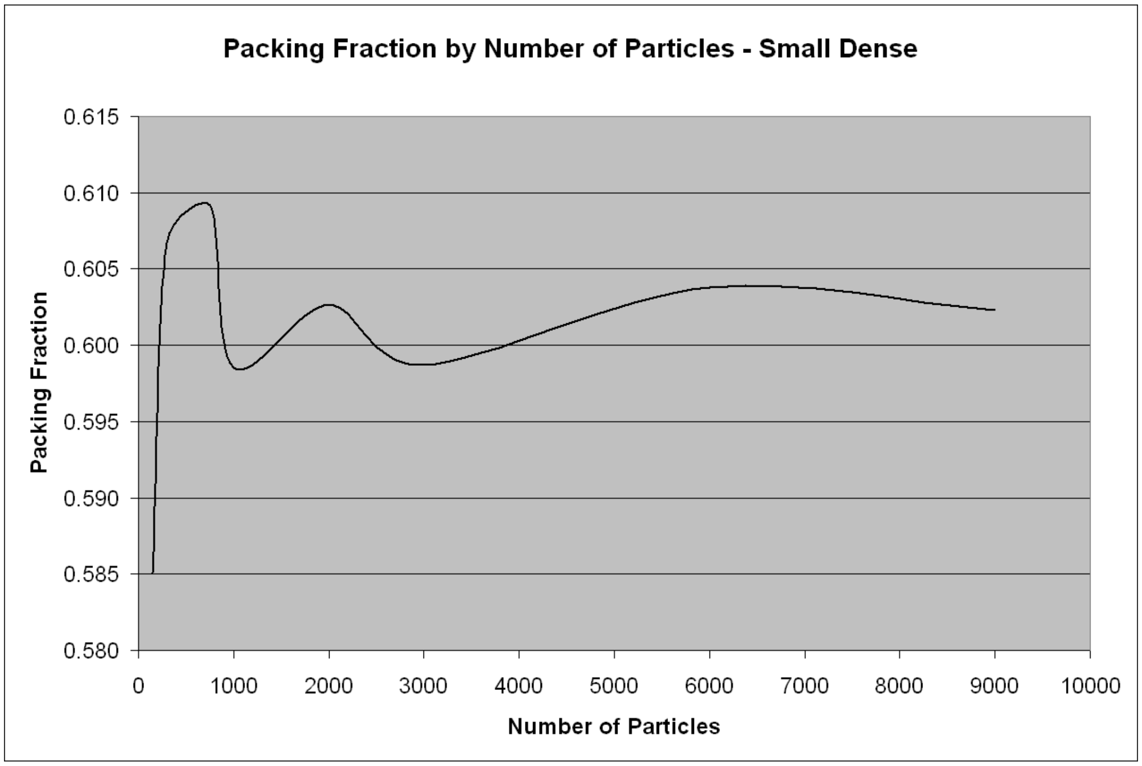


Figure 6.6: Small Dense – Packing Fraction

## 6.3 Packing Fraction

The packing fraction varies from $\approx .58$ to $\approx .61$. It varies up and down as expected since the number of particles spanning the volume change. It is hard to be sure without more data, but it looks like the final packing fraction (as the number of particles reaches $\infty$) would be nearly .60. This is not terrible, but considering the slowness by which this was reached, it's not very useful in the applications for which it is designed.

## 6.4 Radial Distribution Function

The RDF graph for this model, Figure 6.7, looks very normal in that it has the expected characteristic peaks near diameters 1, 2, and 3, with each a little smaller than the former one. By the time we hit the fourth diameter, a peak is difficult to discern at all, giving us the required lack of periodicity. The distressing part is that the unresolved collision issue discussed in the Single Column model analysis is present as well. It's not quite as evident here as it is in Single Column, as there's only one very small peak before diameter 1, but the peaks at 2 and 3 are advanced (early) at about the same rate as in the Single Column model.

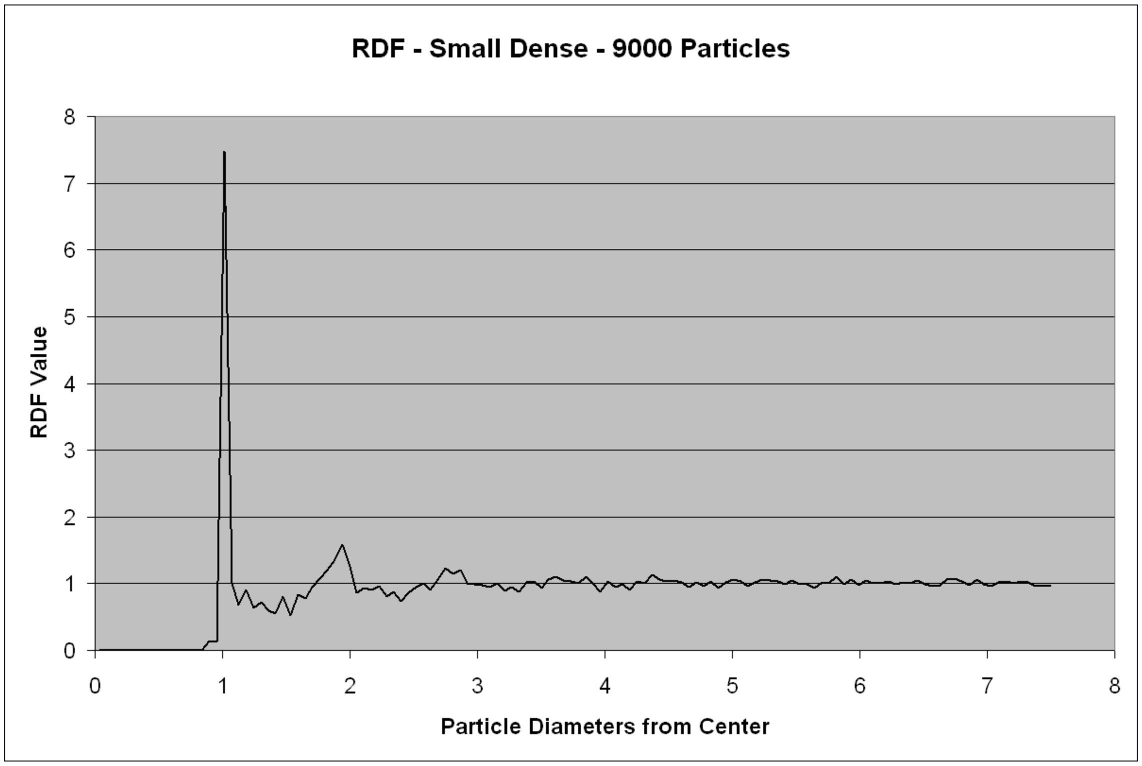


Figure 6.7: Small Dense – Radial Distribution Function

# Chapter 7

# SGMP – Large Dense Starting Pack

## 7.1 Model Setup

The third starting pattern is the Large Dense starting pack. Particles are randomly selected from the ten color bins and placed in the control volume in a repeating pattern. First, a calculation is made to determine how many particles (end-to-end) will fit inside the extended volume. The particles are then placed in flat (fixed Y-Axis value) layers extending from one edge of the extended volume to the other. When a layer has been filled, the Y-Axis value is incremented and the next layer begins. The model was run eight times with different numbers of particles. The results of the runs are summarized in Table 7.1. For illustration, the images of the 9,000 particle pack at three points are shown in Figures 7.1 through 7.3. All three images are taken at the same zoom level and position.

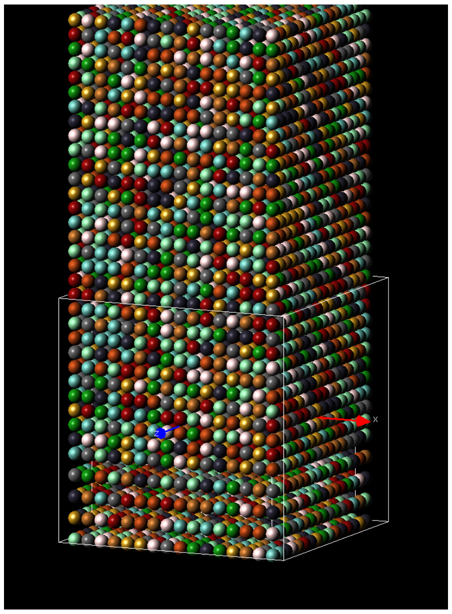


Figure 7.1: Large Dense Pack, 9000 Particles, 0 Particles Fallen

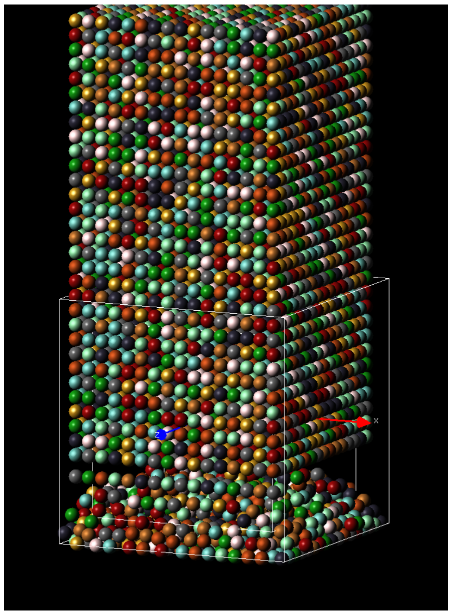


Figure 7.2: Large Dense Pack, 9000 Particles, 1000 Particles Fallen

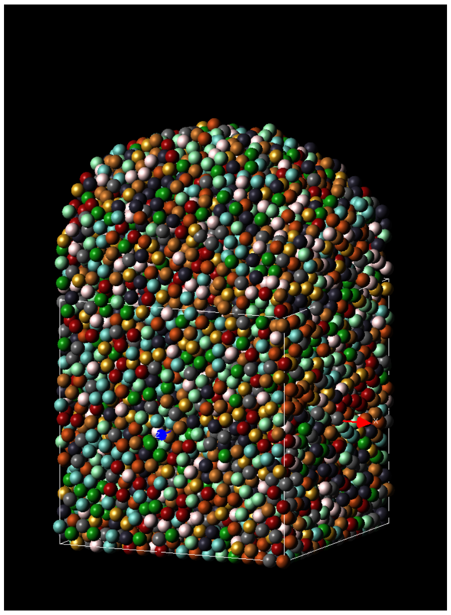


Figure 7.3: Large Dense Pack, 9000 Particles, All Particles Fallen

| **Number of Particles** | Model Runtime (s) | **Packing Fraction** |
|---|---|---|
| 150 | 94 | .5694 |
| 300 | 263 | .5910 |
| 750 | 733 | .5865 |
| 1002 | 999 | .5863 |
| 2001 | 2424 | .5850 |
| 3000 | 4864 | .5984 |
| 6000 | 11463 | .5971 |
| 9000 | 19699 | .5928 |

Table 7.1: Large Dense Result Summary

| Number of Particles | Grid Size | Height of Starting Pack |
|---|---|---|
| 150 | 4x4 | 2.54 |
| 300 | 5x4 | 2.35 |
| 750 | 7x7 | 2.24 |
| 1002 | 8x8 | 1.78 |
| 2001 | 10x10 | 2.05 |
| 3000 | 11x11 | 2.14 |
| 6000 | 14x14 | 2.08 |
| 9000 | 16x16 | 2.11 |

Table 7.2: Large Dense Grid Size and Starting Pack Height Summary

## 7.2 Speed

As expected by the order of operations analysis, the slope of the time vs. number of particles graph follows a $n^2 * log(n)$ trajectory. Like its cousin, the Small Dense model, the graph line is not perfectly smooth. It suffers from the same problem: the particles have to fall from a higher distance in some of the starting positions. It's not as pronounced as the time variations in the Small Dense model, but there is a bit of high point in the 2001-3000 range, where the starting heights are slightly elevated compared to their neighbors, as can be seen in Figure 7.2. The numbers are only a bit higher than their neighbors, so there could be some other factor at work distorting the graph, but what that might be is not apparent.

While this is the fastest model so far, the speed is still unremarkable. The 9000 particle pack took about 20,000 seconds, close to a quarter of a day. This is not fast enough for the needed applications.

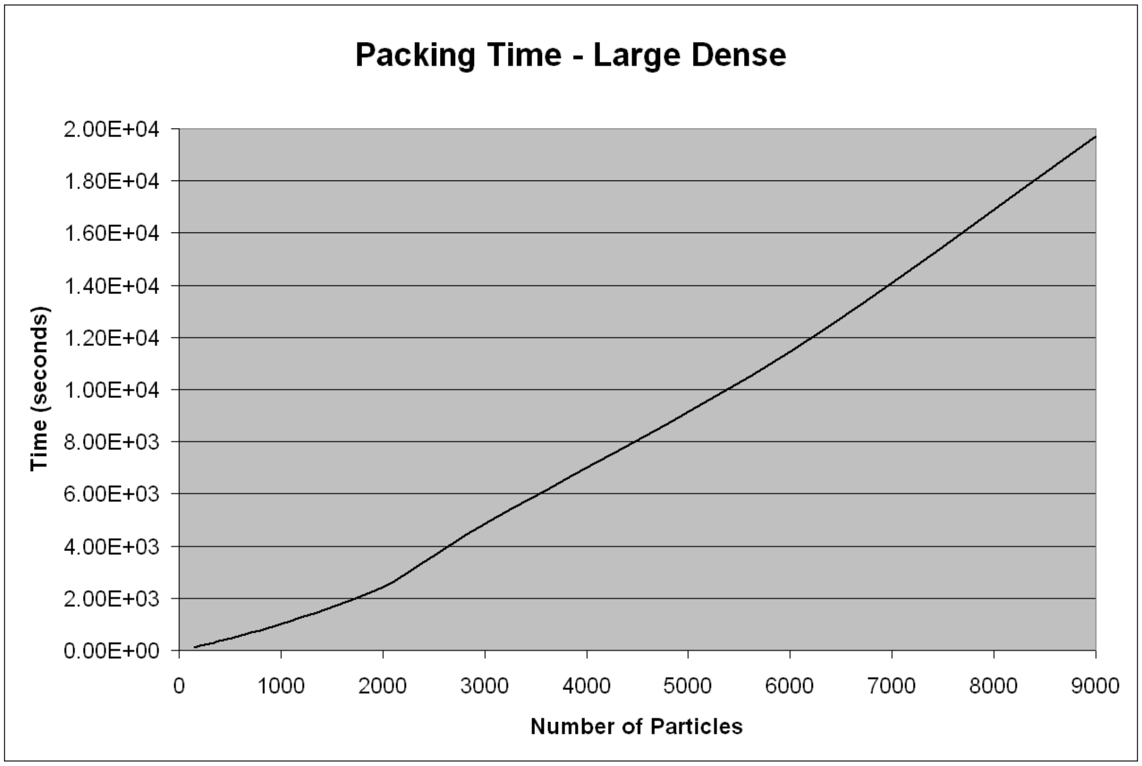


Figure 7.4: Large Dense – Model Speed

## 7.3 Packing Fraction

The packing fraction varies from $\approx .56$ to $\approx .60$. As seen in Figure 7.5, it rises and falls as expected as the number of particles spanning the volume changes. This model, more than the previous ones, has a more extreme variation. This is, most likely, due to a lack of data points with the other models. This model came closer to hitting the packing fraction extremes than the others. In this model, there also seems to be quite a bit of movement from 3000 to 9000 particles, making it difficult to estimate where the final packing fraction might be. At any rate, the final packing fraction (as the number of particles reaches $\infty$) will almost assuredly be below .60. This, like the previous models, is not close enough to the target of .63 to be considered useful.

## 7.4 Radial Distribution Function

The RDF graph for this model, Figure 7.6, looks almost exactly like that of the RDF graph for the Small Dense model. There's a small peak just before the major peak at diameter one. Also, there are the diminishing, but slightly early, peaks at diameters two and three. Basically, the graphs show two things: the model provides sufficiently random non-periodicity and unresolved collisions are still a problem.

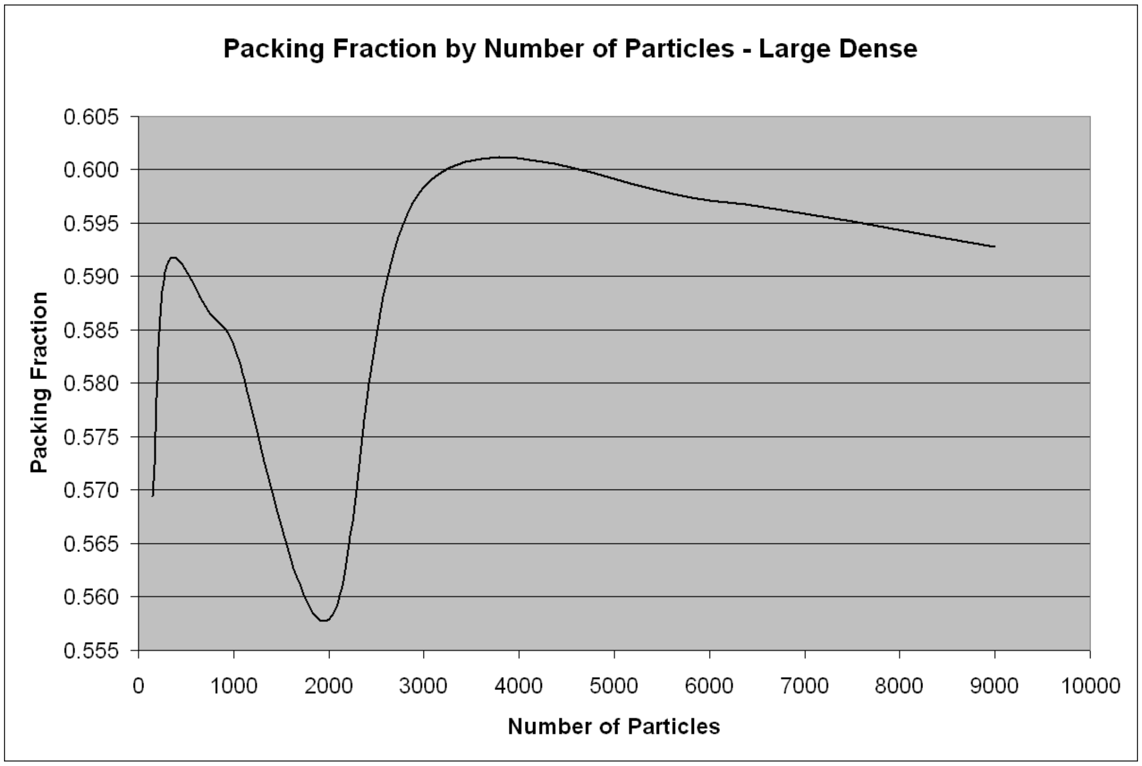


Figure 7.5: Large Dense – Packing Fraction

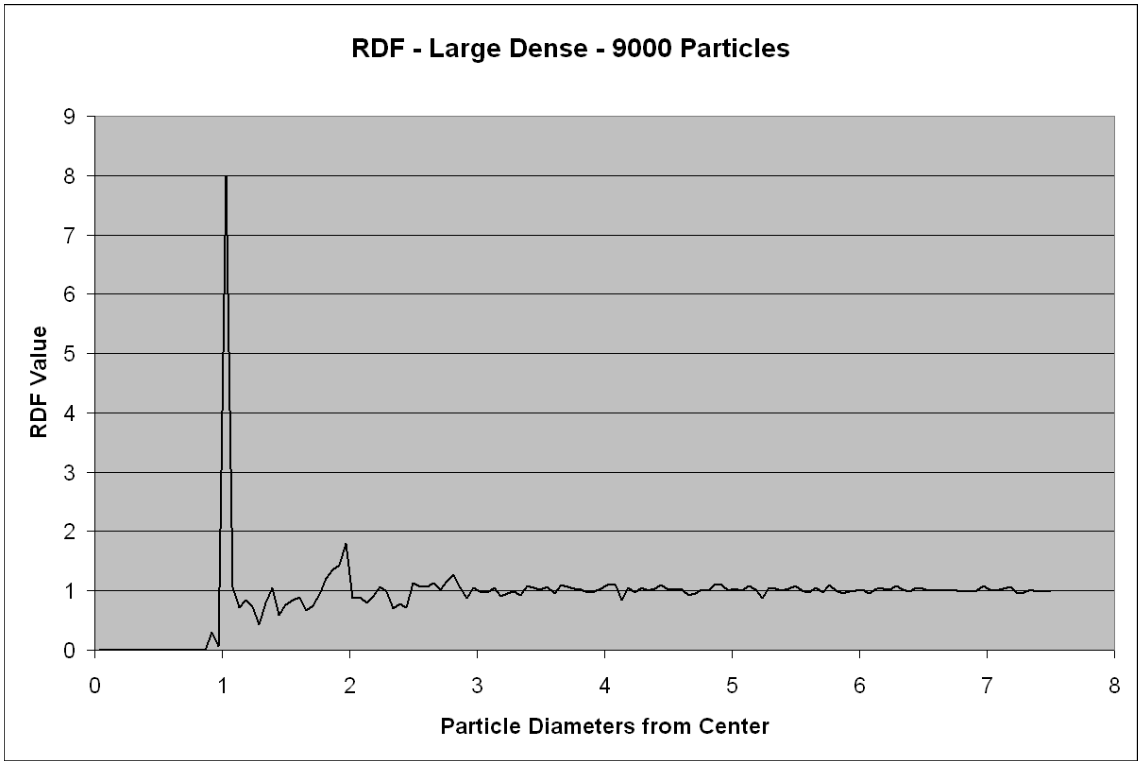


Figure 7.6: Large Dense – Radial Distribution Function

# Chapter 8

# SGMP – Loose Random Starting Pack

## 8.1 Model Setup

The final starting arrangement is the Loose Random pack. Again, particles are randomly selected from the ten color bins and placed in the control volume in a random pattern. Particles are assigned a random X-Axis and Z-Axis value and placed into the extended volume. If the particle collides with a previously placed particle, its Y-Axis value is increased until the collision is resolved. This cycle is repeated until the particle bins are empty.

The model was run eight times with different numbers of particles and the results of the runs are summarized in Table 8.1. For illustration, the images of the 9000 particle pack at three points are shown in Figures 8.1 through 8.3. All three images are taken at the same position and zoom level.

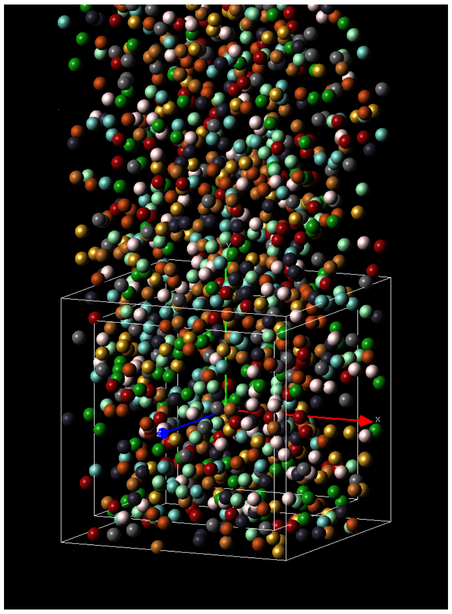


Figure 8.1: Loose Random Pack, 9000 Particles, 0 Particles Fallen

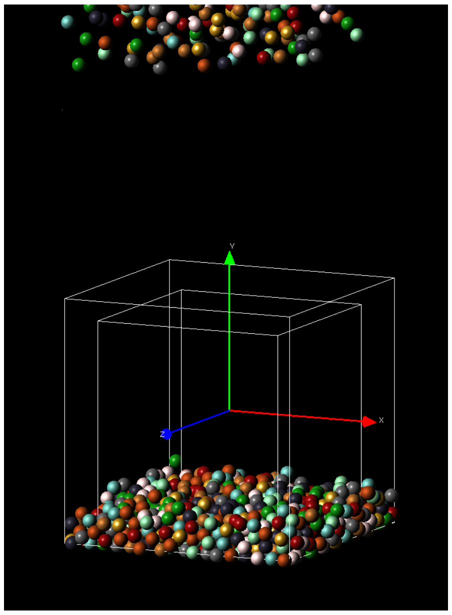


Figure 8.2: Loose Random Pack, 9000 Particles, 1000 Particles Fallen

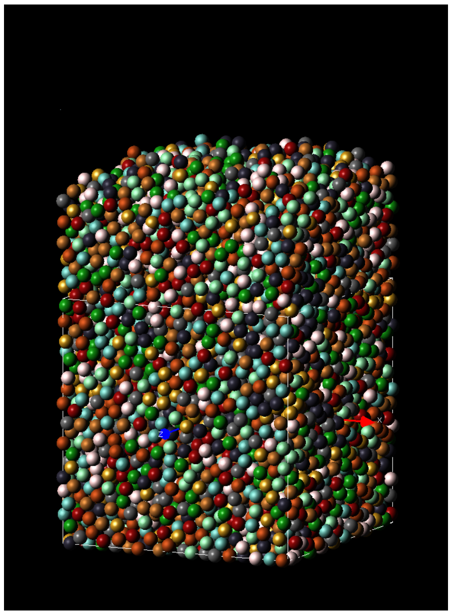


Figure 8.3: Loose Random Pack, 9000 Particles, All Particles Fallen

| **Number of Particles** | Model Runtime (s) | **Packing Fraction** |
|---|---|---|
| 150 | 127 | .5861 |
| 300 | 551 | .5918 |
| 750 | 3456 | .5892 |
| 1002 | 5888 | .5913 |
| 2001 | 23120 | .5904 |
| 3000 | 54234 | .5856 |
| 6000 | 205797 | .5904 |
| 9000 | 456715 | .5897 |

Table 8.1: Loose Random Result Summary

| Number of Particles | Height of Starting Pack |
|---|---|
| 150 | 3.38 |
| 300 | 4.71 |
| 750 | 7.02 |
| 1002 | 7.72 |
| 2001 | 10.4 |
| 3000 | 12.9 |
| 6000 | 17.9 |
| 9000 | 21.7 |

Table 8.2: Loose Random Grid Size and Starting Pack Height Summary

## 8.2 Speed

Like the previous models, the slope of the time vs. number of particles graph follows a $n^2 * log(n)$ trajectory. Unlike the dense models, the graph line is very smooth. The most striking thing about this model is the large amount of time it takes to do a run. The 9000 particle pack took close to 450,000 seconds to complete. That's over five days of computer time, which is unacceptably poor. Again, the author thinks the slowness comes from the height of fall. Looking at Figure 8.2, the height of the column for the 9000 particle starting arrangement is 21.7. That is an enormous height, especially compared to the other models.

## 8.3 Packing Fraction

Not only is the speed poor, but the resulting packing fraction (Figure 8.5) is not very good either. The packing fraction varies from $\approx .585$ to $\approx .592$. It varies

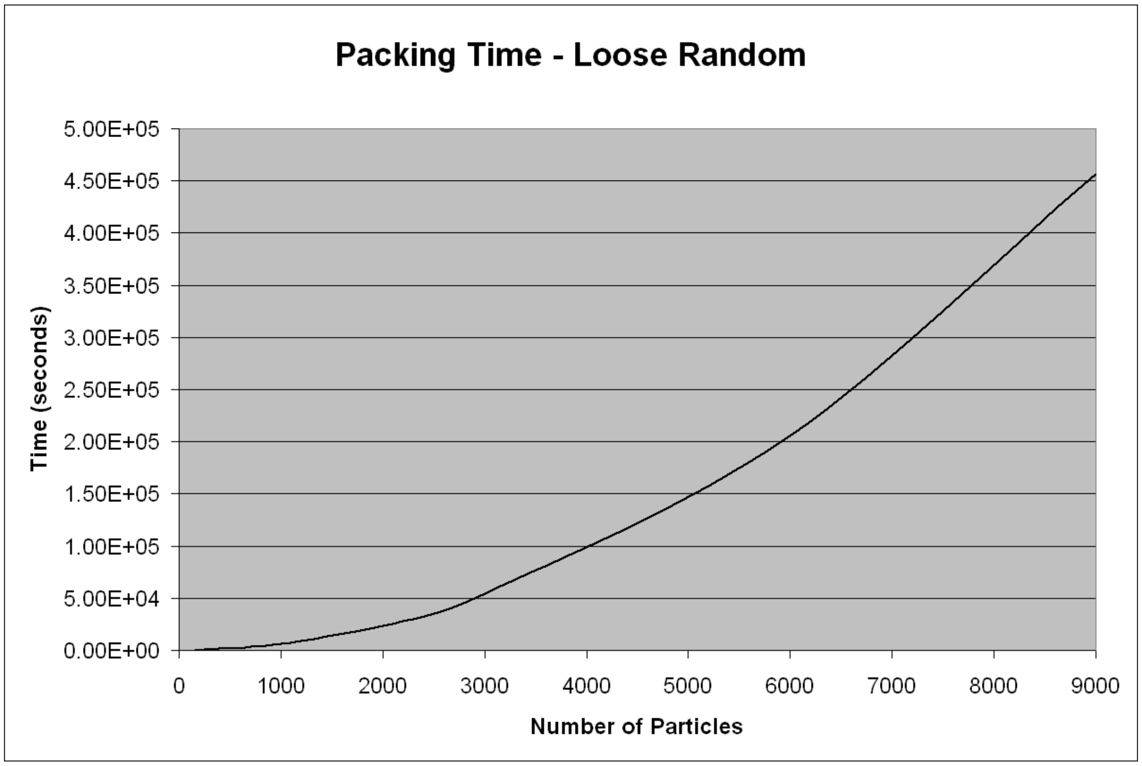


Figure 8.4: Loose Random – Model Speed

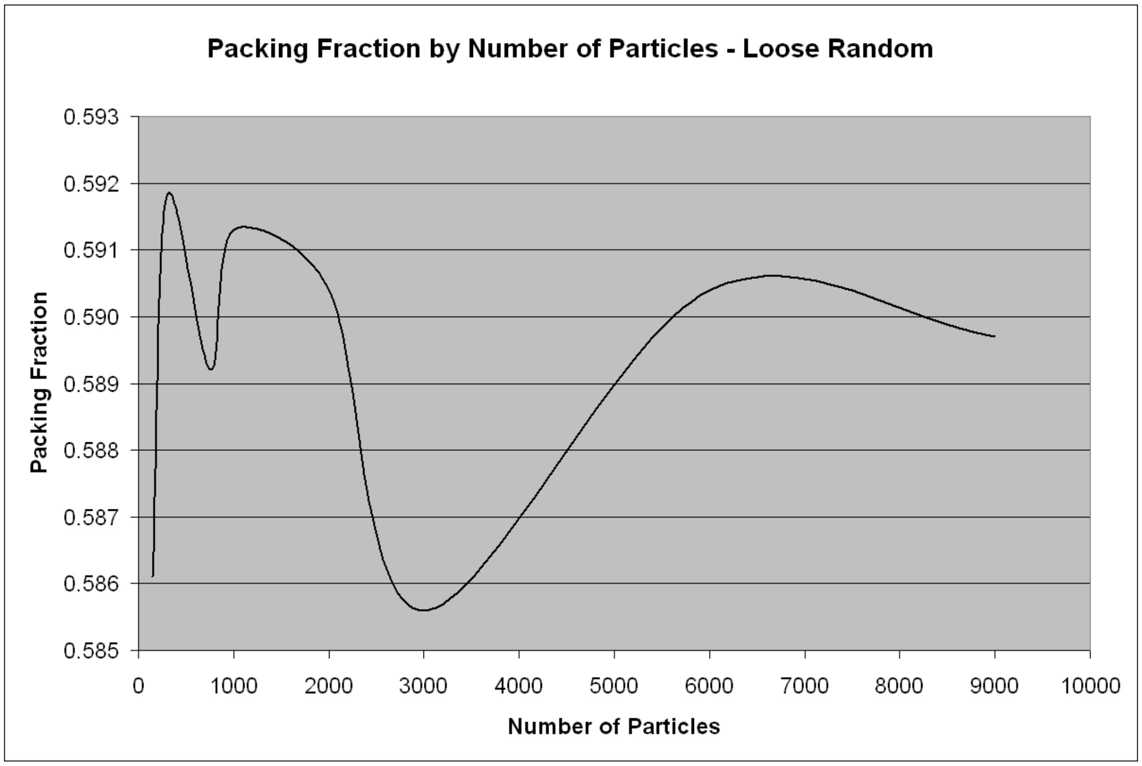


Figure 8.5: Loose Random – Packing Fraction

from high to low as the number of particles spanning the volume change. More interesting is that the range does not vary much when compared to the other models. Again, without further runs made, the final packing fraction (as the number of particles reaches $\infty$) will be no greater than .59.

## 8.4 Radial Distribution Function

The RDF graph for this model, Figure 8.6, looks normal, but seems to be noisier than the other models. This may be due to the lower packing fraction. If the packing fraction is lower, the chance for periodicity is lessened. There are still

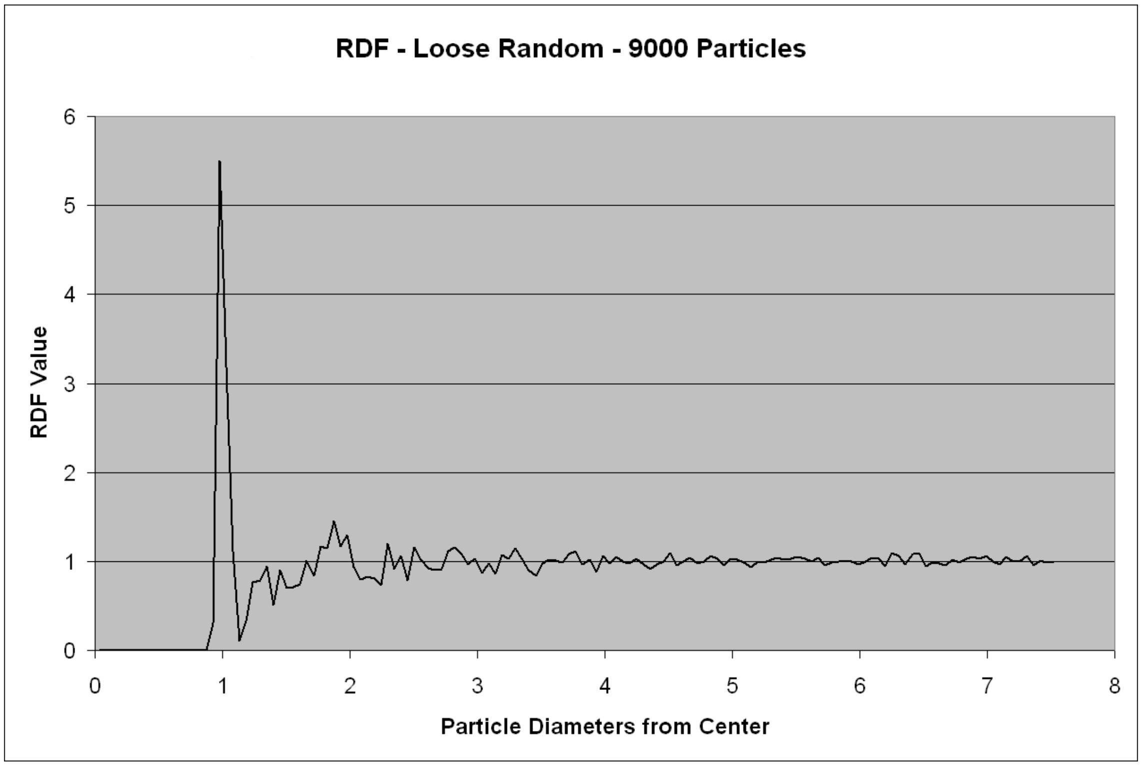


Figure 8.6: Loose Random – Radial Distribution Function

peaks at diameters one and two, but if there's a peak at three, it's difficult to discern. This graph shows signs of unresolved collisions in that the peak at diameter two appears early, meaning that, most likely, there are particles closer together than is physically possible.

# Chapter 9

# Comparison and Analysis

## 9.1 Speed

As far as speed in concerned, none of the models are effective. They all suffer the falling height issue. The way the SGMP algorithm is implemented, particles fall (decrease their Y-Axis values) at a fixed rate per fall cycle, a fall cycle being a downward movement followed by neighbor list collision checks. That value is .001, so it takes 1000 fall cycles to drop the length of the control volume. Looking at the graph, the Single Column is faster than the Small Dense model, but slower than the Large Dense. The reason is the fall phase is implemented differently in the Single Column than the others.

Specifically, in the Single Column model, after a particle finishes the SGMP phases, the entire column is moved down at one time in a single increment to just above the first point of contact of the last particle with the fixed pack. This

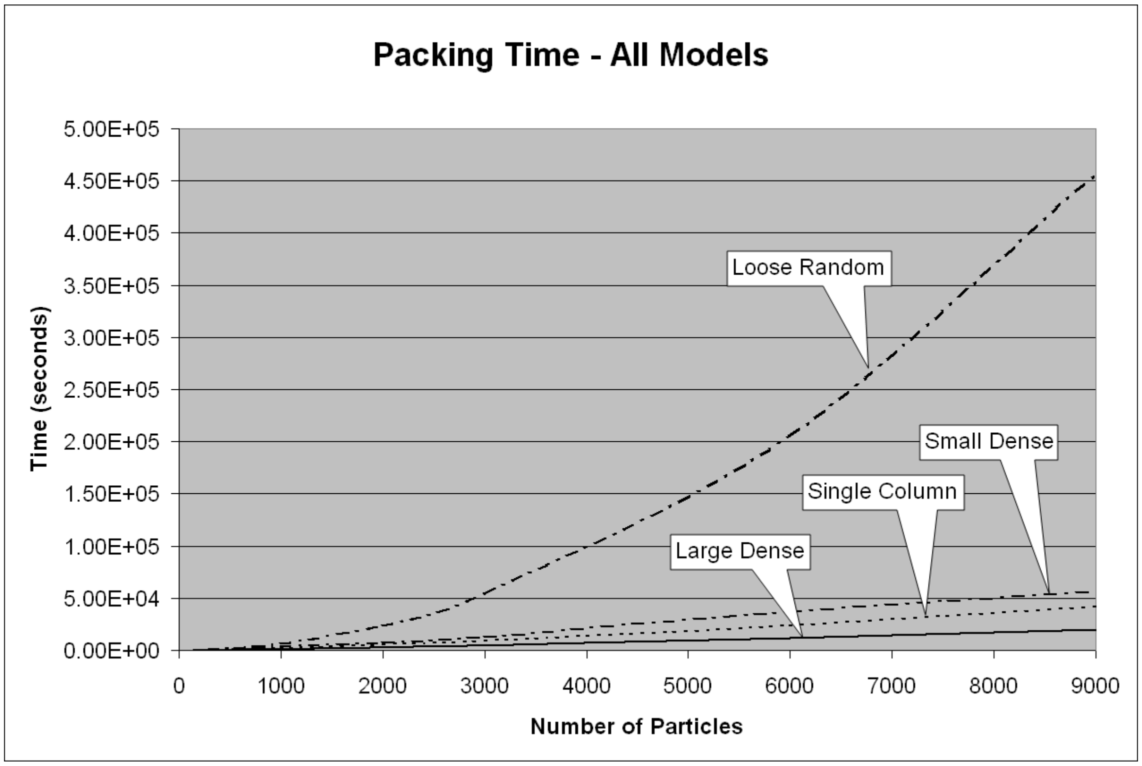


Figure 9.1: All Models – Speed

essentially removes most of the time spent in the falling phase. But if there's very little time spent in the falling phase getting to the initial collision with the fixed pack, why isn't the Single Column model faster than Large Dense?

The second difference between Single Column and Large Dense is the fixed particle bed that a falling particle comes into contact with. The shape of the fixed particle bed will determine how far, on average, a particle will move once it makes initial contact with a fixed particle. For example, in Figures 9.2 and 9.3, the shape of the fixed particle bed in the Single Column model is a steep pyramidal structure, giving the particle an earlier first contact as well as having

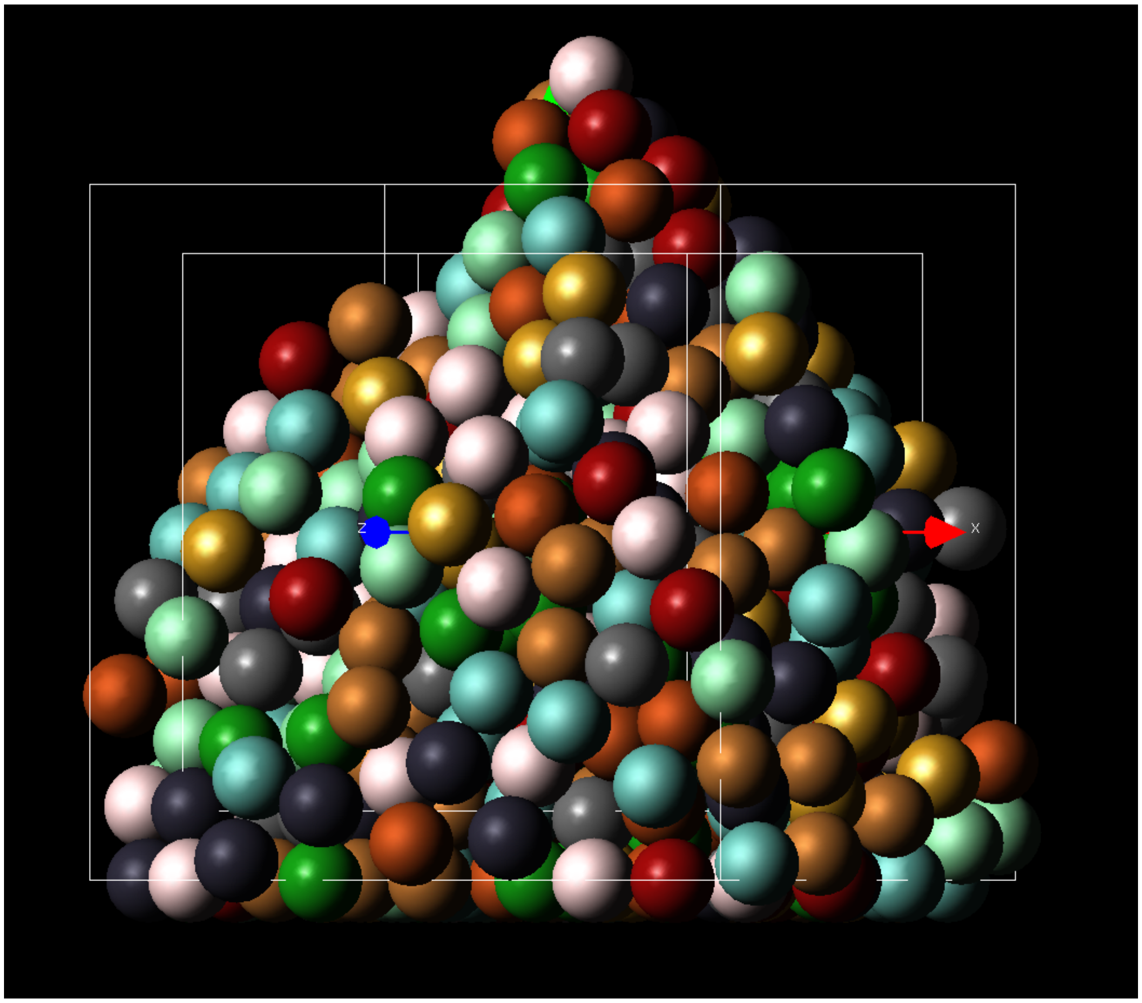


Figure 9.2: Fixed Particle Bed – Single Column – 500 out of 1002 Particles Fallen

farther to fall before others come to rest, on average. The shape of the bed in the Large Dense model is still pyramidal, but much less high and steep. Therefore, an average particle in the Large Dense model will have to go through fewer SGMP phases and come to rest sooner than in the Single Column model. This accounts for the faster pack times in the Large Dense model.

Also of interest is the observation that the pyramidal shape of the fixed par-

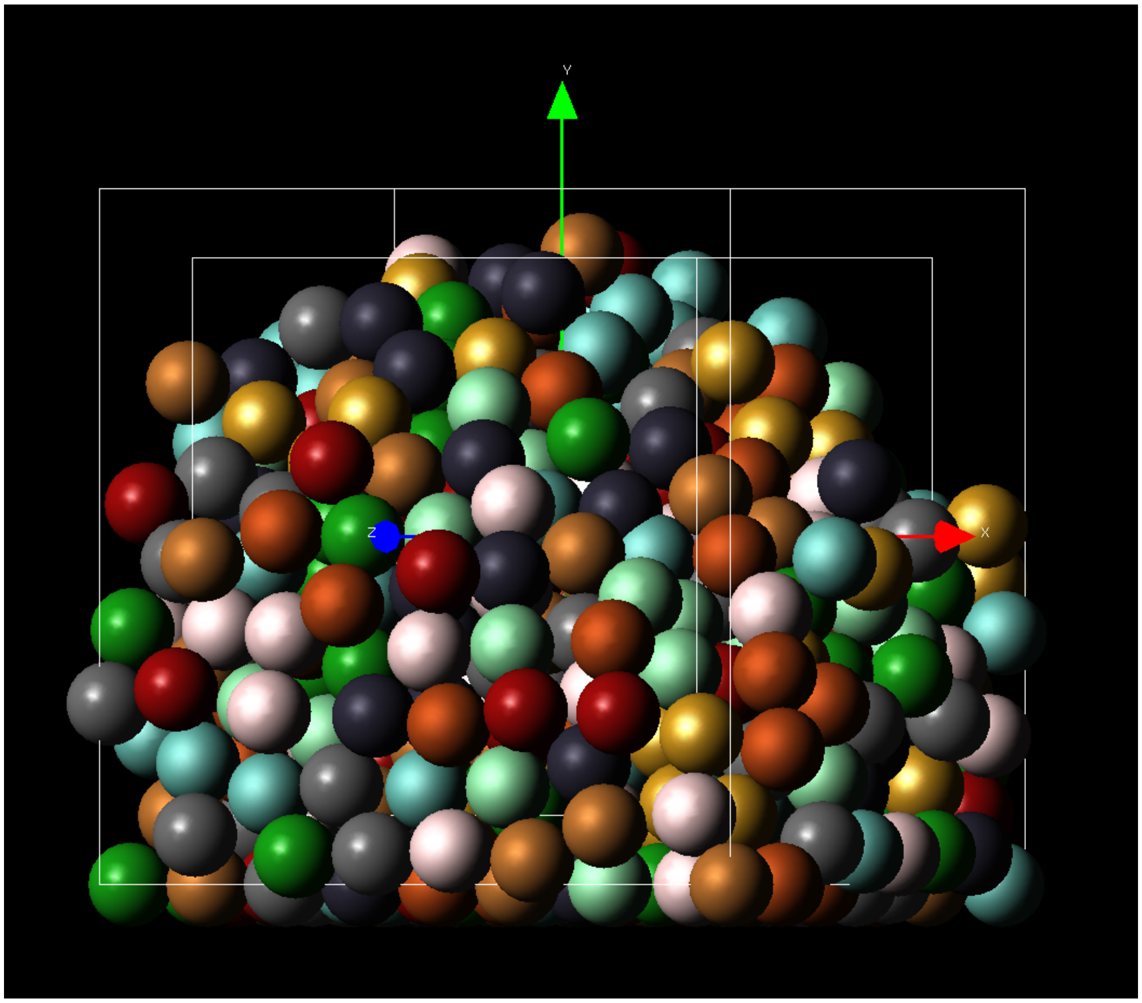


Figure 9.3: Fixed Particle Bed – Large Dense – 500 out of 1002 Particles Fallen

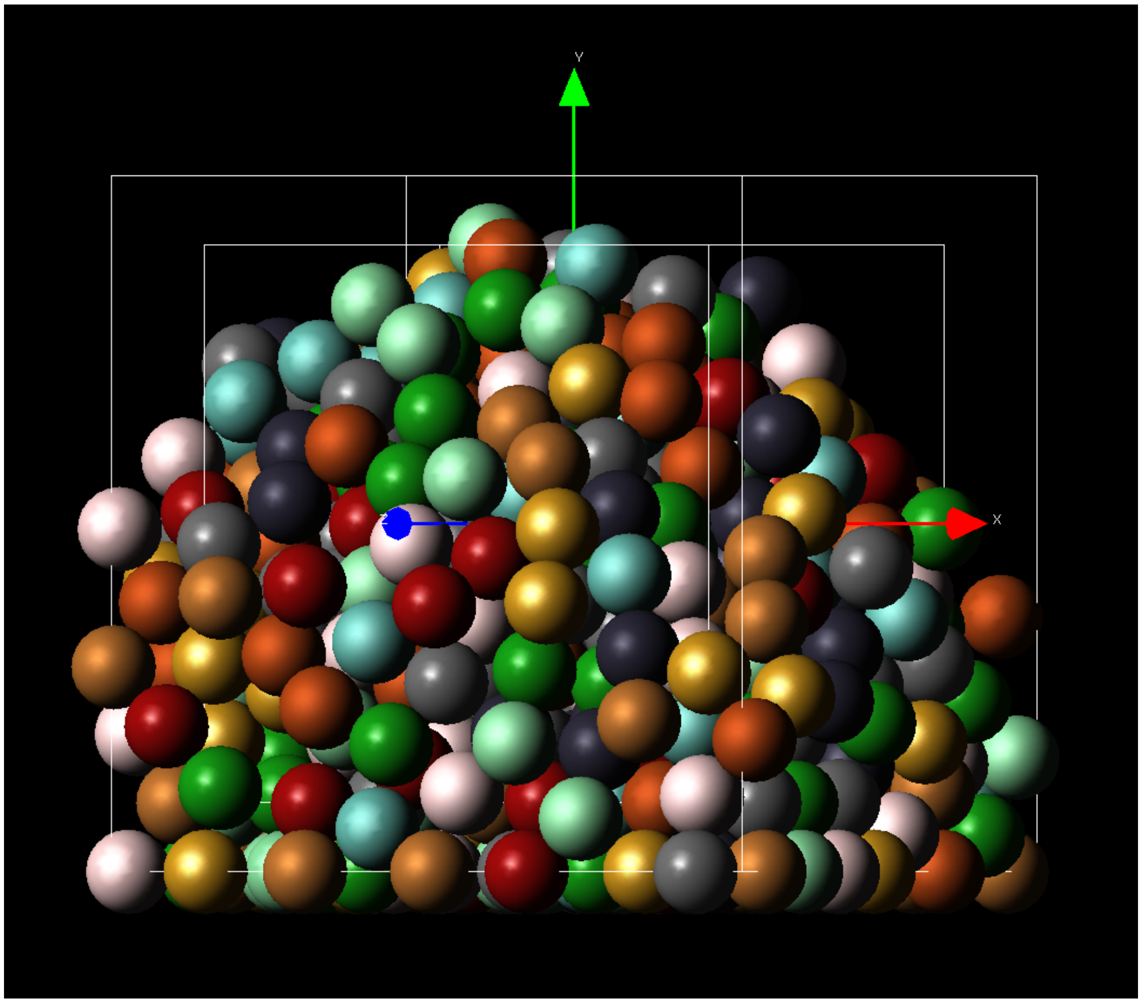


Figure 9.4: Fixed Particle Bed – Small Dense – 500 out of 1002 Particles Fallen

ticle bed in the Small Dense model, Figure 9.4, is slightly higher and steeper than that of the Large Dense model. This, along with higher average falling height, may contribute to the longer packing times of the Small Dense models.

## 9.2 Packing Fraction

Along with the speed of the models, the resulting packing fractions are also a disappointment. While the results are better than those recently obtained by Shi & Zhang (2006), it is difficult to determine exactly how much of an effect the unresolved collision defect had on the final packing fractions. Undoubtedly, the lack of any actual ‘contact’ (average coordination number of zero) of any of the particles only serves to cause an underestimation of the final possible packing fractions. At any rate, the packing fractions, as calculated here, are not sufficient for use in simulation applications.

It is worth noting that each of the models, regardless of starting arrangement appear to begin to coalesce towards a similar packing fraction, as they get to higher numbers of particles. This makes sense intuitively, as the ultimate determining factor of the packing fraction is the method by which they are packed, which all of the models share in common.

## 9.3 Radial Distribution Function

The one bright light of this research is that each model, as shown in their various Radial Distribution Function graphs, produces a sufficiently random, non-periodic pack. In each model, at the 9000 particle level, when the distance is three to four particle diameters away, any periodicity is not evident at all.

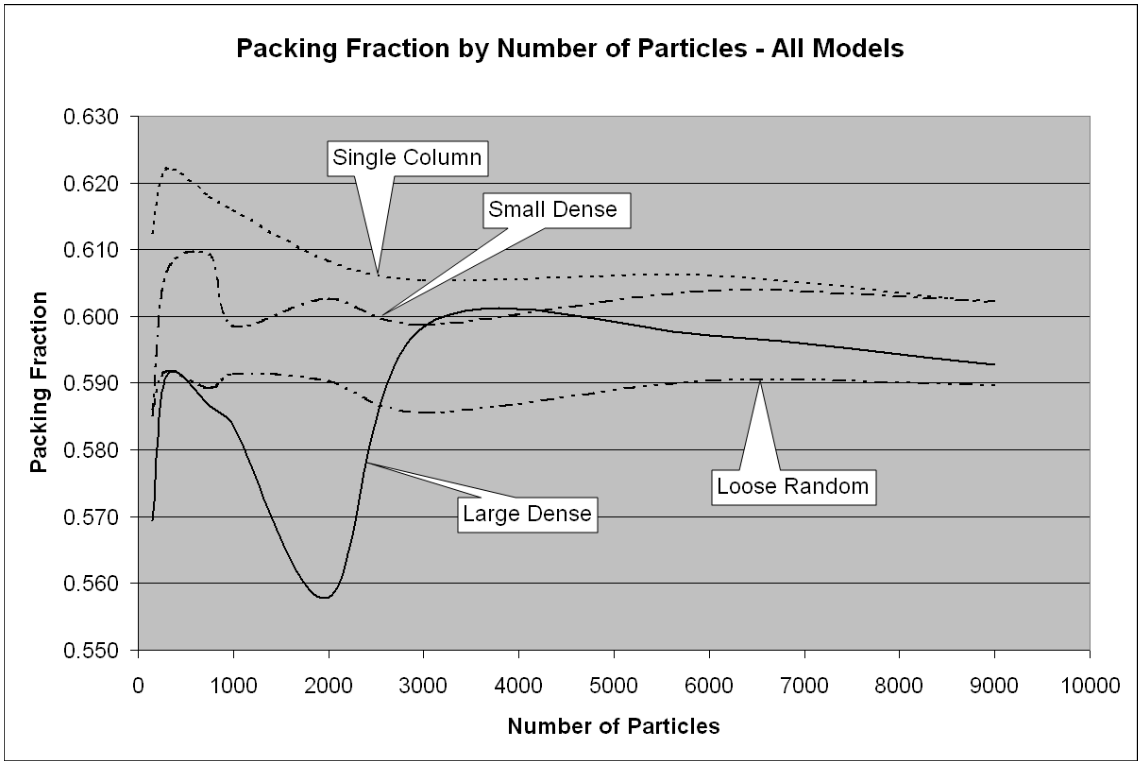


Figure 9.5: All Models – Packing Fraction

# Chapter 10

# Conclusion and Future Work

## 10.1 Conclusion

A few conclusions can be drawn from this research. A negative conclusion is that there's obviously a serious defect in this particular implementation of the SGMP algorithm which allows an unresolved collision to exist in the final pack. If the goal is to model reality, this is the most glaring unreal result of the model.

This implementation of the SGMP is also not nearly fast enough for use in combinatorial chemistry applications. The eventual goal is to model a 100,000 particle pack. None of these starting arrangements (or the actual SGMP model itself) are usable. Even if it were parallelized for multiple processors, it wouldn't be quick enough.

The packing fraction is not high enough, either. Reality (ball bearings in a real volume) creates a packing fraction of over .63. The highest that any of the models here was able to achieve was close to .62, but that was for a special case with a very small number of particles. Looking at the graph with all of the packing fractions from the various runs, the final packing fraction for a suitably large number of particles, will be below .60 if not significantly lower.

While these problems are significant, that does not mean that the algorithm is wholly without merit. The collision defect can be fixed. There are tweaks that can be made and features that can be added to bring the packing fraction up to where it needs to be. The bigger question is if the speed can be brought to a usable value. Typically, increasing the packing fraction requires more calculations and slows down the model. Perhaps with more work, these counterbalancing needs can coexist to make a tool that can be used in combinatorial chemistry applications in the future.

## 10.2 Future Work

The first thing is to fix the collision defect that causes particles to intrude on each other's space. Calculations should be done first in order to determine the magnitude of the problem. It's probably an issue with not building the neighbor list often enough during the SGMP phases. Without fixing this problem, speed and packing fraction improvements are worthless.

Part of the speed problem has to do with the initial falling phase before a particle comes into contact with the fixed particle bed. The particle moves downward at a very slow, fixed rate. There are several possible improvements that could be made. One possibility is to implement an increasing fall distance algorithm. Move the particle downward a fixed distance. If there's no collision, double the distance, and repeat this until a collision happens. Another method could examine the fixed particles in the falling particle's lower neighbor list to see which one is highest. Then the falling particle could be instantly translated to a position slightly above it. Either of these methods could cut a great deal of time from a model run.

Another optimization that could both speed up the model and increase the packing fraction would be a change in the circular sweep. The current implementation of the SGMP algorithm only tests 120 positions on the circular sweep. This was a balance made to attempt to get the packing fraction up without taking too much time. It would be possible to do a multi-stage sweep. The first stage would test only a few points on the circle, maybe five to ten. If a suitable non-collision arc is not found, then a second stage with more tested points could be used, possibly the current 120 points. If a suitable non-collision arc is still not found, then maybe more points could be used, possibly with 720 points on the sweep tested. A careful analysis on multiple data-sets should be made to select the proper number of stages and the number of points in each stage to test. If the right numbers are selected, it should radically speed up the easy SGMP moves, while making the particles fit into tighter spaces than the three degree limit where they are currently cut off.

One more optimization to increase the packing fraction would be to implement a contact model. Currently, the SGMP algorithm leaves all of the particles out of contact with any other particle. After SGMP is done with a particle, the moving particle could be moved slightly to force it to be in contact with its neighbors. This should be a fairly simple (and fast) calculation and would increase the packing fraction, at least somewhat. It would also add an extra dose of realism to the result, as the particles would actually be touching each other.

# Bibliography

Agarwal, P. K., Guibas, L. J., Edelsbrunner, H., Erickson, J., Isard, M., Har-Peled, S., Hershberger, J., Jensen, C., Kavraki, L., Koehl, P., Lin, M., Manocha, D., Metaxas, D., Mirtich, B., Mount, D., Muthukrishnan, S., Pai, D., Sacks, E., Snoeyink, J., Suri, S. & Wolefson, O. (2002), 'Algorithmic issues in modeling motion', *ACM Comput. Surv.* **34**(4), 550–572.

Aste, T. & Weaire, D. (2000), *The Pursuit of Perfect Packing*, Institute of Physics Publishing.

Bagrodia, R., Chandy, K. M. & Liao, W. T. (1991), 'A unifying framework for distributed simulation', *ACM Trans. Model. Comput. Simul.* **1**(4), 348–385.

Balci, O. (2001), 'A methodology for certification of modeling and simulation applications', *ACM Trans. Model. Comput. Simul.* **11**(4), 352–377.

Bennett, C. H. (1972), 'Serially deposited amorphous aggregates of hard spheres', *Journal of Applied Physics* **43**(6), 2727–2734.

Blum, M. (1971), 'On effective procedures for speeding up algorithms', *J. ACM* **18**(2), 290–305.

Clarke, A. S. & Wiley, J. D. (1987), 'Numerical simulation of the dense random packing of a binary mixture of hard spheres: Amorphous metals', *Physical Review B* **35**(14), 7350–7356.

Davis, I. L. & Carter, R. G. (1990), 'Random particle packing by reduced dimension algorithms', *Journal of Applied Physics* **67**(2), 1022–1029.

Ebert, D. S. (1996), 'Advanced modeling techniques for computer graphics', *ACM Comput. Surv.* **28**(1), 153–156.

Furka, A. (1995), ‘History of combinatorial chemistry’, *Drug Development Research* **36**(1), 1–12.

Groult, S. & Bizot, A. (2004), Numerical Simulation of Heterogeneous AP/HTPB Propellant Combustion, *in* ‘40th AIAA/ASME/SAE/ASEE Joint Propulsion Conference’, number 2004-4039.

Gupta, R., Smolka, S. A. & Bhaskar, S. (1994), ‘On randomization in sequential and distributed algorithms’, *ACM Comput. Surv.* **26**(1), 7–86.

Hales, T. C. (2005), ‘A proof of the Kepler conjecture’, *Annals of Mathematics* **162**(3), 1063–1183.

Hartmanis, J. & Hopcroft, J. E. (1971), ‘An overview of the theory of computational complexity’, *J. ACM* **18**(3), 444–475.

Kilgore, G. D. & Scott, D. M. (1969), ‘The density of random close packing spheres’, *Journal of Physics D: Applied Physics* **2**(2), 863–866.

Knott, G. M., Jackson, T. L. & Buckmaster, J. (2001), ‘The random packing of heterogeneous propellants’, *AIAA Journal* **39**(4), 678–686.

Krantz, A. T. (1996), ‘Analysis of an efficient algorithm for the hard-sphere problem’, *ACM Trans. Model. Comput. Simul.* **6**(3), 185–209.

Ling, W. (1947), ‘On the invention and use of gunpowder and firearms in China’, *Isis* **37**(3-4), 160–178.

Lubachevsky, B. D. (1991), ‘How to simulate billiards and similar systems’, *Journal Of Computational Physics* **94**, 255–283.

Lubachevsky, B. D. & Stillinger, F. H. (1990), ‘Properties of random disc packings’, *Journal Of Statistical Physics* **60**(5–6), 561–583.

McGeary, R. K. (1961), ‘Mechanical packing of spherical particles’, *Journal of the American Ceramic Society* **44**(10), 513–522.

Miller, R. B. (1982), Effects of particle size on reduced smoke propellant ballistics, *in* ‘AIAA/SAE/ASME 18th Joint Propulsion Conference and Exhibit’, number AIAA-82-1096.

Motwani, R. & Raghavan, P. (1996), ‘Randomized algorithms’, *ACM Comput. Surv.* **28**(1), 33–37.

Shi, Y. & Zhang, Y. (2006), ‘Simulation of Random Packing of Spherical Particles With Different Size Distributions’, (IMECE2006-15271).

Soderquist, P. & Leeser, M. (1996), ‘Area and performance tradeoffs in floating-point divide and square-root implementations’, *ACM Comput. Surv.* **28**(3), 518–564.

Torquato, S. (2002), *Random heterogeneous materials: Microstructure and macroscopic properties*, 1st edn, Springer-Verlag, New York.

Wang, X., Jackson, T. L. & Massa, L. (2004), ‘Numerical simulation of heterogeneous propellant combustion by a level set method’, *Combustion Theory Modeling* **8**, 227–254.

Weihe, K. (2001), ‘A software engineering perspective on algorithmics’, *ACM Comput. Surv.* **33**(1), 89–134.

Zou, R. P. & Yu, A. B. (1996), ‘Evaluation of the packing characteristics of mono-sized non-spherical particles’, *Powder Technology* **88**, 71–79.