Diffusion Limited Aggregation: Crystal Formation

Vinay Rajur (vsr7)

December 19, 2013

Course: AEP 4380
Instructor: Professor Kirkland

Abstract

This paper explores the properties of crystal formation using a Diffusion Limited Aggregation (DLA) Model. Through simulating various conditions under which crystals and other aggregates may form, several classifications were defined based on relative clustering, apparent symmetries and characteristic visual cues. Limitations of the DLA Model and the classification of crystals are also discussed.

1 Introduction

The presence of pattern formation in nature, the seemingly impossible feat of creating order from random and disordered processes, is an inherently captivating and intriguing phenomenon. For scientists, it is a topic of great interest as well, and creating models to describe these processes has the potential to reveal the much of nature's inner workings.

One model in particular that has received a great deal attention over the past couple decades is the Diffusion-Limited Aggregation (DLA) Model, a deceptively simple stochastic growth model that accurately simulates the growth patterns of objects ranging from snowflakes to entire galaxies. Originally proposed by Witten and Sander in 1981, they developed a simple algorithm to model the growth of clusters in aerosols using diffusion and Brownian motion as the main transport process governing particle behavior [13].

The simplest form of their model starts with an immobile seed particle located at the center of a lattice. At each iteration, a random walker particle is released into the lattice until it reaches a grid spot adjacent to an immobile particle, at which point, the walker becomes immobilized as well. This process is repeated for a set number of particles and the end result is a fractal (self-similar) growth pattern, which is seen in many natural aggregates as well (one

such example is shown in Fig. 1). The complex patterns this model is capable of producing, the accuracy of its results and the simplicity in implementing this algorithm are the key advantages for the DLA Model, and this has been the major influence and drawn significant attention in further researching the additional applications of this type of simulation.

Figure 1: A DLA Cluster of Copper Sulfate grown by electrodeposition. (Image Courtesy: `http://upload.wikimedia.org/wikipedia/commons/b/b8/DLA_Cluster.JPG`)

Over the years, many variations of this simple DLA algorithm have been developed to model a vast array of physical growth processes. For example, varying the lattice geometry has allowed researchers to accurately simulate snowflake growth [8], or even through creating a distribution of seed particles and varying the diffusion properties, one can model the growth of blood vessels in the human retina [6]; clearly there is a wide variety of parameters to expand upon and manipulate with this algorithm. Therefore to simplify, this paper focuses on the development of an accurate 2D model for crystal growth using a continuous surface (i.e. no lattice). Furthermore by incorporating and testing several different sticking probability distributions, I aim to develop generalized classes for different types of aggregate structures based upon fractal dimension, number volume density, apparent symmetries and characteristic visual cues. Additionally, I seek to determine the feasibility of and minimum criteria needed for defining these classes of aggregates.

2 Theory

2.1 Random Walk Behavior in the DLA Model:

As the name suggests, the motion of particles in the DLA Model is assumed to be governed entirely by diffusion through the containing medium. The density

distribution $U(\rho, t)$ in cylindrical coordinates is given by the equation:

$$\frac{\partial U(\rho,t)}{\partial t} = D\nabla^2 U(\rho,t) \tag{1}$$

where D is the diffusion constant, which gives insight into the characteristic length and velocity of particle motion. Solving Eq. 1 for $U(\rho, t)$ yields the result:

$$U(\rho,t) = U_0 \frac{2\pi\rho}{4\pi Dt} e^{\frac{\rho^2}{4Dt}} \tag{2}$$

Thus, the result for the density distribution is a Gaussian curve centered at the origin that spreads out in time (as particles diffuse farther away). Additionally, as Einstein established in a paper written in 1905, the mechanism underlying this diffusion process is the Brownian motion of particles, the seemingly random path taken by particles suspended in some solution [2]. Therefore in the DLA Model, the movement of the unbound aggregate particles can be simulated by a random walk, adhering the "Diffusion-Limited" constraint of the model.

One caveat to consider, however, is that when fixing the seed particle at the center of the coordinate system, we are essentially shifting into the seed particle's frame of motion, and since a general model for DLA should assume that the seed is also undergoing Brownian motion, one would assume that the motion of all other particles must be transformed into the seed's frame of reference as well. Fortunately, since the motion of one Brownian particle is independent of the motion of another, the random walk behavior of the unbound particles is preserved, and so it is still accurate to describe the motion of the unbound particles by a random walk.

2.2 Sticking Probability:

One physical variation that can be applied to the DLA Model is the inclusion of a sticking probability, or the probability that a walker will bind itself to the aggregate after coming into contact with it. A nearly infinite variety of functions can be applied to the walker particles, however, the ones used for this experiment were the following:

1. Constant probability:

$$p = C \tag{3}$$

where $C \in [0, 1]$.

2. Probability dependent on the number of neighboring particles

$$p = A - B\frac{n_1}{n_2} \tag{4}$$

where A and B are set parameters,
n_1 is the number of particles in a $\frac{L}{2} \times \frac{L}{2}$ box around the Brownian particle,
n_2 is the number of particles in a $L \times L$ box around the Brownian particle.

3. Probability dependent on the local curvature at the point of contact [12]

$$p = A(n_L - n_0) + B \tag{5}$$

where A and B are set parameters,
n_L is the normalized number of particles in a $L \times L$ box (e.g. $\frac{N_L}{L^2}$)
$n_0 = \frac{L-1}{2L}$ corresponds to a flat contact surface.

Of these probability functions, the constant probability function shown in 1 is the simplest, and physical interpretation for this is that every particle has some set likelihood of binding to the aggregate at each point of contact (and conversely there is some fixed probability of the particle continuing to diffuse as well).

This type of probability dependence easily demonstrates the implications of including a sticking probability into our DLA Model. For a low sticking probability, random walkers will tend to pass through regions of low particle density and will more likely accumulate in regions of high particle density; therefore in this scenario aggregates will form dense clumps will appear more condensed and "filled in". The opposite is true for a high sticking probability, as the diffusing particles will immediately bind to any part of the aggregate that it comes into contact with. Thus, the end result in this case is a wispier, more spread out aggregate structure (see Figure 2).

This concept can be extrapolated to explain the local behavior of any arbitrary sticking probability function, due to the self-similar nature of aggregates formed by DLA. Regions of low sticking probability will result in denser patterns, while regions with high sticking probability will result in more tenuous growth patterns. This local rule will always apply when considering the results produced by using other probability functions (such as the proposed functions 2 and 3), and the overall impact these functions will have on the total aggregate structure stems from manipulating the locations of these regions of high and low particle density (due to manipulating the sticking probability). Thus, the properties of aggregates formed with a constant sticking probability are similar to the local probabilities of aggregates with more complex sticking probability functions.

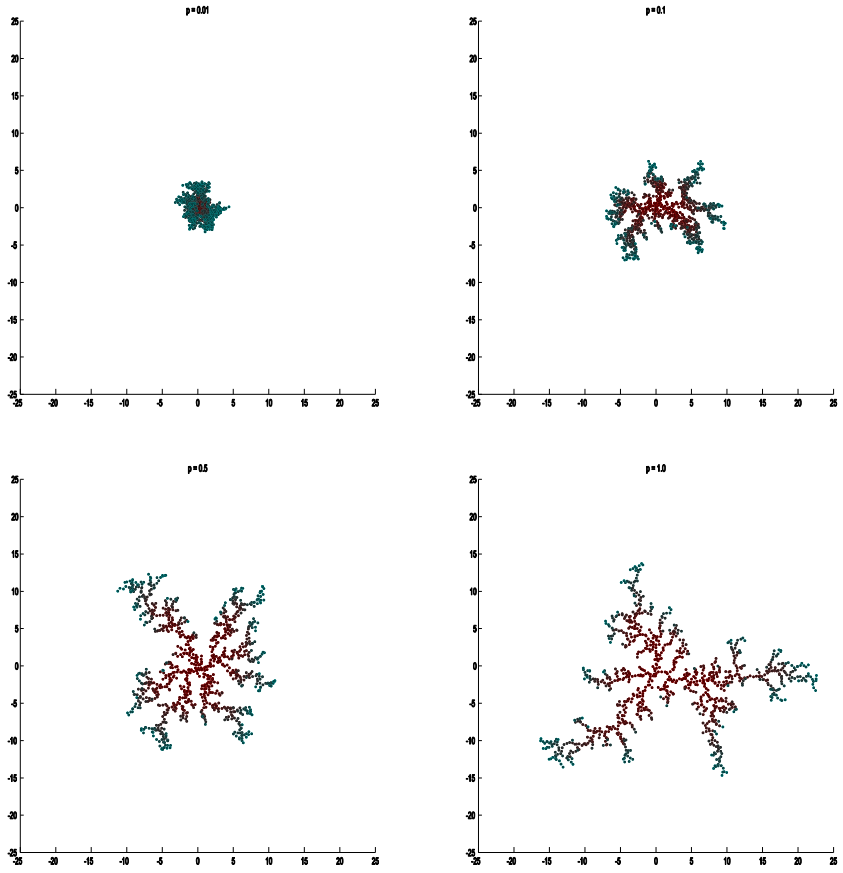


Figure 2: Aggregates grown with various constant sticking probability values (from left to right): $p = \{0.01, 0.1, 0.5, 1.0\}$. As the probability increases, the collections tend to become more tenuous and branched, occupying larger regions of space, and the opposite is true as the sticking probability decreases. Each aggregate contains $N = 1000$ particles.

3 My Algorithm for Simulating 2D Crystal Growth

3.1 Assumptions:

In order to simplify the implementation of the DLA Model described in this section, the following assumptions were made:

1. Simulated particles undergo simple 2D Brownian motion, thus there is no motion or aggregation in the $\hat{z}$ direction[1]. Additionally, particle displacements are of constant step size but with a continuous range in direction (and so particles have an approximately continuous motion).

2. Aggregate member particles have some attractive potential well surrounding them, inside of which walker particles are likely to bind to the aggregate. The probability of binding and the functional dependence of this potential well is described by the sticking probability.

3. In conjunction with 1, particles have no inherent directional preference in binding (circular symmetry of system).

4. Particles are stably bound, thus decrystallization effects (such as thermal activity, momentum transfer during binding, coulomb force interaction, etc.) are negligible.

These assumptions, while not all are entirely physical, are still representative of a true DLA system that one would observe in an experimental setting, simplifying the implementation while preserving the behavior of the model.

3.2 The Algorithm:

In my implementation of DLA Model (presented in Appendix C), I created the simulation on an arbitrary scale with the basic unit length being the particle diameter. Since Particle diameter, step size, binding radius and other parameters are all relatively defined without reference to any physical constants or absolute reference, the following implementation is a general model for Diffusion Limited Aggregation that can be applied to DLA processes on any length scale[2].

For the simulations I ran, I initialized the aggregate with a single seed particle placed at the center of the coordinate system. With each iteration, a random walker particle is initialized at some point along the circumference of a circle that spans the entire aggregate, and the particle undergoes Brownian motion until it binds to the aggregate. For typical simulation with about 1000 particles and constant sticking probability $\rho \in [.25, .75]$, each particle reaches its final state after around 50,000-150,000 steps[3], although due to the nature of the algorithm, these values are subject to significant statistical fluctuations. The entire binding process a single particle undergoes is composed of two main steps:

[1] This is not entirely true in this implementation model, since particles were allowed to overlap. This effect was small, however, except for in areas of very high particle density so the general behavior of the DLA Model was still 2 dimensional

[2] Although the type of model I based this implementation off of was an electrodeposition model

[3] This range of values defines the average number of steps each individual walker particle takes until it binds. Physically, these particles will move in parallel, however, this simulation is for dilute particle concentrations such that the particles can be simulated as arriving one at a time.

1. *Diffusion*:
   Diffusion is the result of Brownian motion, which this model simulates by using Monte Carlo methods to generate a random walk. Each step the particle takes is of a fixed size, however the particle has full directional freedom; thus in this simulation, the particle has a continuous range of motion. After each step, the particle is evaluated as being either bound or not. If the particle is evaluated as unbound, it will continue to diffuse.

2. *Binding*:
   Binding occurs when two conditions are met: a) the particle is within the binding radius of some member of the aggregate and b) the particle is evaluated with some probability as being bound or not (where the probability is generated from the sticking probability function). Only once both conditions are met will the particle stop diffusing and bind to the aggregate. The binding process used in this model is rather simple (perhaps overly simple) where the newly bound particle simply stops in its place. This is slightly unphysical since it allows for overlapping particles, but it can be easily remedied in future updates or implementations by simply adding in a setting method (that "sets" the particle in the closest position that maximizes its sticking probability). The correction by inclusion of a setting method is small, however, and the results of this simulation are generally unaffected as the system scales to larger particle counts.

One feature included in this implementation is the presence of a killing distance. As mentioned previously, particles that diffuse beyond this distance are reinitialized and this serves to prevent wasted computing time and to increase the convergence of the random walks. Additional considerations to improve the processing time include increasing the step size (up to a limit), setting a lower bound on the sticking probability function and the use of a bias parameter (described further in Appendix A).

4 Results

The results of simulations run under various conditions are included in this section. The code presented in Appendix C allowed for the parametrization of many of the aspects of the DLA Model described earlier. Thus, a sampling of aggregates were simulated in an attempt to derive any meaningful means of quantifying and ultimately classifying the aggregate structures observed. Some of the values computed in for these aggregates were fractal dimension, average number density, circularity and additionally the distribution of all aggregate member distances from the seed particle was determined as well. In general, all aggregates demonstrated approximate relative circular symmetry (the average circularity for all observed aggregates was 0.722 ± 0.099), which is to be expected due to the circular symmetry of the simulation (see Section 3.1 Assumptions 1, 3).

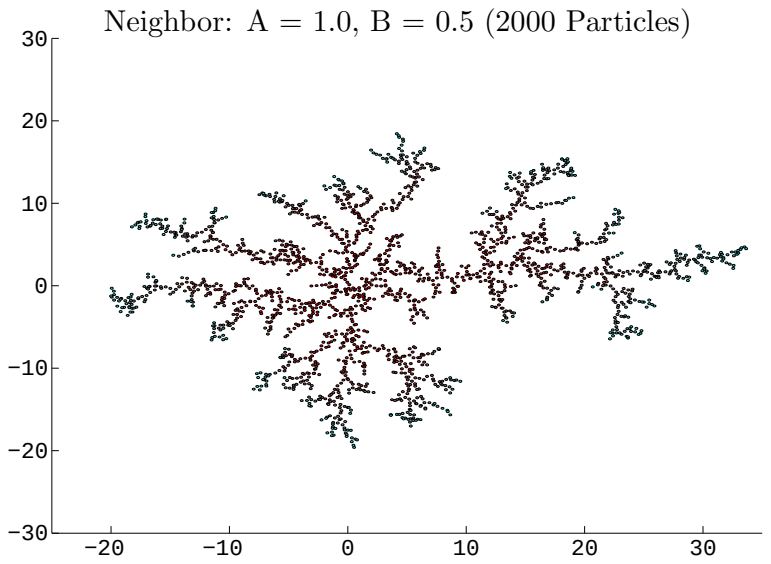


Figure 3: DLA simulation for $N = 2000$ particles using a neighbor dependent sticking probability (Eq. 4) with A = 1.0 and B = 0.5.

The sample of aggregates studied were simulations of 1000 particles with constant sticking probabilities given by p = 0.01, 0.1, 0.25, 0.5. 0.75, 1.0(see Table 1). The aggregates were sampled with a constant sticking probability in order to serve a foundation and provide insight into aggregates formed with more complicated sticking probability distributions. The values obtained from these simulation measurements revealed patterns reflecting general attributes of the aggregates. For example, the average distance from the seed particle and the approximate number density ($\bar{r}$ and ρ) are both measures of compactness, and this was generally observed in aggregates with low $\bar{r}$ and high ρ. Additionally, the measure of circularity, which was found by approximating the aggregate as a 12-sided polygon and comparing its perimeter to its area, showed the relative circular symmetry of the aggregate, and aggregates with circularities under 0.5 tended to have significant branching in one or two particular directions (comparing the relative percentage of the maximum distance to the average distance $\frac{r_{max}}{\bar{r}}$ was also indicative of the magnitude of the largest branch). Lastly the measure of characteristic length, which was the average displacement due to diffusion for some given time interval was used to calculate the fractal dimension according to the formula:

$$D = \frac{\log N}{\log L_{char}} \tag{6}$$

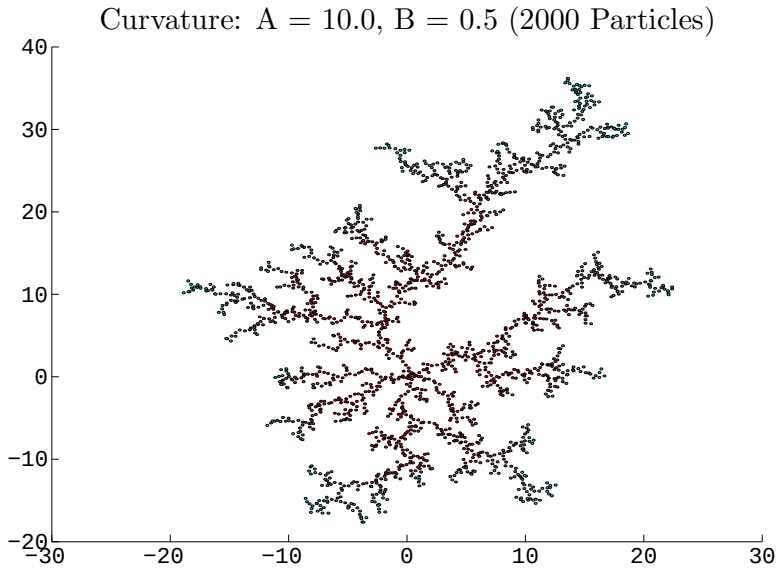


Figure 4: DLA simulation for $N = 2000$ particles using a sticking probability dependent on the local curvature at the point of contact (Eq. 5) with A = 10.0 and B = 0.5.

Fractal dimension is a measure of how "filled in" an object is and because these aggregates lie in only 2 dimensions their fractal dimension is somewhere between 1 and 2. Figure 5 plots the fractal dimension as a function of various sticking probabilities. Interestingly, however, the results across all probability functions (greater than about .2) show that the fractal dimension is approximately constant with a value generally around 1.38[4].

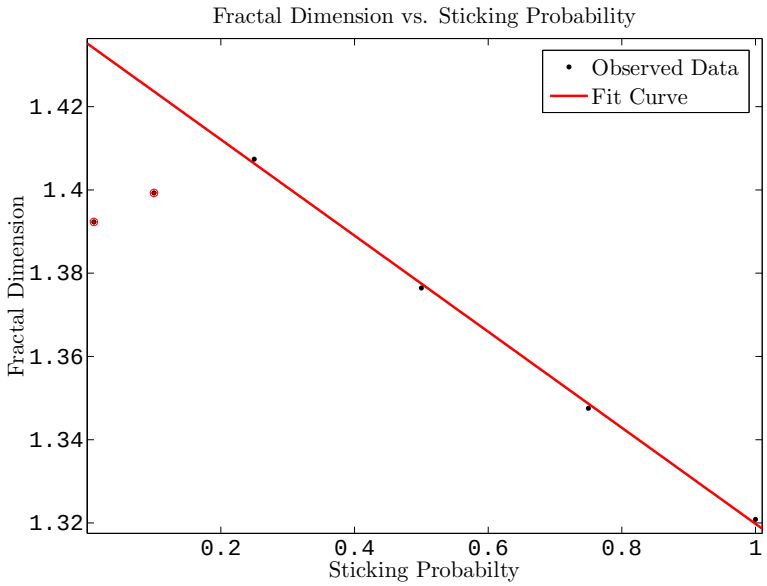


Figure 5: A plot of fractal dimension as a function of sticking probability. Data was gathered from Table 1.

This brings up an important finding about the nature of these measurements: due to the high degree of statistical uncertainty that these aggregates are subject to, these measurements of average distance, density, fractal dimension and circularity are not conclusive on their own. While they serve as a quantitative and useful means of comparing various characteristics of these aggregates, the information they convey is insufficient to try and define classes of aggregates or to even develop any significant form of categorization. This issue manifests itself in all the measurements presently recorded. For example, while it may be possible to say that the observed aggregates formed with a sticking probability of $p = 0.25$ had an average density of 26.04 ± 1.9, circularity of $0.76 \pm .07$, average distance of 6.21 ± 0.21 and fractal dimension of about 1.41 it is impossible to

[4] Note that in Figure 5, the first two points are marked as outliers. This is due to the earlier oversight of allowing particles to overlap, thus making these points unrepresentative of the true relationship between fractal dimension and sticking probability. Note as well, though, that even despite this source of error, the approximate fractal dimension is still relatively constant.

say that any aggregate (or even most of the aggregates) I observe with similar parameters were formed with a sticking probability of 0.25 (or something close), and this is simply because there is too much uncertainty and overlap with other probabilities (due to the random nature of DLA). Thus in order to create any meaningful categorization of these aggregates, other parameters must be sought out as well. Since the ones used in this study were the simplest to calculate, a fully definitive set of categorization may be a complex system to develop.

5 Conclusions:

The results of this DLA simulator are clear, and despite being slightly oversimplified in a few of its assumptions, the aggregates generated are still remarkably similar to the physical aggregates observed in experiments. Additionally, the inclusion of a sticking probability allowed pattern formation in theses aggregates to be studied under various conditions, although it was observed that aggregates formed under higher sticking probabilities were physically more accurate than under low sticking probability conditions.

One of the major secondary objectives of this study was to determine the relevant parameters that characterize an aggregate and furthermore to determine the feasibility of creating classes of aggregate structures. After observing these aggregates being formed under various conditions, only a few relevant parameters were determined, these being fractal dimension, number density, mean radius, and circularity. In conjunction with one another, these values give reasonable insight into the distribution of the aggregate members, and we can begin to see relationships between these values and certain patterns in the aggregate structure. These different parameters create a means of studying and comparing different aggregates. As a method of classifying these aggregates, however, they are insufficient. For example, Figure 5 showed that on average as the sticking probability varied (occasionally resulting in dramatically different aggregate patterns) the fractal dimension remained relatively constant, despite being a relevant characterization parameter. Thus, patterns formed in DLA structures are subject to an overwhelming amount of statistical variation and the aggregates observed in these simulations would blur over any hard definitions formed on these parameters alone.

One potential source of the statistical homogeneity of these aggregates not accounted for in this simulation was the lack of anisotropy during the binding process. In this simulation, this was a result of the continuous and circular symmetry present in the particle motion and binding process. Simulators built on the geometry of a square or hexagonal lattice, for example, have observed various symmetries, which can be attributed restricted nature of the particle behavior in these simulations [12] [3]. Therefore the comparison of the resulting symmetries between aggregates formed on various lattice geometries may be an interesting point of further research, and could potentially be an additional relevant parameter in characterizing an aggregate.

References

[1] Paul Bourke. Constrained diffusion-limited aggregation in 3 dimensions. *Computers & Graphics*, 30(4):646–649, 2006.

[2] Albert Einstein. *Investigations on the Theory of the Brownian Movement*. DoverPublications. com, 1956.

[3] Janko Gravner and David Griffeath. Modeling snow crystal growth ii: A mesoscopic lattice map with plausible dynamics. *Physica D: Nonlinear Phenomena*, 237(3):385–404, 2008.

[4] Thomas C Halsey. Diffusion-limited aggregation: a model for pattern formation. *Physics Today*, 53(11):36–41, 2000.

[5] Michael J Lewis. Analysis of the n-dimensional snowflake. *MIT Undergraduate Journal of Mathematics*, 2009.

[6] Martin A Mainster. The fractal properties of retinal vessels: embryological and clinical implications. *Eye*, 4(1):235–241, 1990.

[7] Takashi Nagatani and H Eugene Stanley. Double-crossover phenomena in laplacian growth: Effects of sticking probability and finite viscosity ratio. *Physical Review A*, 41(6):3263, 1990.

[8] J Nittmann and H Eugene Stanley. Non-deterministic approach to anisotropic growth patterns with continuously tunable morphology: the fractal properties of some real snowflakes. *Journal of Physics A: Mathematical and General*, 20(17):L1185, 1987.

[9] Bogdan Ranguelov, Desislava Goranova, Vesselin Tonchev, and Rositsa Yakimova. Diffusion limited aggregation with modified local rules. *arXiv preprint arXiv:1105.5558*, 2011.

[10] Subir K Sarkar. Saffman-taylor instability and pattern formation in diffusion-limited aggregation. *Physical Review A*, 32:3114–3116, 1985.

[11] James Theiler. Estimating fractal dimension. *JOSA A*, 7(6):1055–1073, 1990.

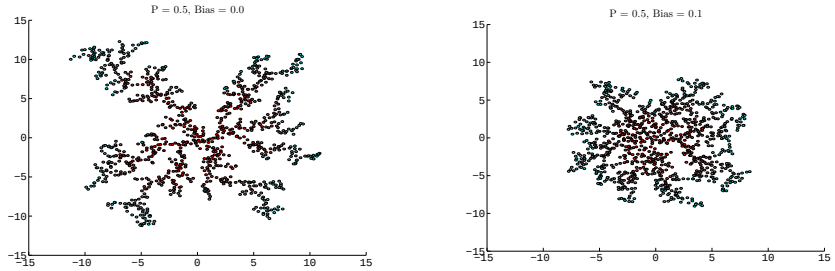
[12] Tamas Vicsek. Pattern formation in diffusion-limited aggregation. *Physical review letters*, 53(24):2281, 1984.

[13] TA Witten Jr and Leonard M Sander. Diffusion-limited aggregation, a kinetic critical phenomenon. *Physical review letters*, 47(19):1400, 1981.

Appendices

A Bias:

The bias parameter was initially included as a means of speeding up the convergence of walker particles towards the aggregate, however, it is observed to have physical implications as well. The bias parameter is controls the magnitude of a constant displacement towards the origin that is added to the net displacement at each step. Thus, the bias simulates the effect of having a constant radial force which attracts the walkers.



(a) Bias = 0.0

(b) Bias = 0.1

Figure 6: Comparison of the effects of the bias parameter. In both cases, aggregates were grown with constant sticking probability $p = 0.5$. Notice how, with a bias applied, aggregate members are more evenly distributed throughout the occupied area and the overall distribution is more compact. Simulated for $N = 1000$ particles.

As Figure 6 shows, the inclusion of even a slight bias has immediately noticeable effects on the particle distribution within the aggregate. The most immediate observation is the lack of branching, a feature that is much more apparent in the Bias = 0 simulation. Further characterizing these two aggregates, other noticeable differences present themselves, such as the the compactness of the aggregate in 6b (given by average distance $\bar{r} = 5.1$ and particle density $\rho = 37.1$ compared to $\bar{r} = 7.3$ and $\rho = 18.6$ for 6a). Additionally, the circularity of the aggregate in 6b is 0.92 whereas the circularity for 6a is 0.70. These, stark differences in values for these aggregates show that the effects of even a slight bias are significant.Even in terms of processing time and convergence speed, the bias parameter significantly increased the convergence speed of the algorithm reducing the total number of iterations from 373,976 (with no bias) down to 43,419 (with slight bias).

The effect of this bias parameter opens the door to further topics to explore in future implementations of this simulation. For example, it would be quite interesting to study DLA in the presence of other field geometries, such aggregation in the presence of a tangential force (in the $\hat{\theta}$ direction as opposed to

force in the currently implemented $\hat{r}$ direction) which could lead to spiral-like patterns.

B Sample Data and Calculations:

The following data presented in Table 1 was collected by simulating aggregates constructed using a range of constant sticking probabilities. Each aggregate was simulated for $N = 1000$ particles and only the average values of each measurement are presented in the table. The values measure are:

Mean Distance ($\bar{r}$):
The average distance of each particle from the origin.

Average Collision or Binding Time ($\bar{t_{coll}}$):
The average time a particle diffuses until it binds to the aggregate.

Maximum Distance (r_{max}):
The farthest distance a particle was from the origin.

Circularity:
A measure of the compactness and symmetry of the aggregate.

Characteristic Length:
The average displacement of a Brownian particle after an interval t_{coll}.

These measurements are relatively straightforward to compute from the aggregate's data, however, characteristic length and circularity will be explained further.

Characteristic length is relatively simple to compute, however, in order to determine the average distance a particle diffuses in t_{coll} a simulation of thousands of Brownian particles was done (see header file r_walk.h in Appendix C).

The calculation of circularity is a bit more involved, requiring one to fit a polygon (arbitrarily chosen to have 12-sides, but accuracy increases with more sides) and to calculate the the perimeter (L) and area (A) of the fit polygon. Then circularity is:

$$4\pi \frac{A}{L^2} \leq 1 \tag{7}$$

A visual representation of this calculation in shown in Figure 7.

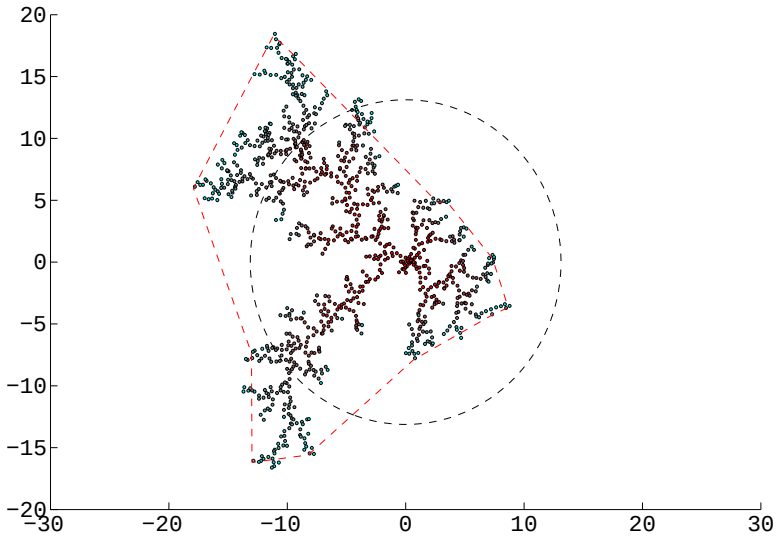


Figure 7: A visual representation of the calculation of circularity. The perimeter and area of the 12-sided polygon used in Eq. 7. This calculation essentially computes the relative similarity between the spanning polygon (red dashed line) and the circle with $r = \bar{r}$ (black dashed line).

C Source Code Listings:

```
1  //-------------------------------------------------------------------
2  /*
3  Final.cpp
4  AEP 4380 Final Project: Diffusion Limited Aggregation
5
6      Simulate the DLA Model and grow aggregates from seed
7      particle(s). Release walker particles individually
8      and model with random walk using Monte Carlo Methods.
9
10     Observe aggregate properties for different sticking
11     probabilities (input from header files) and
12     additionally test the average behavior of Brownian
13     motion for several different time intervals.
14
15
16     Run on Core I2 with g++ 4.8.1 on Windows 7
17
18     Vinay Rajur (vsr7)       18-Dec-2013
19 */
20
21 #include<cstdlib>
22 #include<cstdio>
23 #include<cmath>
24
25 #include<fstream>
26 #include<iostream>
27
28 //Define the following symbol to enable bounds checking:
29 #define ARRAYT_BOUNDS_CHECK
30
31 #include "arrayt.hpp"
32 #include "nr3.h"
33 #include "ran.h"
34
35 using namespace std;
36
37 //-----------------------------------------
38 //PARAMETER INITIALIZATIONS:
39
40 const int N = 1000;                    // Number of Particles in
       Aggregate
41 const float bias = 0.0;                // Central tendency of random
       walkers (increase to speed up simulation)
42
43 Ullong seed = time(NULL);             // Seed for Random Number
       Generator
44 //Ullong seed = 17;                      // Seed for Random Number
       Generator
45
46
47 //OTHER INITIALIZATIONS:
48 double diameter = 1.0;              // Particle Diameter
49 double r_step = .50*diameter;       // Step size
50 double r_bind = .5*diameter;        // Distance to bind particle
51
```

```
52  Ullong counts = 0;                  //Counting steps (as a measure of
        time)
53  arrayt<double> x(N,3);              // Particle Array
54  Ran ran(seed);                      // Random Number Generator
55
56  //OTHER CONSTANTS:
57  const double PI = 4.0*atan(1.0);
58
59  #include "p_coeff.h"
60  //#include "p_neighbors.h"
61  //#include "p_curvature.h"
62
63  #include "r_walk.h"
64  #define RWALKTEST       // run random walk simulation:
65
66  //-----------------------------------------
67  // MAIN LOOP:
68  int main(){
69
70  #ifdef RWALKTEST
71      // Simuate Random Walks
72      rwalktest();
73
74  #else
75
76      clock_t start = clock();
77
78      int i,n0;
79      double tx,ty,r,rmax,r_kill,r_spawn,maxx,maxy,theta;
80      bool bound;
81
82      bool bindcheck(int, double, double);
83
84      ofstream fppos;
85      fppos.open("pos.dat");
86
87      //---------------------------------------
88      // Initializations:
89
90      n0 = 1;                             //Number of initial particles
91
92      Create Distribution of Initial Particles:
93      for(i = 0; i < n0; i++){
94          x(i,0) = 0;
95          x(i,1) = 0;
96      }
97      maxx = 0; maxy = 0;
98      rmax = 2.0*r_step;
99
100     //---------------------------------------
101     // DLA Start:
102
103     for(i = n0; i < N; i++){
104         rmax = sqrt(maxx*maxx + maxy*maxy);
105         r_spawn = rmax + 2.0*diameter;
106         r_kill = r_spawn + 2.0*diameter;
107
```

```
108         // Initialize Particle i:
109 
110         theta = 2*PI*ran.doub();
111         tx = r_spawn*cos(theta);
112         ty = r_spawn*sin(theta);
113         r = sqrt(tx*tx + ty*ty);
114         bound = false;
115 
116         // Simulate Brownian Motion Until Particle is Bound:
117         while(!bound){
118             //Random Walk:
119             theta = 2*PI*ran.doub();
120             tx += r_step*cos(theta) - bias*tx/r;
121             ty += r_step*sin(theta) - bias*ty/r;
122             r = sqrt(tx*tx + ty*ty);
123             counts++;
124 
125             if(r > r_kill){
126                 //Reinitialize Particle
127 
128                 theta = 2*PI*ran.doub();
129                 tx = r_spawn*cos(theta);
130                 ty = r_spawn*sin(theta);
131                 r = sqrt(tx*tx + ty*ty);
132                 bound = false;
133             }
134 
135             bound = bindcheck(i,tx,ty);
136         }
137 
138         if(x(i,0) > maxx){ maxx = x(i,0); }
139         if(x(i,1) > maxy){ maxy = x(i,1); }
140 
141         cout << i << ":\t" << bound << endl;
142     }
143 
144     // Print Array to File
145     for(i = 0; i < N; i++){
146         fppos << x(i,0) << "\t" << x(i,1) << "\t" << x(i,2) << "\t"
                ;
147         fppos << sqrt(x(i,0)*x(i,0) + x(i,1)*x(i,1)) << "\t" <<
                atan2(x(i,1), x(i,0)) + PI << endl;
148     }
149 
150     cout << "Number of Steps Taken: " << counts << endl;
151     fppos.close();
152 
153     clock_t end = clock();
154     double time = (double)(end - start)/CLOCKS_PER_SEC;
155 
156     cout << "Processing Time: " << time << endl;
157 
158 #endif // RWALKTEST
159 
160     cout << "DONE" << endl;
161     return 0;
162 }
```

```
163
164 //-----------------------------------------
165 /*
166 Function bindcheck:
167     Return a random theta value between 0 and 2PI (uniform
        distribution of random variable)
168 */
169 bool bindcheck(int ind, double tx, double ty){
170     bool bound = false;
171     double dx, dy, dr, R;
172     double P(int, double, double);
173
174     for(int i = 0; i < ind; i++){
175         dx = tx - x(i,0);
176         dy = ty - x(i,1);
177         dr = sqrt(dx*dx + dy*dy);        // Distance between
            particles
178
179         if(dr < r_bind){
180             R = ran.doub();
181             if(P(ind, tx, ty) > R){
182                 //Bind the particle
183                 x(ind,0) = tx;
184                 x(ind,1) = ty;
185                 x(ind,2) = counts;
186                 bound = true;
187             }
188         }
189     }
190
191     return bound;
192 }
193 //-----------------------------------------------------------
```

```
1 //-----------------------------------------------------------
2 /*
3 Header File: p_coeff.h
4
5 Function P:
6     Return the probability of the particle sticking to the
        aggregate
7     probability is a constant coefficient between 0 and 1
8 */
9
10 double P(int ind, double tx, double ty){
11     double p;
12
13     p = 0.50;         // Sticking probability
14
15     return p;
16 }
17 //-----------------------------------------------------------
```

```
1 //-----------------------------------------------------------
2 /*
3 Header File: p_curvature.h
```

```
4
5  Function P:
6      Return the probability of the particle sticking to the
           aggregate
7      probability is a function number of neighbors
8  */
9
10
11 double P(int ind, double tx, double ty){
12     int i, n1, n2;
13     double A,B,L, p, C;
14
15     L = 5.0*diameter;            // Sampling cell size
16     n1 = 0;                      // Particle count 1
17     n2 = 0;                      // Particle count 2
18
19     A = 1.0;                     // Probability intercept
20     B = -1.0;                    // Proabability dependence (slope)
21
22     C = 0.01;                    // Min probability
23
24     for(i = 0; i < ind; i++){
25         if( fabs(x(i,0) - tx) < L/2 && fabs(x(i,1)- ty) < L/2){
26             n1++;                 // count particles within L/2 x L
                   /2 cell surrounding ind
27         }
28
29         if( fabs(x(i,0) - tx) < L && fabs(x(i,1)- ty) < L){
30             n2++;                 // count particles within L x L
                   cell surrounding ind
31         }
32     }
33
34
35     p = A + B*n1/n2;             // evaluate probability
36     if(p < C){ p = C; }
37
38     return p;
39 }
40 //-------------------------------------------------------------
```

```
1  //-------------------------------------------------------------
2  /*
3  Header File: p_curvature.h
4
5  Function P:
6      Return the probability of the particle sticking to the
           aggregate
7      probability is a function of the local curvature at the point
           of contact
8  */
9
10 double P(int ind, double tx, double ty){
11     int i,n;
12     double A, B, L, p, C, nl, n0;
13
```

```
14      A = 10.0;                       // Parameter for probability
            function
15      B = 0.50;                       // Parameter for probability
            function
16
17      L = 5.0*diameter;               // Sample Cell size
18      n = 0;                          // Particle Count (inside cell)
19      n0 = (L - 1)/2/L;               // Particle count for straight line
             of particles
20      C = 0.01;                       // Min bound on probability p
21
22      for(i = 0; i < ind; i++){
23          if( fabs(x(i,0) - tx) < L && fabs(x(i,1)- ty) < L){
24              n++;                    // count particles within
                    surrounding L x L cell
25          }
26      }
27
28      nl = n/L/L;
29      p = A*(nl - n0) - B;            // evaluate p
30      //p = ran.doub();
31      if(p < C){ p = C; }             // ensure p > C
32
33      return p;
34  }
35  //-------------------------------------------------------------
```

```
1   //-------------------------------------------------------------
2   /*
3   Header File: r_walk.h
4
5       Function rwalktest:
6           test the properties of the random walk model for
7           the conditions specified in Final.cpp
8
9           Print array of n1 average r^2 for n2 particles after n3
                steps
10  */
11
12  void rwalktest(){
13      int i,s,t;
14      double theta, tx, ty, sum;
15
16      int n_sim = 50;                 // Number of calculations of mean
             r^2 (n1)
17      int n_particles = 100;          // Number of particles simulated
            each simulation (n2)
18      int n_steps = 111973;           // Number of time steps for
            particles to move (n3)
19
20
21      Ullong seed2 = time(NULL);
22
23      Ran myran(seed2);
24
25      ofstream fpwalks;
26      fpwalks.open("walks.dat");
```

```
27
28      for(i = 0; i < n_sim; i++){
29          //Run n_sim simulations
30
31          sum = 0;
32          for(s = 0; s < n_particles; s++){
33              //Simulate n_particles particles:
34              tx = 0; ty = 0;
35
36              for(t = 0; t < n_steps; t++){
37                  // Random Walk for 1 Particle
38                  theta = 2*PI*myran.doub();
39                  tx += r_step*cos(theta);
40                  ty += r_step*sin(theta);
41              } // END STEPS
42
43              sum += tx*tx + ty*ty;
44          } // END PARTICLES
45
46          // Print average r^2 for 1 simulation (to file)
47          fpwalks << sum/n_particles << endl;
48
49          // Print average r^2 for 1 simulation (to output)
50          cout << sum/n_particles << endl;
51      } // END TIMES
52
53      fpwalks.close();
54  }
55  //-------------------------------------------------------------
```

Table 1: Sample DLA Data Collected:

p = 0.01				p = 0.5			
		Mean	St. Dev.			Mean	St. Dev.
$\bar{r}$		1.878887	0.181816	$\bar{r}$		7.562778	0.415017
$\bar{t_{coll}}$		82474.82	13725.67	$\bar{t_{coll}}$		93528.33	36828.52
Density		290.26	53.83976	Density		17.62005	1.865095
rmax		4.745631	0.52132	rmax		16.00884	1.536601
Circularity		0.807827	0.056796	Circularity		0.691915	0.072349
Characteristic Length	142.79			Characteristic Length	151.19		

p = 0.1				p = 0.75			
		Mean	St. Dev.			Mean	St. Dev.
$\bar{r}$		4.614607	0.175134	$\bar{r}$		8.501541	0.403492
$\bar{t_{coll}}$		78262	18361.7	$\bar{t_{coll}}$		111973.3	28866.56
Density		47.1379	3.491457	Density		13.88823	1.234468
rmax		10.62307	0.968741	rmax		17.88367	1.007205
Circularity		0.720795	0.118113	Circularity		0.684356	0.136055
Characteristic Length	139.3			Characteristic Length	168.37		

p = 0.25				p = 1.0			
		Mean	St. Dev.			Mean	St. Dev.
$\bar{r}$		6.206999	0.211136	$\bar{r}$		9.394982	0.833224
$\bar{t_{coll}}$		74976.1	23724.19	$\bar{t_{coll}}$		138017.6	68030.42
Density		26.04078	1.863252	Density		11.54414	1.765577
rmax		13.42167	0.852675	rmax		20.31463	3.087495
Circularity		0.763463	0.068235	Circularity		0.750503	0.083498
Characteristic Length	135.39			Characteristic Length	186.74		