Exploring two and three dimensional DLA Growth through different sticking probabilities

Maria Gragera Garces

ABSTRACT

In this paper, we investigate the influence of sticking probabilities on the growth dynamics of Diffusion-Limited Aggregation (DLA) in both two and three-dimensional systems. Our study focuses on comparing the growth patterns, particularly the evolution of radius and fractal dimension, under varying probabilities of particle sticking. Through extensive simulations, we analyse how the introduction of a gravitational field affects the aggregation process and the resulting morphology of DLA structures. We observe average fractal dimensions of 2.019 ± 0.070 in 2D systems, and 2.373 ± 0.641 in 3D systems, across the full sticking probability range. As well as squared root growth relationship between the size of the system, and it cluster radius in 2D and 3D. Many of our results are consistent with existing DLA literature, and elucidate on further research avenues that could be explored in this field.

1 Introduction

Diffusion Limited Aggregation, DLA, is a well established model which imitates the formation of clusters by particles diffusing through a medium. It was initially established by Witten and Sander[1], as a generator of tree-like particle clusters with obvious similarity centred on a fixed point. The growth of such clusters can be measured by their fractal dimension, which is established from the number of particles in the cluster and their density, as well as from their total radius in the space.

DLA is often associated with fractal growth, and has been studied in a myriad of contexts: from urban growth[2] to the formation of snowflakes[3]. In this paper, we will discuss how DLA cluster growth differs between 2D and 3D environments. Furthermore, we examine the effects of introducing a probability of sticking in both models and analyse its impact on DLA growth patterns.

2 Method

2.1 DLA Methodology Overview

In DLA a cluster of particles grows around an initial particle. Through this study we evaluated a series of clusters in 2D and 3D spaces, with grids of size $1600 * 1600$ and $400 * 400 * 400$ particles respectively. Each particle was modelled as a point in the grid. Starting from initial particle in the zero position, 3 circles were set with growing radius: the cluster circle, a generation circle in which particles appear, and a killing circle, beyond which following the particles Brownian motion would be inefficient[4]. Figure 1 illustrates this system in three dimensions. The cluster circle englobes all particles that have stuck to the circle and has a radius equivalent to the distance between the furthest particle from the centre.

A travelling particle appears within the generation circle and begins a random walk through the grid until it sticks to another particle in the cluster or leaves the kill circle boundary. The probability of sticking to the cluster can be tuned through the system as a probability or a relation to the gravitational force.

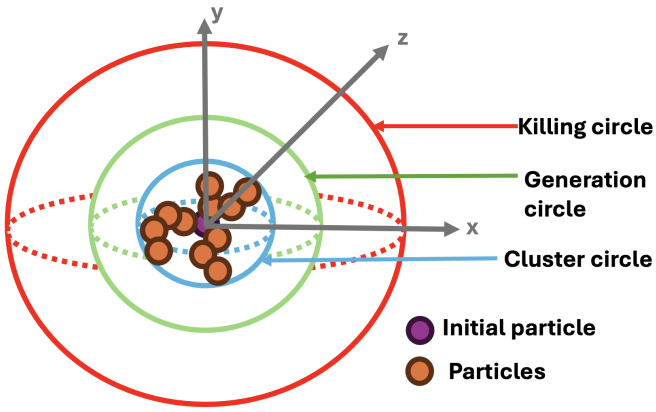


Figure 1. The figure presents a diagram illustrating the circles employed in the DLA code. The initial particle is represented in purple and serves as the central point for all circles depicted. Within the diagram, three distinct circles are highlighted: the "cluster circle" approximating the cluster size, the "starting circle" marking where particles are introduced into the system, and the "killing circle" indicating points where particles are removed from the system to prevent significant deviation from the cluster.

Each cluster is established based on a randomly generated seed, thus mimicking the formation of genuine fractal growth. All circles expand alongside the cluster, with the generation circle's radius being 1.2 times larger than that of the cluster circle, and the killing circle's radius being 1.7 times larger than the cluster circle.

2.2 Sticking Probability Algorithm

The sticking probability algorithm incorporates a stochastic element into the growth of the cluster. Upon contact with existing particles, each new particle has a chance of adhering

to the cluster, determined by a specified sticking probability, p. If the random ratio generated from 1 to an upper limit m is less than p, the particle becomes part of the cluster. Otherwise, it proceeds independently, following the original trajectory rules. This adjustment introduces variability into the cluster's expansion, as particles may or may not attach depending on the probabilistic outcome.Those that do not stick continue their movement without contributing to cluster growth, but still interact with existing particles according to system rules.

2.3 Error analysis

To calculate error bars through this work, we used the standard deviation method. Each experiment was conducted at minimum of 5 times for every probability setting. This ensured that the standard deviation provides a reliable estimate of variability. However, it's worth noting that the standard deviation does not capture all aspects of data dispersion, particularly in cases of non-normal distributions or outliers.

3 Results

Our work can be divided in two comparable sections: the initial foundational model, and a probabilistic model that introduces a sticking probability into the initial system. Our results will be presented for these two sections separately, and will be explored and compared in the discussion.

3.1 Foundational model

3.1.1 Radius

As per Figure 2 we can see a squared root like relationship between the number of particles and radius of the cluster. This is an expected behaviour, as the cluster size and thus radius should grow with the number of particles. DLA process lead to fractal-like structures where the cluster's perimeter grows faster than its radius. Mathematically, this results in a growth rate that is not linear but rather exhibits a power-law behaviour, as obtained. The average cluster radius for clusters with particles in the range from 2 to 1000 in 2D is found at 33.776 ± 0.423.

In three dimensional systems this behaviour behaves less nicely. As per Figure 3, we observe a correlation between higher cluster radius and larger particle systems. This relation is expected, but the nature of this growth is less well behaved. The average cluster radius for clusters with particles in the range from 2 to 1000 in 3D is found at 1.516 ± 11.16.

3.1.2 Fractal Dimension

On the other hand, dimension shows a different relation to the number of particles in a system. In a two dimensional system, as shown in Figure 4 there is an exponential decay that stabilizes past 50 particle systems. At this point the rate of particle aggregation balances out with the rate of particle diffusion. Once this equilibrium is achieved, the system stabilizes, resulting in the observed exponential decay in the growth of particle systems beyond a certain size threshold. This stabilization is a characteristic feature of diffusion-limited aggregates. We

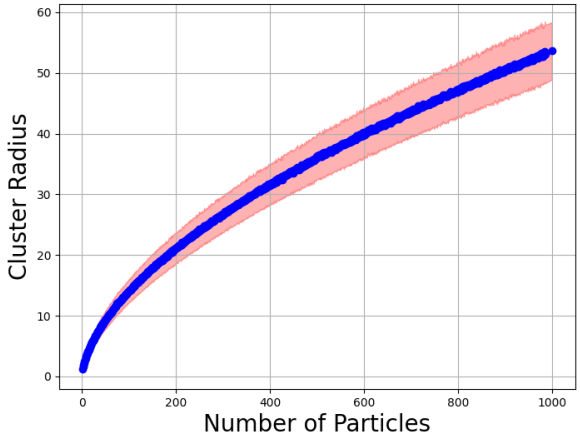


Figure 2. The figure illustrates the relationship between cluster radius and the number of particles in 2D DLA clusters, for clusters ranged from 2 to 1000 particles. A square root-like growth pattern is observed, indicating an increase in cluster radius with the number of particles. The standard deviation error is illustrated in light red.

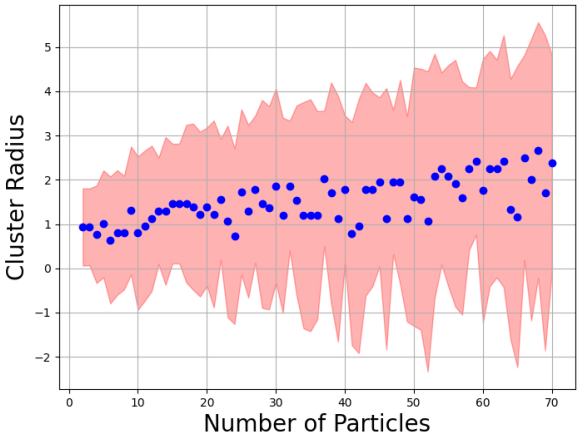


Figure 3. The figure illustrates the relationship between cluster radius and the number of particles in 3D DLA clusters, for clusters ranged from 2 to 1000 particles. We observe a faint correlation between growth in cluster radius and number of particles. The standard deviation error is illustrated in light red.

obtain an average fractal dimensions of 1.736 ± 0.001 in 2D systems.

In three dimensional systems, as per Figure 5, the relationship observed in two dimensions is once again lost. We observe a physically appropriate correlation between fractal dimension growth and cluster particle size, with an average fractal dimension of 2.373 ± 2.449. For the equivalent range of cluster particle numbers, we obtain an average fractal dimension of 2.025 ± 0.005 in our 2D simulations.

3.2 Probabilistic

3.2.1 Radius

As per Figure 6, in two-dimensional systems, we observe quite consistent cluster radius' through the different sticking probabilities. To the exception of one outlier at $p = 0.95$, which shall be investigated further, we observe consistency even through the lower probability ranges $0.05 < p < 0.10$. Through this smaller probability range, we do observe a small progressive cluster radius growth across the probabilities.

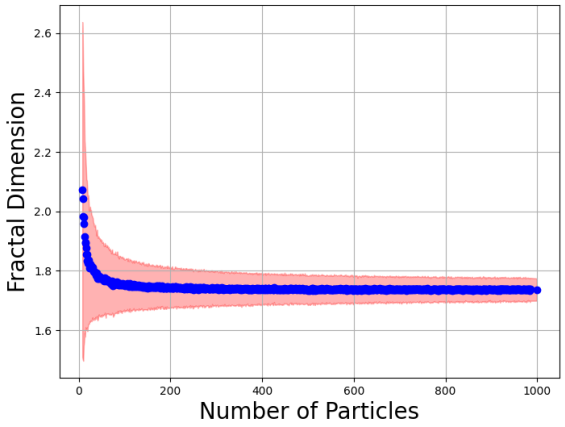
The obtained average cluster radius across this range with clusters sized between 0 and 70 particles is 1.695 ± 0.641, this value is very high compared to our foundational model result. This value is similar, and has a much lower error rate than the average cluster radius in clusters ranged from 0 to 70 particles across all sticking probabilities: 1.516 ± 11.16.

Similarly to the $p = 1$ case discussed in 3.1.1, 3D systems showcase less well-behaved relationships, with very high error rates, but which still respect the physicality of the system, in other words whose radius is still reasonable. You can find the relevant graphs and discussion in Appendix A.2.

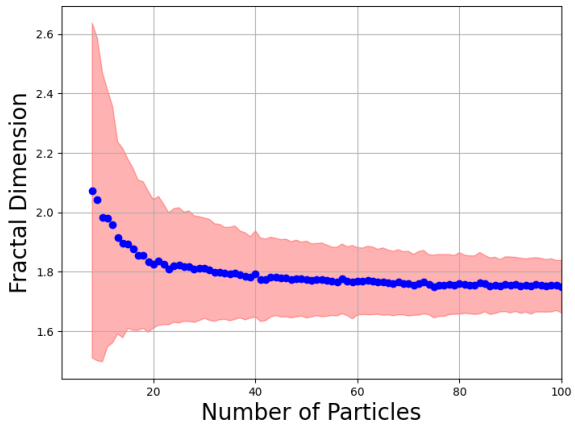
3.2.2 Fractal Dimension

In two-dimensional systems, we observe the opposite effect when working with fractal dimensions. As the probability of sticking grows, the fractal dimension of clusters decreases. This can be observed in Figure 7. The obtained average fractal dimension across all probabilities is 1.843 ± 0.044, this value is very similar to our foundational model result. Furthermore, at low probabilities, $0.05 < p < 0.10$, the average fractal dimension averages at 2.373 ± 2.449, which is above the expected maximum fractal dimension of 2. This is consistent with the wider probability range average fractal dimension of clusters sized between 0 and 70 particles, which lies at 2.373 ± 0.641, as well as the $p = 1$ case which has an average fractal dimension of clusters sized between 0 and 70 particles at 2.437 ± 0.057.

Similarly to the $p = 1$ case discussed in 3.1.2, 3D systems showcase less well-behaved relationships, with very high error rates, but which still respect the physicality of the system, in other words whose fractal dimension is inferior or equal to 3. You can find the relevant graphs and discussion in Appendix A.2.



[a]



[b]

Figure 4. **a)** The figure illustrates the relationship between fractal dimension and the number of particles in 2D DLA clusters, for clusters ranged from 2 to 1000 particles. A exponential like decay pattern is observed, indicating a fast decrease following a stabilization of the fractal dimension across the growing cluster sizes around the 40 particle mark. The standard deviation error is illustrated in light red. **b)** The figure illustrates the relationship between fractal dimension and the number of particles in 2D DLA clusters, for clusters ranged from 2 to 100 particles. Thus better illustrating the initial decay of fractal dimension for smaller clusters. The standard deviation error is illustrated in light red.

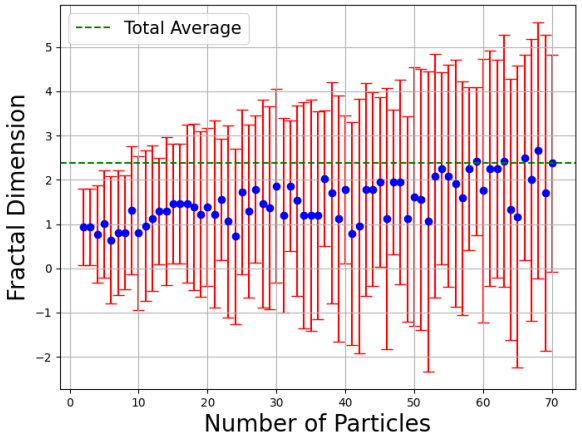


Figure 5. The figure illustrates a plot of fractal dimension against the number of particles, ranging from 2 to 70, for clusters in 3D. The total average, depicted by a dotted line, is 2.373 ± 2.449, with standard deviation error bars shown in red. A noticeable correlation between the growth in the number of particles and fractal dimensions is observed from the plot.

4 Discussion

The results obtained from the foundational model provide valuable insights into the behaviour of cluster properties in both two and three-dimensional systems. Early computer experiments, showed that the fractal dimension 1.71 for clusters in 2D and 2.5 for clusters in 3D[5], which aligns with our findings. Nonetheless, our findings also showcase some interesting deviations from expected results, which we will now proceed to discuss.

As depicted in Figure 2, the relationship between the number of particles and the cluster radius follows a square root-like pattern in 2D systems, indicating an expected growth in cluster size with the number of particles. Similarly, the exponential decay observed in fractal dimension in Figure 4 aligns with the expected stabilization of fractal structures in diffusion-limited aggregates. However, an interesting observation arises when considering the influence of sticking probability on fractal dimension. An increase in sticking probability leads to a decrease in the fractal dimension of clusters, as shown in Figure 7. Further investigation is warranted to better understand the underlying mechanisms driving this phenomenon.

Conversely, in 3D systems, the behaviour of cluster properties appears to deviate from 2D patterns. While a correlation between higher cluster radius and larger particle systems is observed, as depicted in Figure 3, the growth pattern exhibits unexpected variability and high error rates, indicating com-

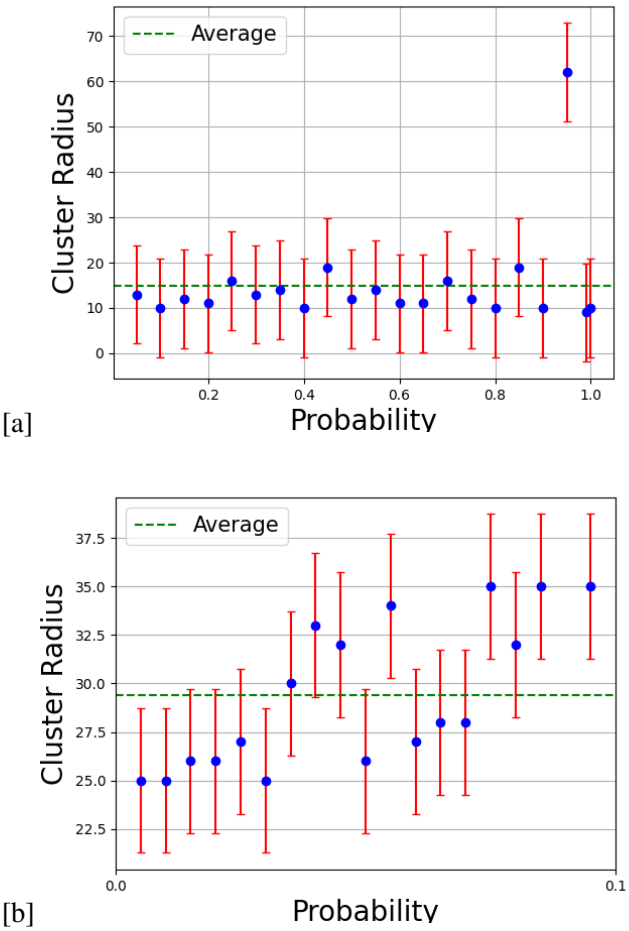
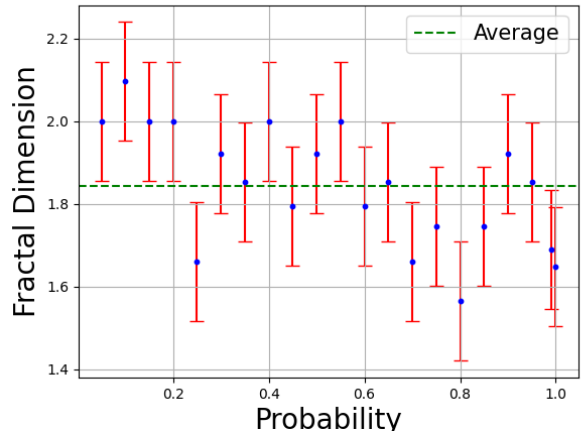
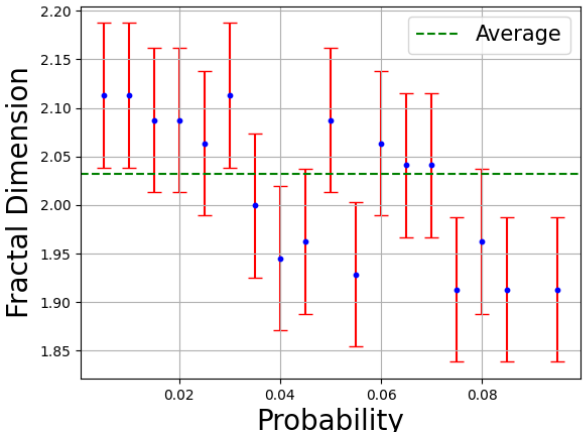


Figure 6. **a)** The figure presents a comparison between cluster radius and sticking probability for 2D clusters ranging from 2 to 1000 particle numbers. Across most probabilities, a consistent set of values is observed, except for an outlier at $p = 0.95$. The standard deviation error bars are depicted in red. The average, illustrated by a dotted line, is 14.952 ± 10.887. **b)** The figure presents a comparison between cluster radius and sticking probability for 2D clusters ranging from 2 to 1000 particle numbers, and probabilities between 0.05 and 0.1. We observe a growth of cluster radius with probability. The standard deviation error bars are depicted in red. The average, illustrated by a dotted line, is 29.388 ± 3.728.



[a]



[b]

Figure 7. **a)** The figure presents a comparison between fractal dimension and sticking probability for 2D clusters ranging from 2 to 1000 particle numbers. We observe a decrease in fractal dimension across growing probabilities. The standard deviation error bars are depicted in red. The average, illustrated by a dotted line, is 2.373 ± 2.449. **b)** The figure presents a comparison between fractal dimension and sticking probability for 2D clusters ranging from 2 to 1000 particle numbers, and probabilities between 0.05 and 0.1. The behaviour of this relation is consistent with the wider behaviour of fractal dimension and probability, we observe a decrease of fractal dimension as we reach higher probabilities. The average, illustrated by a dotted line, is 2.373 ± 0.641.

plexities in three-dimensional cluster dynamics. With no variability is sticking probability, 3D systems showcase a growth in fractal dimension as the number of particles in the cluster grows, contrary to the decrease observed in 2D clusters.

It is important to note that the range of number of particle sizes studied in 3D clusters is significantly lower than the range studied for 2D cluster. In 2D systems, we observed a stabilization of fractal dimension past the 40 particle cluster size. We would expect this stabilization to appear later for 3D clusters due to the increase in available growth space. Therefore, our variability and error rates may be due to the reduced scope of our 3D data.

5 Conclusion

In summary, our study confirms established findings regarding cluster properties in two and three-dimensional systems. However, deviations from expected behaviour were observed, particularly regarding the influence of sticking probability on fractal dimension and the complexities in three-dimensional cluster dynamics. These deviations highlight the need for further investigation to understand the underlying mechanisms. Additionally, our comparison between 2D and 3D systems underscores the unique characteristics of cluster dynamics in different dimensions. It's important to note the limitations of our 3D data due to the narrower range of particle sizes studied. Addressing these limitations and exploring the factors contributing to observed deviations will contribute to refining our understanding of cluster formation processes.

References

1. T. A. Witten, J. & Sander, L. M. Diffusion-limited aggregation, a kinetic critical phenomenon. *Phys. Rev. Lett.* **47**, 1400 (1981).

2. Batty, M. & Longley, P. Urban growth and form: scaling, fractal geometry, and diffusion-limited aggregation. *Environ. Plan. A* **21**, 1447–1472 (1989).

3. Packard, N. H. Lattice models for solidification and aggregation. (1985).

4. Feynman, R. The brownian movement. *The Feynman Lect. Phys.* **I** (1964).

5. Benoit B. Mandelbrot, M. F. Fractals. *Encycl. Phys. Sci. Technol.* (2003).

A Appendices

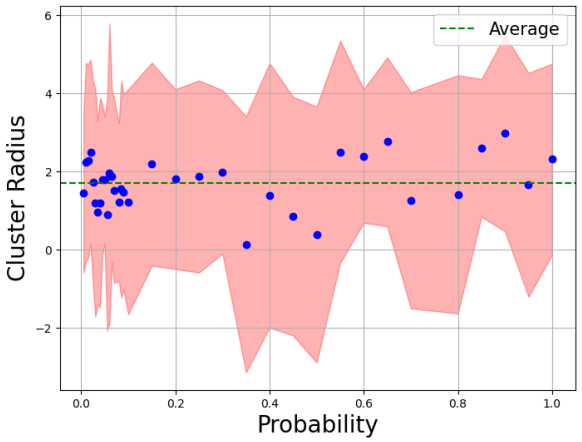
A.1 Computational Specifications

The code presented herein has been successfully executed and compiled on a computing platform equipped with the following specifications:
Processor: 2 GHz Quad-Core Intel Core i5 Graphics: Intel Iris Plus Graphics with 1536 MB capacity

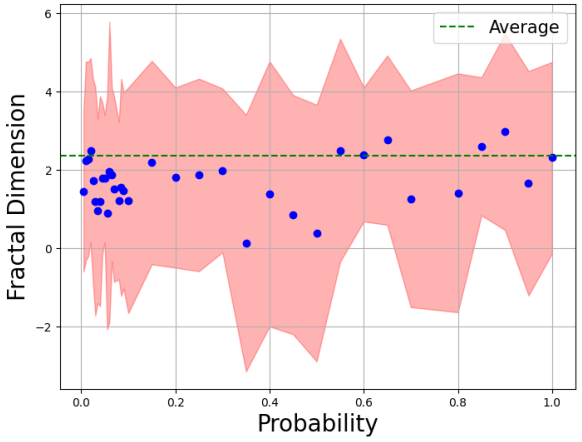
Memory: 16 GB of 3733 MHz LPDDR4X RAM
Operating System: macOS version 14.2.1 (23C71)

A.2 3D Probability dependent graphs

The figure bellow illustrates the relationship between cluster radius and sticking probability in 3D DLA clusters, for clusters ranged from 2 to 70 particles. The standard deviation error is illustrated in light red.



The figure bellow illustrates the relationship between fractal dimension and sticking probability in 3D DLA clusters, for clusters ranged from 2 to 70 particles. The standard deviation error is illustrated in light red.

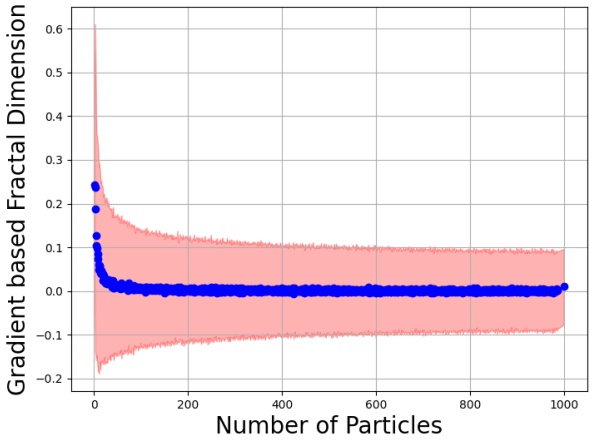


A.3 Fractal dimension via ln relationship

In the study above, we have utilized the approximation of fractal dimension defined by

$$d_f = \frac{N_c}{R/a} \tag{1}$$

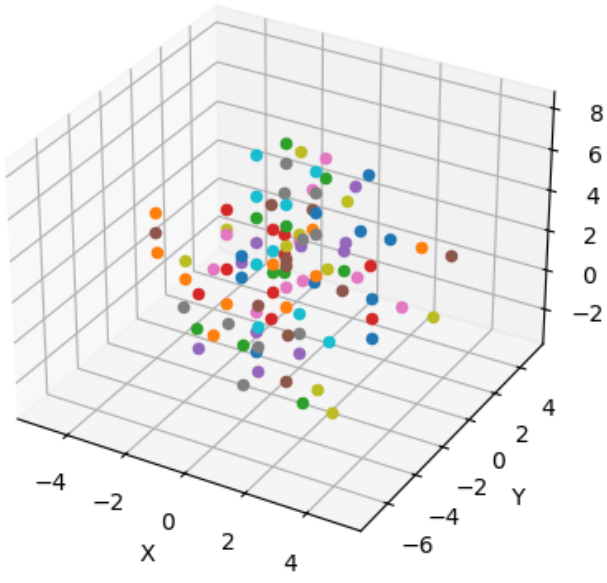
with $a = 1$, and N_c number of particles, R Cluster radius. Nonetheless, it is also possible to estimate d_f by plotting $ln(R)$ against $ln(N_c)$, and measuring the gradient. To confirm this we plotted $ln(R)$ against $ln(N_c)$ and then compared the fractal dimension obtained through this method to our previously obtained results.



As you can observe, the resulting graph is identical in terms of values. The only difference one can observe is associated with the error bars. Given both of the ways of measuring this value are approximations, a slight difference in error values isn't uncommon and can be attributed to various factors. For the sake of aesthetics, we decided to utilize the method with the lower error rates.

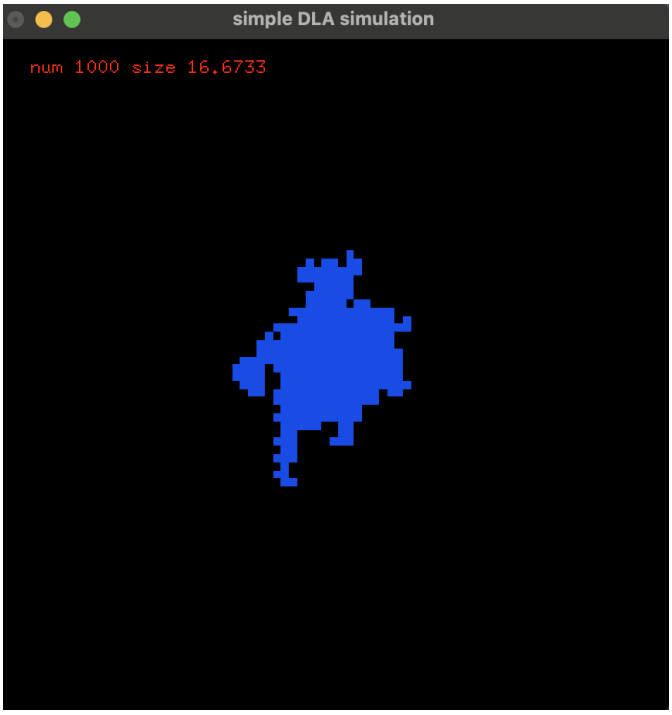
A.4 3D Visualisation DLA cluster

This graph showcases one of our 3D DLA clusters. Each particle position is represented by a coloured dot.



A.5 Little 2D DLA monster

In one of our runs, we got this little DLA monster, and thought it might brighten your day.



A.6 Theoretical gravitational field implementation

An interesting addition to this work would comprise the addition of a gravitational field into the simulation. We created an initial layout for such an implementation which would implement a gravitational force, which directly affected the probability of a particle sticking to the cluster once it comes in contact with it. This force is set through equation:

$$F = G\frac{M-m}{|R-r|^2} \tag{2}$$

and considers the mass of each particle to be equivalent, thus comparing the travelling particle mass to the cluster total mass as well as the distance between the travelling particle and the centre of the cluster. This code can be found in the Appendix "Gravitational Force implementation code" section. Unfortunately, due to computational costs and time, we were not able to test this work. We would highly recommend this test as a next step in our research.

We expect to see significant differences in the growth behaviour between gravitational and non-gravitational DLA systems. By elucidating these dynamics, a further study could contribute to a deeper understanding of the fundamental principles governing DLA growth and its potential applications in diverse fields such as materials science, physics, and biology. Furthermore, exploring the relationship between gravitational strength and the observed changes in DLA characteristics, would provide insights into the complex interplay between diffusion, aggregation, and external influences.

The full code for such an implementation has been uploaded to the code section, as well as partially appended below.

A.7 Gravitational Force implementation code

```
//
//  DLASystem.cpp
//

#include "DLASystem.h"
#include <fstream> //work with csv
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <random>

int DLASystem::Update() {

cout << "Update get's called" << endl;

bool end = 0;

if (lastParticleIsActive == 1)
moveLastParticle();
else if (numParticles < endNum) {
addParticleOnAddCircle();
setParticleActive();
}
else if (numParticles == endNum)
{
end = 1;
}
if (lastParticleIsActive == 0 ||
 slowNotFast == 1)
glutPostRedisplay();

return end;
}

void DLASystem::clearParticles() {

cout << "clearParticles get's called"
 << endl;
for (int i = 0; i < numParticles; i++) {
delete particleList[i];
}
particleList.clear();
numParticles = 0;
}

void DLASystem::Reset() {

cout << "Reset get's called" << endl;
// stop running
running = 0;

clearParticles();
```

```
lastParticleIsActive = 0;

// set the grid to zero
for (int i = 0; i < gridSize; i++) {
for (int j = 0; j < gridSize; j++) {
for (int k = 0; k < gridSize; k++)
grid[i][j][k] = 0;
}
}

// initial condition & parameters
addCircle = 10;
killCircle = 2.0*addCircle;
clusterRadius = 0.0;
// add a single particle at the origin
double pos[] = { 0.0, 0.0, 0.0 };
addParticle(pos);

// set the view
int InitialViewSize = 40;
setViewSize(InitialViewSize);

}

void DLASystem::setGrid(double pos[],
int val) {
cout << "setGrid get's called"
 << endl;
int halfGrid = gridSize / 2;
grid[(int)(pos[0] + halfGrid)]
 [(int)(pos[1] + halfGrid)]
 [(int)(pos[2] + halfGrid)]
 = val;
}

// read the grid cell for position
int DLASystem::readGrid(double pos[]) {
cout << "readGrid get's called"
 << endl;
int halfGrid = gridSize / 2;
return grid[(int)(pos[0] + halfGrid)]
 [(int)(pos[1] + halfGrid)]
 [(int)(pos[2] + halfGrid)];
}

// check if the cluster is big enough
int DLASystem::checkStop() {
cout << "checkStop get's called"
 << endl;
if (killCircle + 2 >= gridSize / 2) {
pauseRunning();
cout << "STOP" << endl;
glutPostRedisplay(); // update display
return 1;
}
else return 0;
}

// add a particle to the system
void DLASystem::addParticle(double
pos[]) {
cout << "addParticle get's called"
 << endl;
Particle * p = new Particle(pos);
particleList.push_back(p);
numParticles++;
setGrid(pos, 1);
}

void DLASystem::addParticleOnAddCircle()
{
cout << "addParticleOnAddCircle get's
 called" << endl;
double pos[3];
double theta = random_max(0,1) * 2 * M_PI;
double phi = random_max(0,1) * M_PI;
pos[0] = ceil(addCircle * cos(theta));
pos[1] = ceil(addCircle * sin(theta));
pos[2] = ceil(addCircle * sin(phi));
if (readGrid(pos) == 0)
addParticle(pos);
else
cout << "FAIL " << pos[0] << " "
  << pos[1] << endl;
}

void DLASystem::setPosNeighbour
(double setpos[], double pos[],
int val) {
int x = random_max(0,100);
cout << "setPosNeighbour get's called:"
 << pos[0] << "," << pos[1] << ","
 << pos[2] << endl;

// Gravity version
/*
switch (val) {
case 0:
if ((x < prob) && (pos[0] < 0)){
setpos[0] = pos[0] + 1.0;
setpos[1] = pos[1];
setpos[2] = pos[2];
}
else if ((x < prob) && (pos[0] > 0)){
setpos[0] = pos[0] - 1.0;
setpos[1] = pos[1];
setpos[2] = pos[2];
}
else {
int randomValue = rand() % 2;
```

```
if (randomValue == 0) {
setpos[0] = pos[0] - 1.0;
} else {
setpos[0] = pos[0] + 1.0;
}
setpos[1] = pos[1];
setpos[2] = pos[2];
}
break;
case 1:
if ((x < prob) && (pos[1] < 0)){
setpos[0] = pos[0];
setpos[1] = pos[1] + 1.0;
setpos[2] = pos[2];
}
else if ((x < prob) && (pos[1] > 0)){
setpos[0] = pos[0];
setpos[1] = pos[1] - 1.0;
setpos[2] = pos[2];
}
else {
int randomValue = rand() % 2;
if (randomValue == 0) {
setpos[1] = pos[1] - 1.0;
} else {
setpos[1] = pos[1] + 1.0;
}
setpos[0] = pos[0];
setpos[2] = pos[2];
}
break;
case 2:
if ((x < prob) && (pos[2] < 0)){
setpos[0] = pos[0];
setpos[1] = pos[1];
setpos[2] = pos[2] + 1.0;
}
else if ((x < prob) && (pos[2] > 0)){
setpos[0] = pos[0];
setpos[1] = pos[1];
setpos[2] = pos[2] - 1.0;
}
else {
int randomValue = rand() % 2;
if (randomValue == 0) {
setpos[2] = pos[2] - 1.0;
} else {
setpos[2] = pos[2] + 1.0;
}
setpos[0] = pos[0];
setpos[1] = pos[1];
}
break;
}
*/
```

```
//Probability, but no gravity version

switch (val) {
case 0:
setpos[0] = pos[0] + 1.0;
setpos[1] = pos[1];
setpos[2] = pos[2];
break;
case 1:
setpos[0] = pos[0] - 1.0;
setpos[1] = pos[1];
setpos[2] = pos[2];
break;
case 2:
setpos[0] = pos[0];
setpos[1] = pos[1] + 1.0;
setpos[2] = pos[2];
break;
case 3:
setpos[0] = pos[0];
setpos[1] = pos[1] - 1.0;
setpos[2] = pos[2];
break;
case 4:
setpos[0] = pos[0];
setpos[1] = pos[1];
setpos[2] = pos[2] + 1.0;
break;
case 5:
setpos[0] = pos[0];
setpos[1] = pos[1];
setpos[2] = pos[2] - 1.0;
break;
}
}

void DLASystem::viewAddCircle() {
cout << "viewAddCircle get's called"
 << endl;
setViewSize(2.0*addCircle);
}

void DLASystem::updateClusterRadius
(double pos[]) {

cout << "updateClusterRadius get's called"
 << endl;

double rr = distanceFromOrigin(pos);
if (rr > clusterRadius) {
clusterRadius = rr;

if (addCircle < check) {
addCircle = check;
```

```
killCircle = killRatio * addCircle;
updateViewSize();
}
checkStop();
}
}

void DLASystem::moveLastParticle() {
cout << "moveLastParticle get's called"
 << endl;

int rr = random_max(0,5);
double newpos[3];

Particle *lastP =
 particleList[numParticles - 1];

setPosNeighbour(newpos,lastP->pos,rr);

if (distanceFromOrigin(newpos) >
 killCircle) {
//cout << "#deleting particle" << endl;
setGrid(lastP->pos, 0);
particleList.pop_back();
numParticles--;
setParticleInactive();
}

// check if destination is empty
else if (readGrid(newpos) == 0) {

setGrid(lastP->pos, 0);

// update the position
particleList[numParticles - 1]
  ->pos[0] = newpos[0];
particleList[numParticles - 1]
  ->pos[1] = newpos[1];
particleList[numParticles - 1]
  ->pos[2] = newpos[2];
setGrid(lastP->pos, 1);

// check if we stick
if (checkStick()) {
//cout << "stick" << endl;
int n_p = numParticles-1;

printpositions3d(n_p, newpos[0],
   newpos[1], newpos[2], clusterRadius,
   prob,G);

setParticleInactive();
if (numParticles % 100 == 0
   && logfile.is_open()) {
logfile << numParticles << " "
    << clusterRadius << endl;
}

cout << "working:" << lastP->pos[0]
   << "," << lastP->pos[1] << ","
   << lastP->pos[2] << endl;

}
}
else {
cout << "reject " << rr << endl;
cout << lastP->pos[0] << " " << lastP->pos[1]
  << " " << lastP->pos[2] << endl;
}
}

// constructor
DLASystem::DLASystem(Window *set_win) {
cout << "DLASystem get's called" << endl;
cout << "creating system, gridSize "
 << gridSize << endl;
win = set_win;
numParticles = 0;
endNum = 10;

// allocate memory for the grid,
 remember to free the memory in destructor
grid = new int**[gridSize];
for (int i = 0; i < gridSize; i++) {
grid[i] = new int*[gridSize];
for (int j = 0; j < gridSize; j++) {
grid[i][j] = new int[gridSize];
}
}
slowNotFast = 1;
// reset initial parameters
Reset();

addRatio = 1.2;
killRatio = 1.7;
}

// destructor
DLASystem::~DLASystem() {
cout << "deleting system" << endl;
// delete the particles
clearParticles();
// delete the grid
for (int i = 0; i < gridSize; i++)
delete[] grid[i];
delete[] grid;

if (logfile.is_open())
logfile.close();
}
```