Efficiency of Planar Disk Packings

Rohit Banerjee Ivan Cruz Jacob Krol Supervisor: Ali Mohajer
University of Illinois at Chicago
Mathematical Computing Laboratory

Summary

Two novel ways of creating saturated packings were explored in this project. The first one involves a gaseous phyiscs engine with variable physics factors and the second uses additively weighted Voronoi decompositions of a plane containing randomly inserted disks.

Motivation

A popular chemistry experiment shows that a mixture of certain pure liquids can be denser than each of the constituents. Mixing a liter of methanol with a liter of ethanol gives a solution with volume measurably less than 2 liters. A mathematical analog of this experiment is the fact that a packing of unequal disks in the plane can be denser than a packing of equal disks, as long as the radii of the disks are not very close. Exactly how close is ”very close” is an area of active research. This project aims to study the behavior of two-species packings, except for special ratios of radii where everything fits together very nicely. In this research project we will study randomly generated two-species packings in order to gain insight into the shape of the density bounding function.

Background

It was previously proven by Joseph Lagrange in 1773 that a single species packing density of two-dimensional disks on an infinite plane is $\frac{\pi}{\sqrt{12}}$, ($\simeq 0.9069$). However, it is still unknown what radii form the highest boundary condition, and which proportions of those radii create an optimal packing.

Generating a Palette

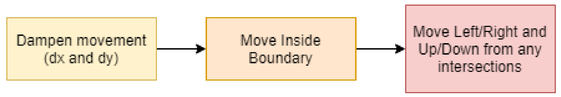
Random. Our first generator selected a random point on the palette, checked if there was room at that point, and added the new circle. This clearly was foundational, as it vastly undershot the predicted density.

Gravity. The next step was to create a physics engine so that the disks could interact as if they were truly touching each other in the physical world. A downward gravity, however, tended to separate the two species over time since the slight perturbations pushed the smaller species to the bottom. This would not be ideal since it would approximate two single species packings.

Gaseous. Our final iteration kept the concepts of the physics engine, but eliminated the gravity influence. In this stage, disks floated around freely, only affected by air resistance (to dampen shaking), other disks, and the walls.

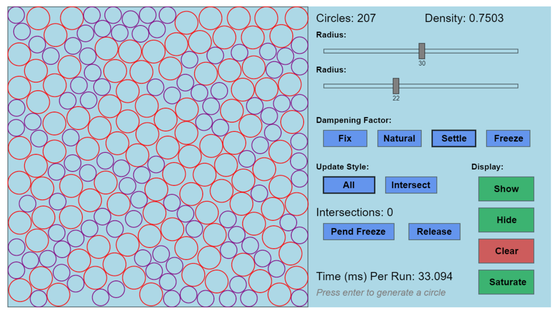
Disk Physics

Our update function utilizes mostly dx and dy values for each disk to persuade them out of each other's borders. We found using hard boundaries made it difficult for disks to order themselves neatly, since preventing movement of disks when they are tangential to another will inadvertently prevent movement of a group of disks in unison. Therefore, a very simplified form of our physics is visualized below, where these steps are evaluated for every disk in the palette.



Visualizing the Palette

To comprehend the interactions, relationships, and general tendencies of our palette, we created an interactive, live visualizer to run our packings using JavaScript. See an example of it in use below.



Button	Function
”Fix”	Sets dampening factor to 0.99
”Natural”	Sets dampening factor to 0.98
”Settle”	Sets dampening factor to 0.94
”Freeze”	Sets dampening factor to 0
”All”	All circles update
”Intersect”	Only intersecting circles update
”Pend Freeze”	Freezes palette at 0 intersections
”Release”	Resets dampening factor after ”Pend Freeze” activates
”Show” and ”Hide”	Display and Hide Circles
”Clear”	Removes all circles
”Saturate”	Runs saturate()

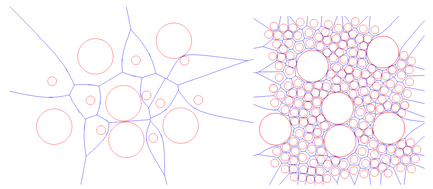
Saturation

A saturation algorithm was necessary in order to ensure the validity of density calculations from a given palette. The current methodology deletes disks slowly and allows the palette more time to settle and possibly reach zero intersections rather than attempting to add disks back into the palette post-deletion.

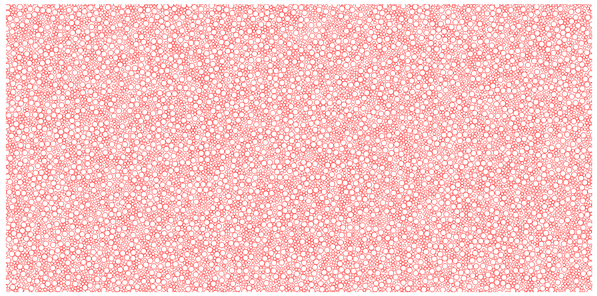
The possibility of over-deleting disks and the human attention needed to determine reasonably saturated packing presented limitations when collecting data.

Saturation (continued)

A more efficient saturation algorithm using additively weighted Voronoi diagrams was also investigated but not implemented into our physics and data collection engines.



The algorithm works by checking a disk centered on a vertex of the additively weighted Voronoi diagram for intersection with one of its nearest neighbors. This determines whether a disk can be inserted into the palette. After every insertion, new vertices are created and tested until exhaustion of non-intersecting vertices. Once exhaustion is reached, a set of static disks can be considered saturated.



Shown above is a palette consisting of 462,282 static disks of radii r=1.0 and r=0.5 with proportions of each determined arbitrarily that was created in just over seven minutes.

GPE Data Collector

Concept. In order to collect data, we constructed a series of data collection engines, so that multiple palettes could be generated without manipulation of individual parameters in every iteration.

Method.

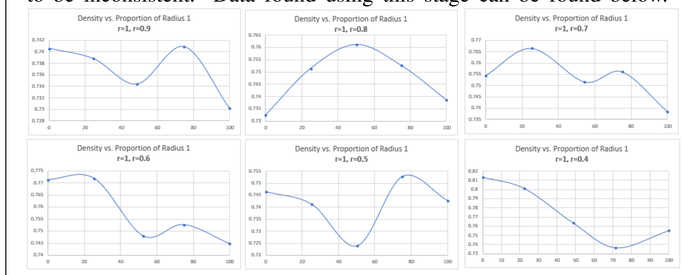
1. Prompt user for 2 radii and the number of data points.
2. Divide the spectrum of 0:100 to 100:0 ratios of the two species into the specified number of data points.
3. Fill a palette with disks according to the current proportion.
4. Update the palette until the number of intersections reaches zero.
5. Save the results in a csv file.
6. Repeat steps 3-5 until data is collected for each data point.

Building. Our first construction had a simple boundary condition, adding circles until the density reached 0.7, far below the anticipated maximum. This allowed for the first collection of data, but clearly, this is far under-estimating the palette's capabilities, and is giving results with very little utility.

Stage One. Our improvement to this C++ program shell integrated a manual saturation function within. After several models of this function, however, the ability to automate the process of filling, updating, and cleaning a palette to a high standard of saturation was unfortunately not reached.

GPE Data Collector (continued)

Stage Two. In order to still explore the validity of our saturation function, we added the same function into our JavaScript visualizer (to offer us more control by being able to see what we are manipulating), we were able to collect some data. Though it must be addressed that this data, due to a weak saturation function that fails to mathematically calculate the saturation of a palette, proved to be inconsistent. Data found using this stage can be found below.



It would be expected that a single species packing (the endpoints of the curves, where only species exists at 100%), would be the least dense, with the mixture of the two being able to fit more disks into the same space. The familiar curve from the chemistry experiment was only found in the case of r=1 and r=0.8, which suggests that a more rigorous saturation function is needed to obtain more precise data.

Results

While preliminary data on our saturation algorithm was collected manually via our Stage Two data collector, there are still two glaring limitations of these results. First, a lack of a proper saturation function leaves the program to decide incorrectly when a palette is full. Second, without enough computing power, our disks are relatively large compared to their boundary size, leaving our results not closely approximating an infinite plane as originally desired.

Conclusions

In this project we have created a gaseous physics engine with friction, a working Voronoi decomposition-based saturation algorithm, and a data engine for variable disk proportions. Further incorporation of the additively weighted Voronoi saturation algorithm with an efficient physics engine should be pursued in order to obtain a comprehensive insight into the density bounding function of two-species disk packings.

References

Hale, Thomas C. *Cannonballs and Honeycombs* Notices of the AMS **Volume 47, Number 4**.(April 2000), 440-449.

Heppes, Aladar. *Some Densest Two-Size Disc Packings in the Plane*. Discrete and Computational Geometry **30:241262**. (2003), 241-262.

Lagarias, Jeffrey C. *The Kepler Conjecture and Its Proof* Godfrey Tampach. (1611), 3-26.

Karavelas, Menelaos and Yvinec, Mariette. *Apollonius Graphs (Delaunay Graphs of Disks)* CGAL Editorial Board **4.12**. 2018.