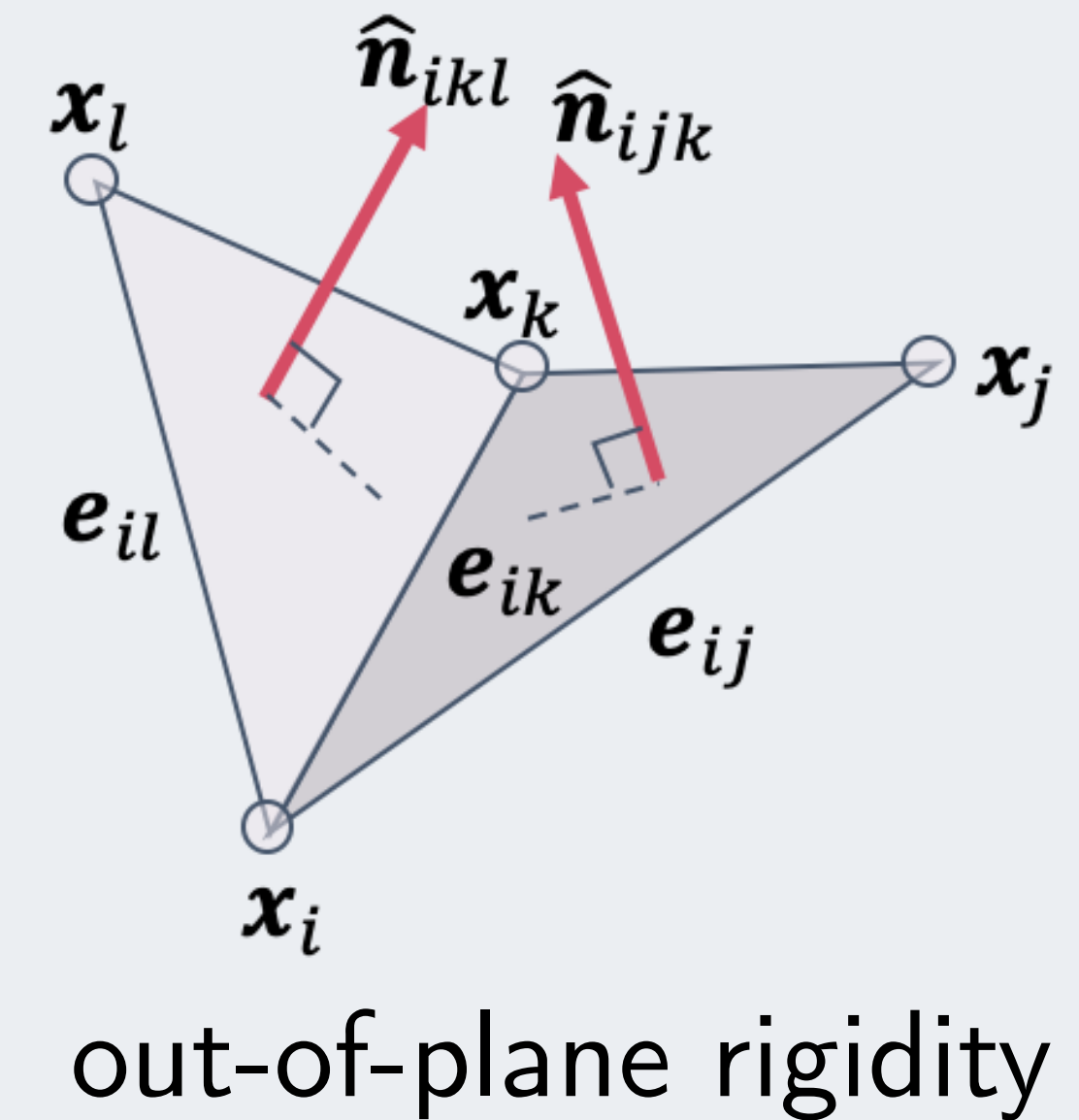
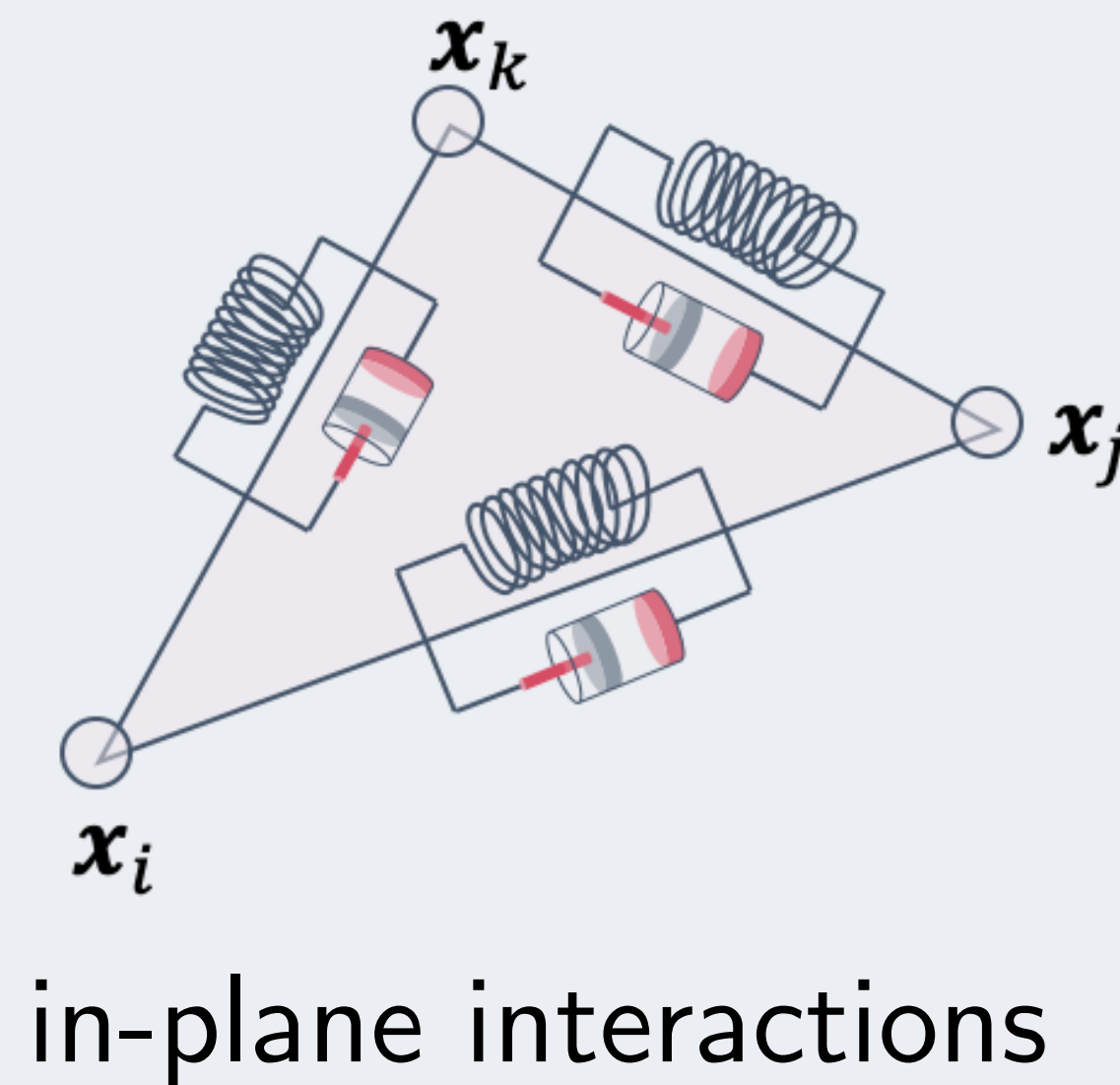
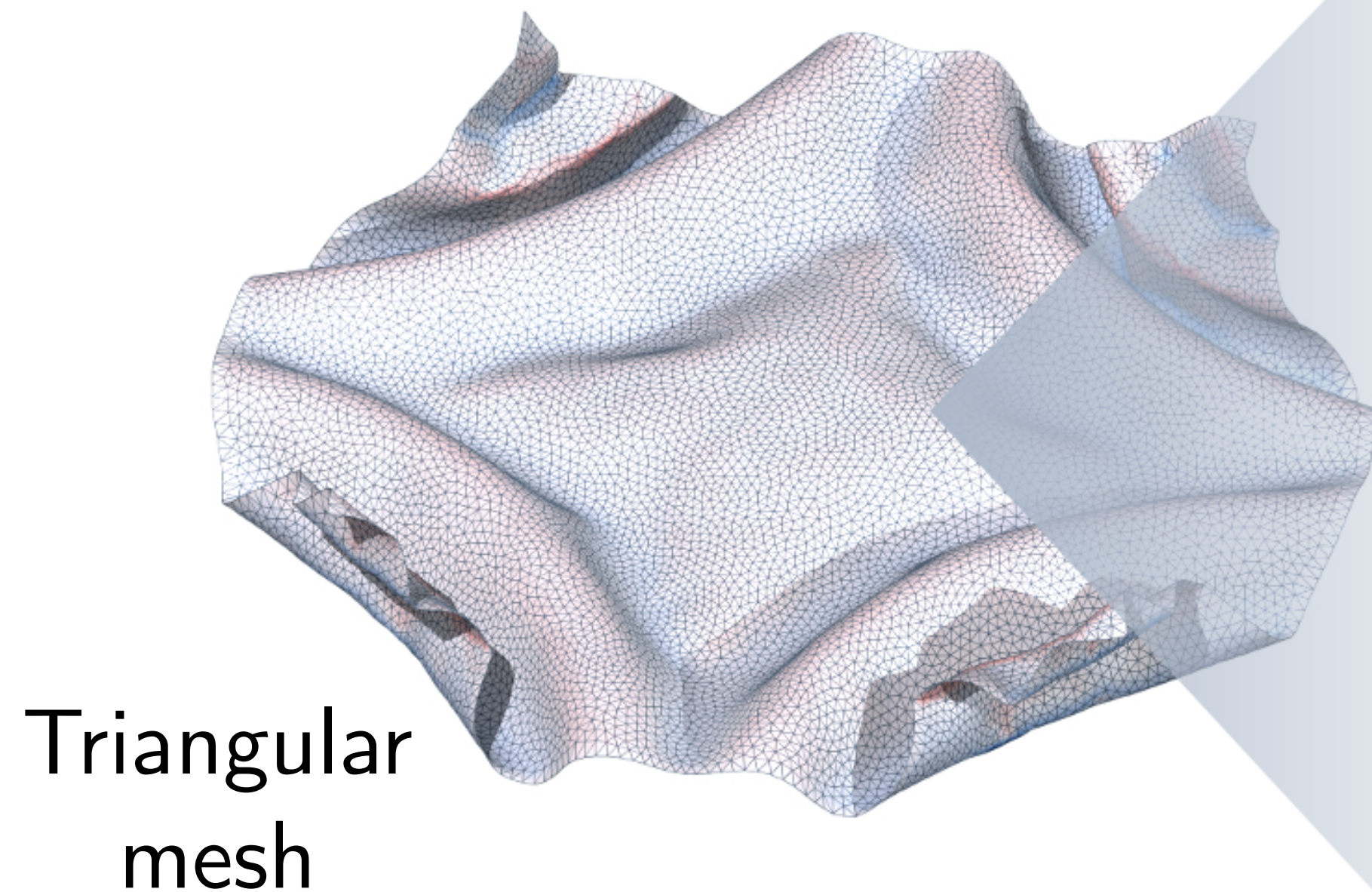


Computational models of elastic sheets II

2026 Boulder Summer School for Condensed Matter and Materials Physics
June 6–10, 2026

Chris H. Rycroft, University of Wisconsin–Madison
(rycroft@wisc.edu)

Building a mesh for sheet simulation



- In the previous lecture, we introduced a model for simulating an elastic sheet based on a triangular mesh
- We will now look at techniques from computational geometry for building meshes like this

Associated software on GitHub

- The remaining two lectures will have a number of code examples
- I have put supporting software on GitHub at the following location:

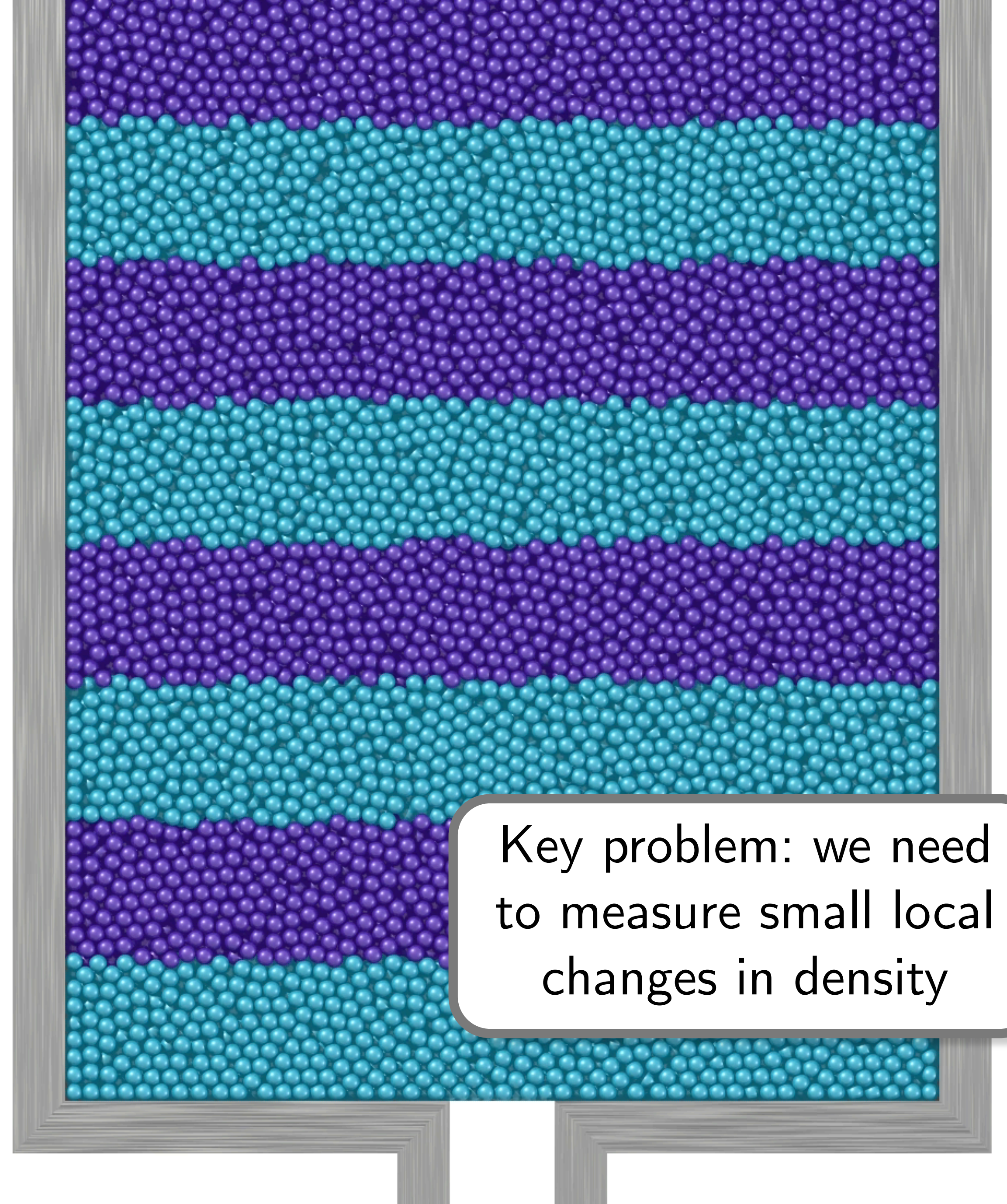
<https://github.com/chr1shr/bss2026>

Motivating example: density changes in dense granular drainage

Consider a dense granular sphere packing flowing out of an orifice by gravity, simulated with the discrete-element method (DEM)

Some models are based on introducing *free volume* at the orifice that propagates upward, moving the particles down.

Can we track these small changes in volume?



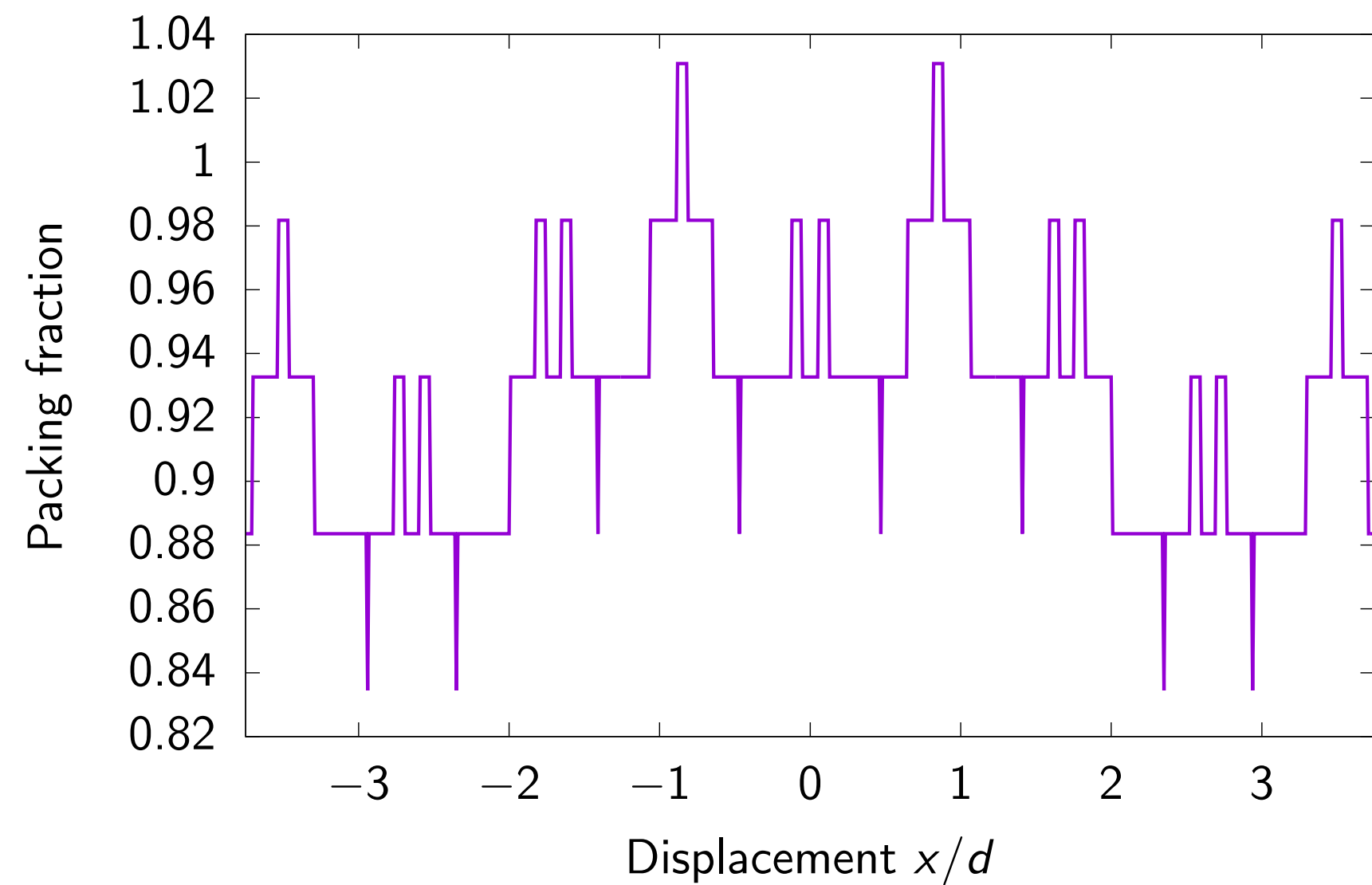
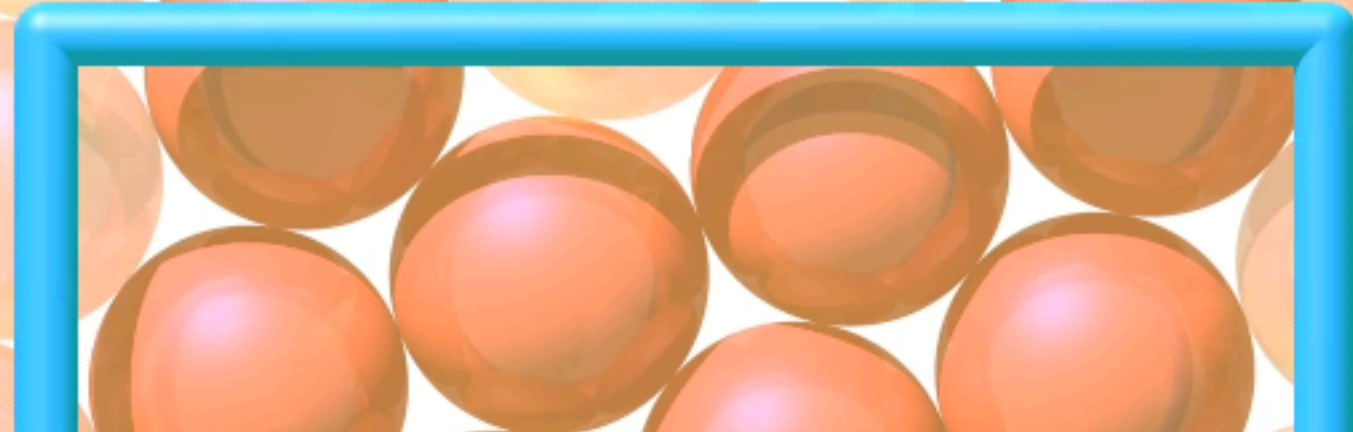
Key problem: we need to measure small local changes in density

W. W. Mullins, J. Appl. Phys. **43**, 665–678 (1972).

H. Caram and D. C. Hong, Phys. Rev. Lett. **67**, 828–831 (1993).

C. H. Rycroft *et al.*, Phys. Rev. E **73**, 051306 (2006).

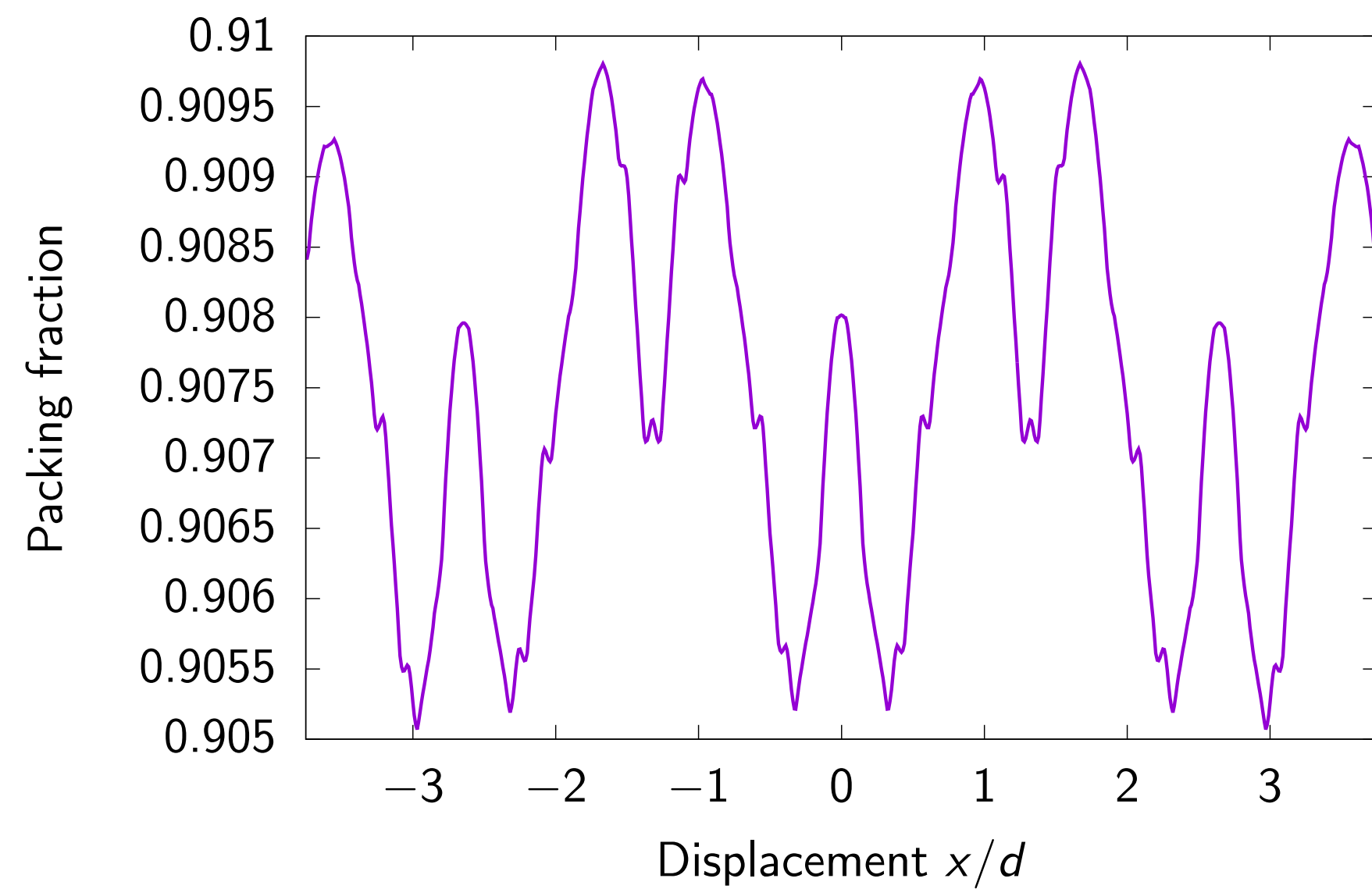
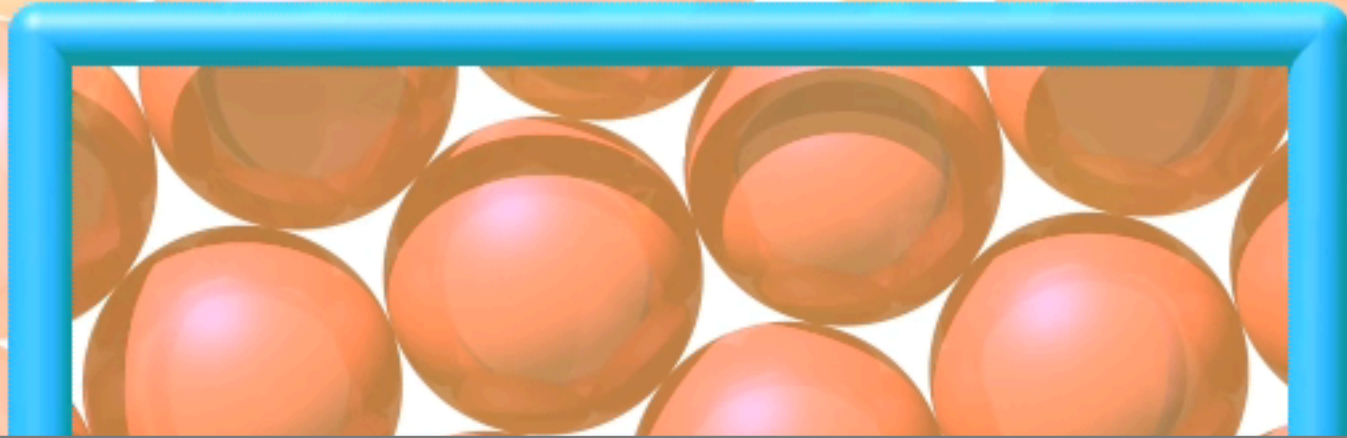
Measuring 2D density (version 1)



Packing fraction varies by 20%,
and sometimes exceeds 100%

We use **packing fraction**, the proportion of space occupied by particles, as a proxy for density

Measuring 2D density (version 2)



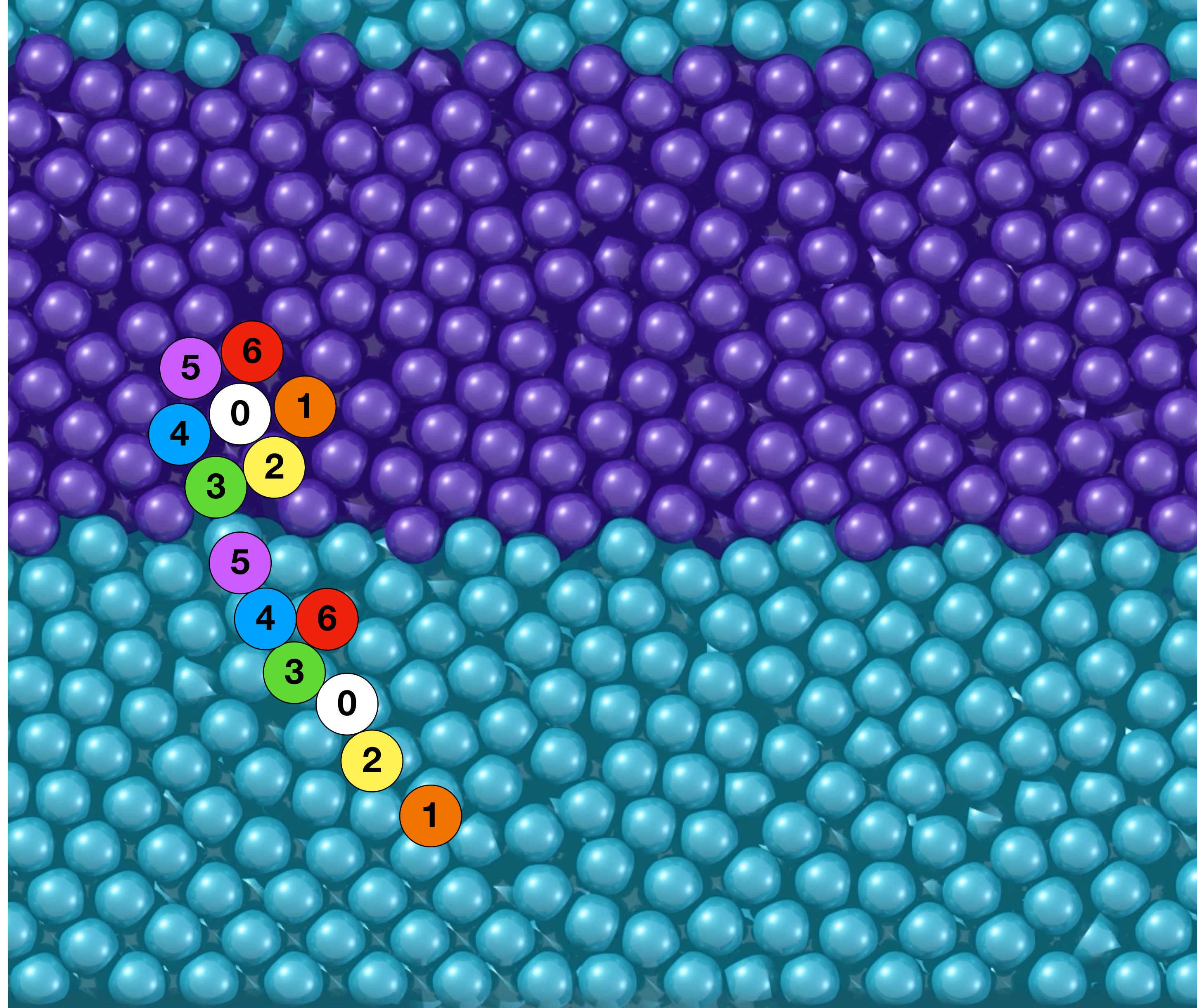
More accurate, but still
has variations of 0.5%

Another question: packing structure

Question

The DEM simulation realistically models how particles move past each other*

How do we track the neighbor relations of particles during flow?



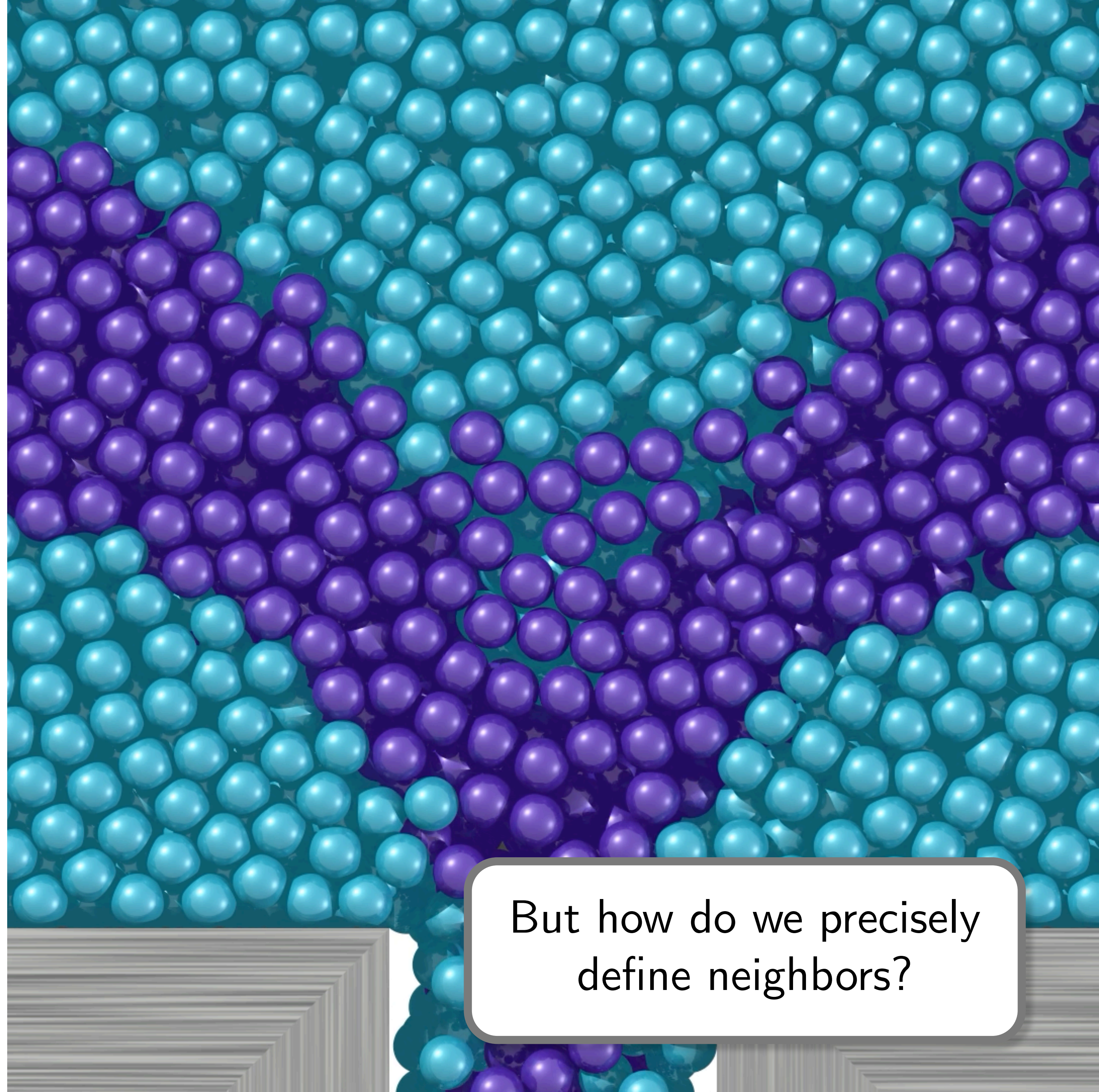
* C. H. Rycroft *et al.*, *Physical test of a particle simulation model in a sheared granular system*, Phys. Rev. E **80**, 031305 (2009).

Another question: packing structure

Question

The DEM simulation realistically models how particles move past each other*

How do we track the **neighbor** relations of particles during flow?



But how do we precisely define neighbors?

* C. H. Rycroft *et al.*, *Physical test of a particle simulation model in a sheared granular system*, Phys. Rev. E **80**, 031305 (2009).

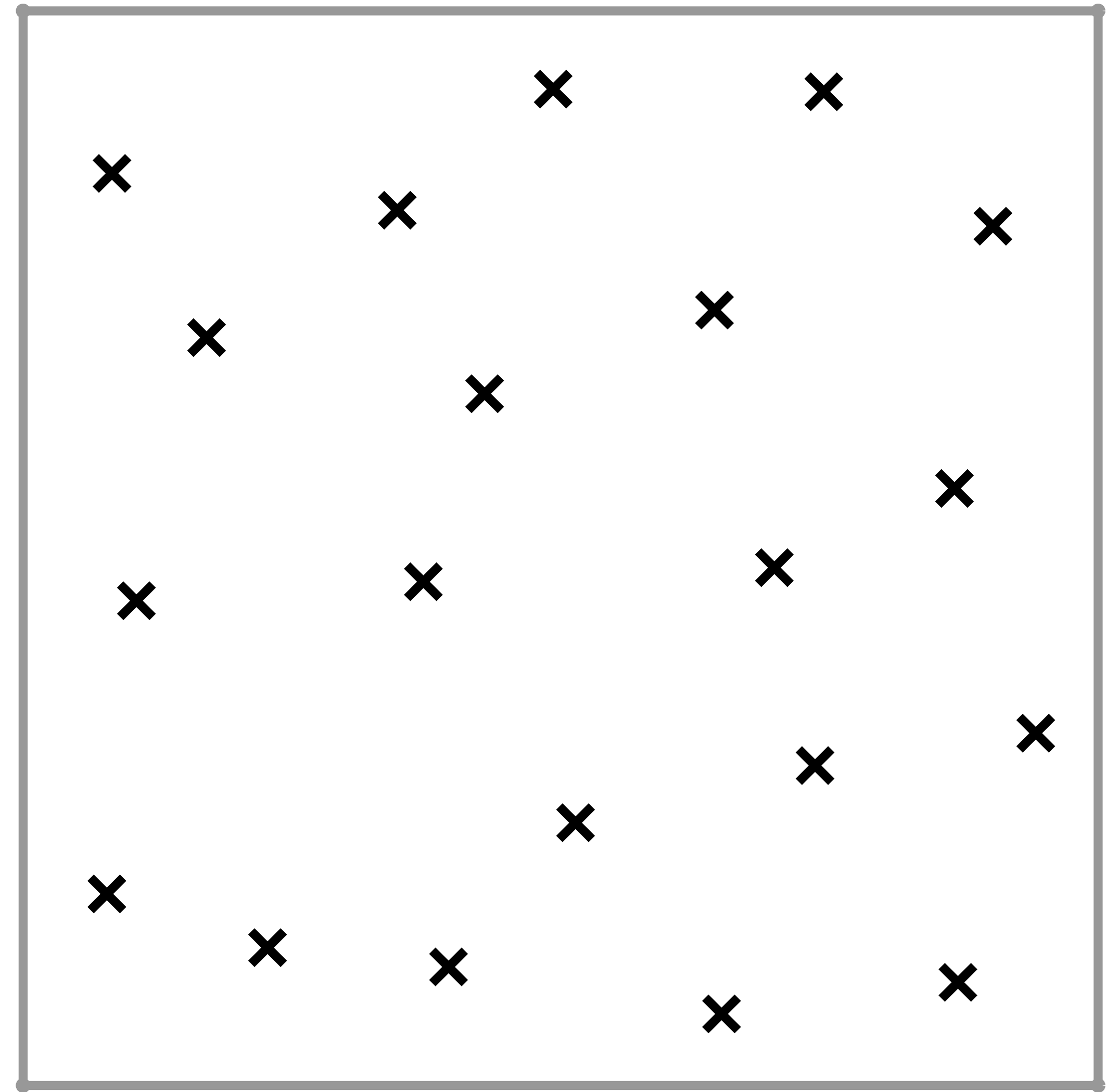
A technique from computational geometry

- Consider a domain Ω with distance metric d , and introduce a set of points x_i
- The **Voronoi cell** for point* x_i is the space x that satisfies

$$d(x, x_i) < d(x, x_j)$$

for any $j \neq i$

- Together the Voronoi cells form the **Voronoi tessellation** of Ω



* We can think of *point* and *particle* as synonymous in this definition

A. Okabe *et al.*, *Spatial tessellations: Concepts and Applications of Voronoi Diagrams*, Wiley, 2000.

E. A. Lazar *et al.*, *Voronoi cell analysis: the shapes of particle systems*, Am. J. Phys. **90**, 469 (2022).

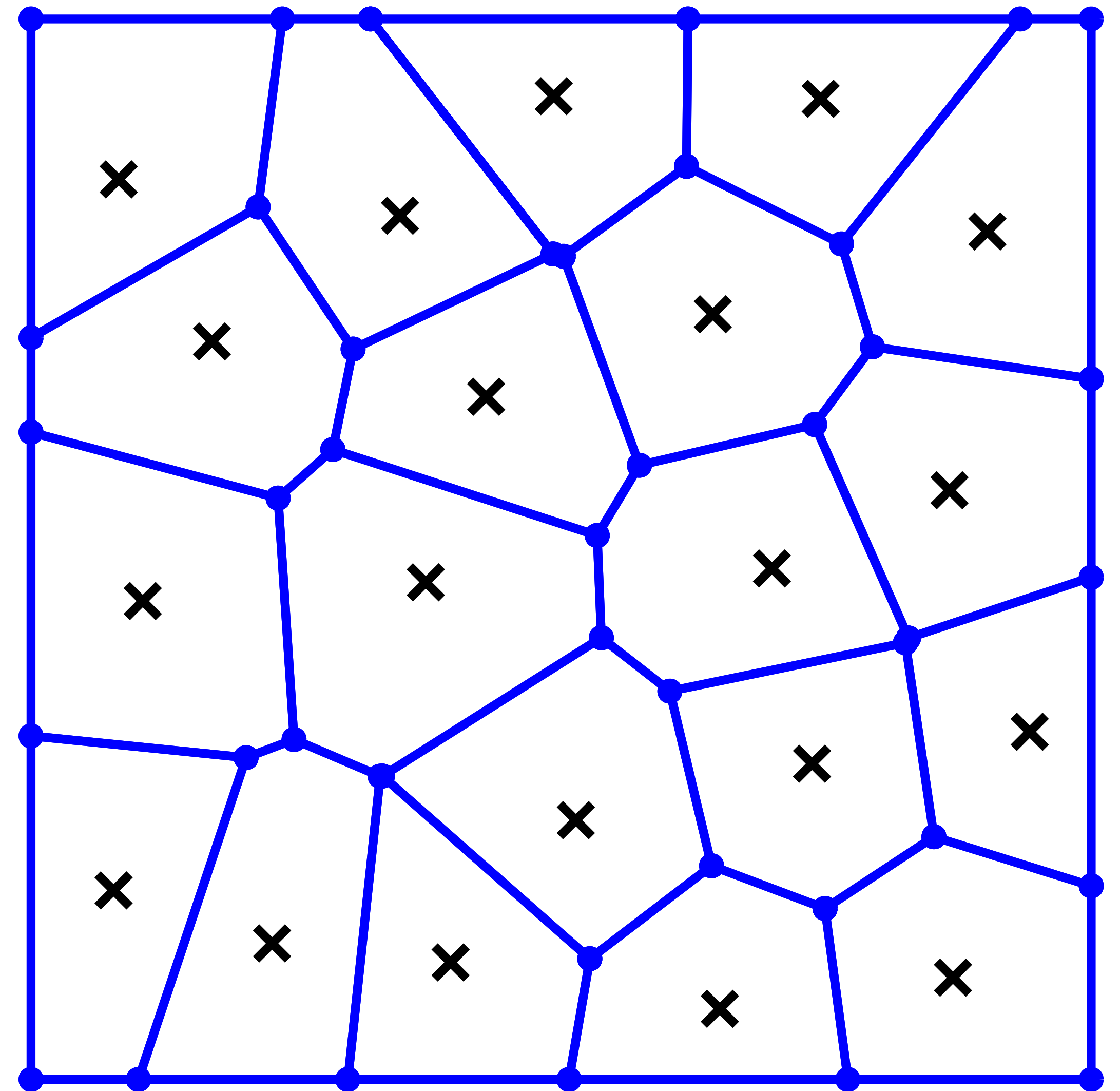
A technique from computational geometry

- Consider a domain Ω with distance metric d , and introduce a set of points x_i
- The **Voronoi cell** for point* x_i is the space x that satisfies

$$d(x, x_i) < d(x, x_j)$$

for any $j \neq i$

- Together the Voronoi cells form the **Voronoi tessellation** of Ω



* We can think of *point* and *particle* as synonymous in this definition

A. Okabe *et al.*, *Spatial tessellations: Concepts and Applications of Voronoi Diagrams*, Wiley, 2000.

E. A. Lazar *et al.*, *Voronoi cell analysis: the shapes of particle systems*, Am. J. Phys. **90**, 469 (2022).

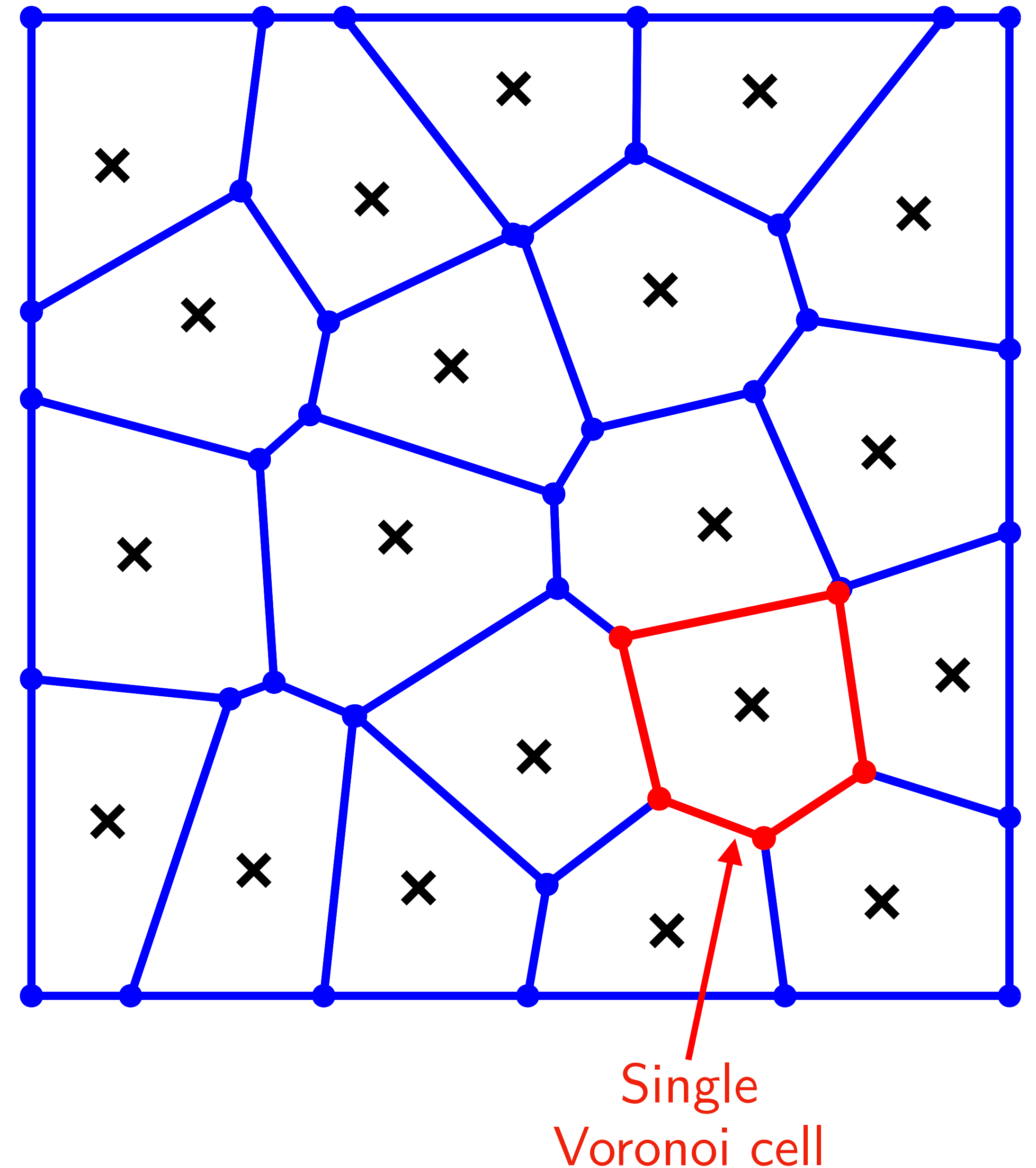
A technique from computational geometry

- Consider a domain Ω with distance metric d , and introduce a set of points x_i
- The **Voronoi cell** for point* x_i is the space x that satisfies

$$d(x, x_i) < d(x, x_j)$$

for any $j \neq i$

- Together the Voronoi cells form the **Voronoi tessellation** of Ω



* We can think of *point* and *particle* as synonymous in this definition

A. Okabe *et al.*, *Spatial tessellations: Concepts and Applications of Voronoi Diagrams*, Wiley, 2000.

E. A. Lazar *et al.*, *Voronoi cell analysis: the shapes of particle systems*, Am. J. Phys. **90**, 469 (2022).

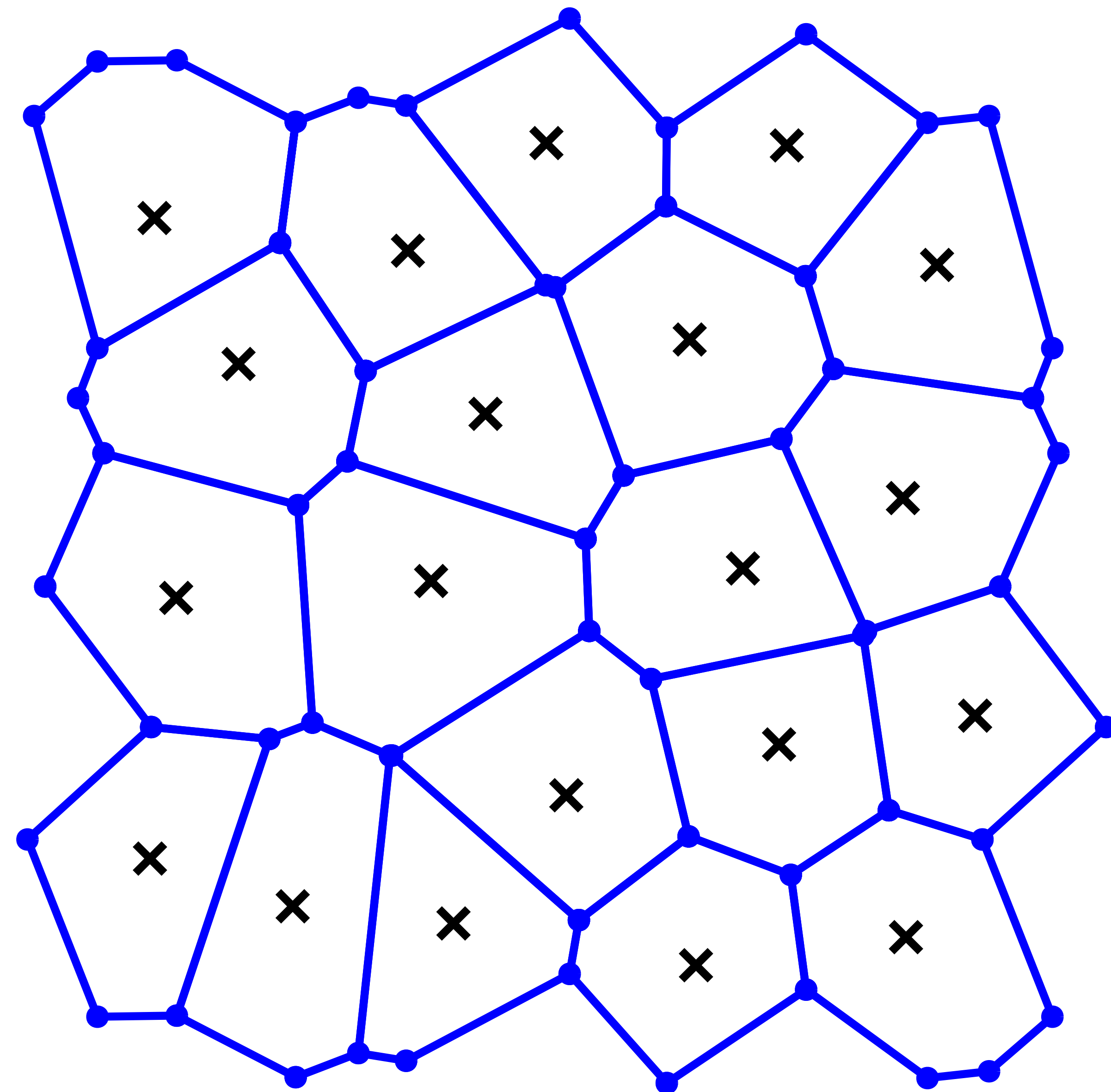
A technique from computational geometry

- Consider a domain Ω with distance metric d , and introduce a set of points x_i
- The **Voronoi cell** for point* x_i is the space x that satisfies

$$d(x, x_i) < d(x, x_j)$$

for any $j \neq i$

- Together the Voronoi cells form the **Voronoi tessellation** of Ω



(Can also use periodic boundary conditions)

* We can think of *point* and *particle* as synonymous in this definition

A. Okabe *et al.*, *Spatial tessellations: Concepts and Applications of Voronoi Diagrams*, Wiley, 2000.

E. A. Lazar *et al.*, *Voronoi cell analysis: the shapes of particle systems*, Am. J. Phys. **90**, 469 (2022).

Some history



- Developed by Georgy Voronyi (1868–1908) in Imperial Russia (present day Ukraine)
- Introduced in two pure mathematics papers in 1908
- Commemorated in 2008 in Ukraine with two-hryvnia coin



G. Voronoï, *Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Premier mémoire* ..., J. reine angew. Math. **133**, 97–179 (1908).

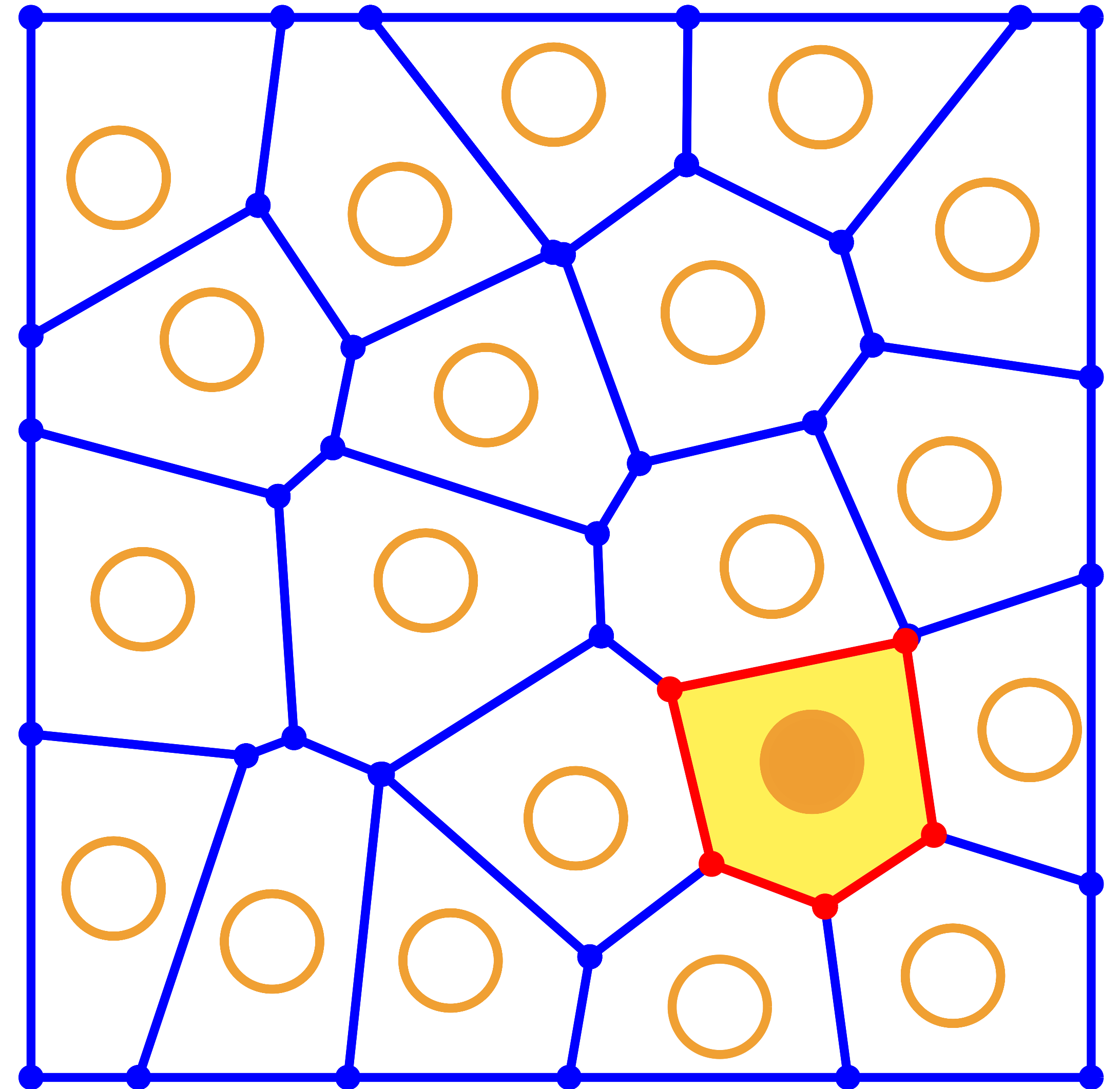
G. Voronoï, *Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire* ..., J. reine angew. Math. **134**, 198–287 (1908).

Accurate density measurements

- The Voronoi tessellation provides method to compute the packing fraction ϕ accurately
- For a single particle, measure packing fraction as

$$\phi = \frac{\text{Particle volume}}{\text{Voronoi cell volume}}$$

- Could also be average over several particles and Voronoi cells



Revisiting 2D hexagonal packing

Each particle has area

$$V_p = \frac{\pi d^2}{4}$$

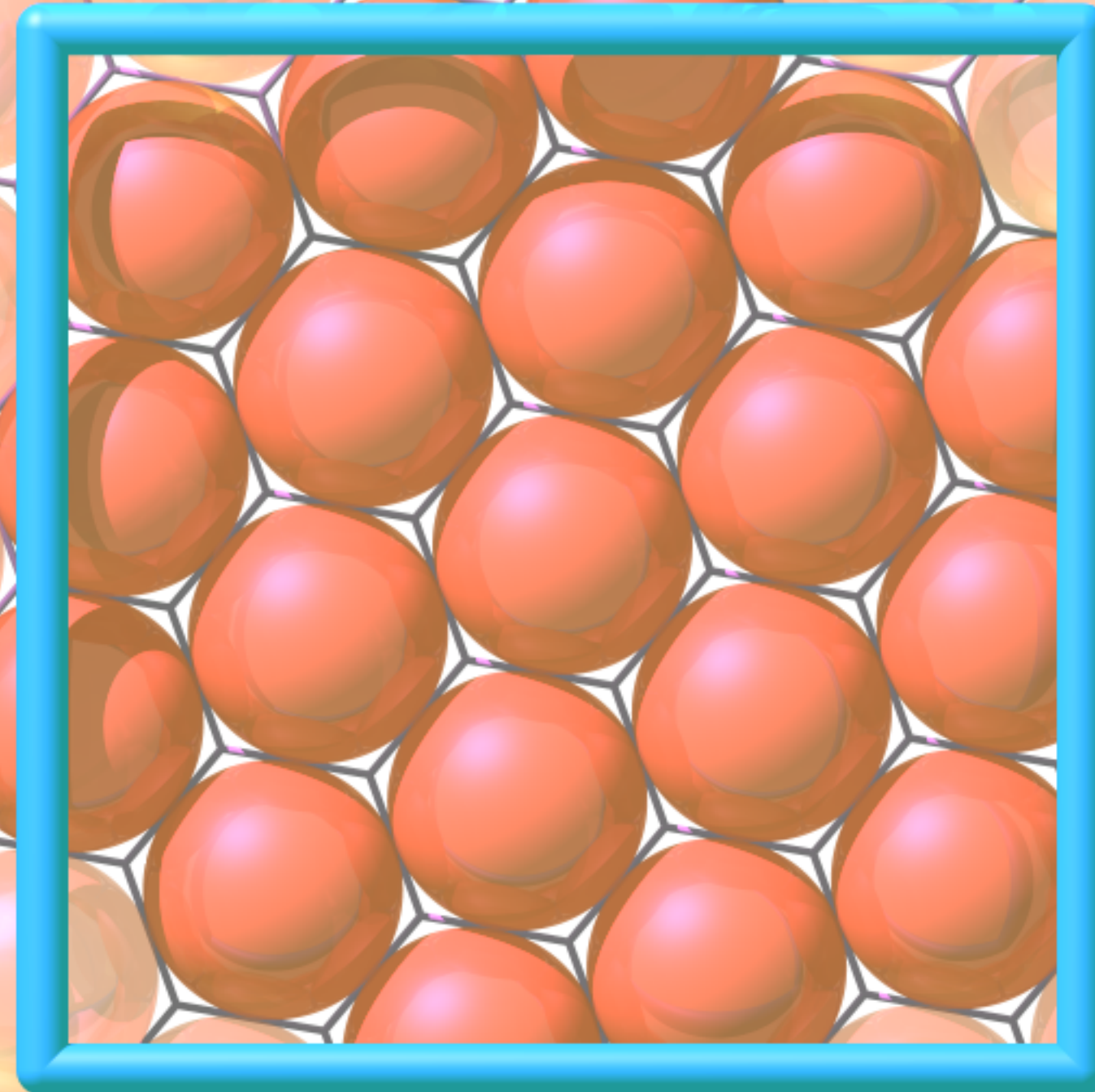
Computing the Voronoi cells

The Voronoi cells will
all be regular hexagons

Each has area

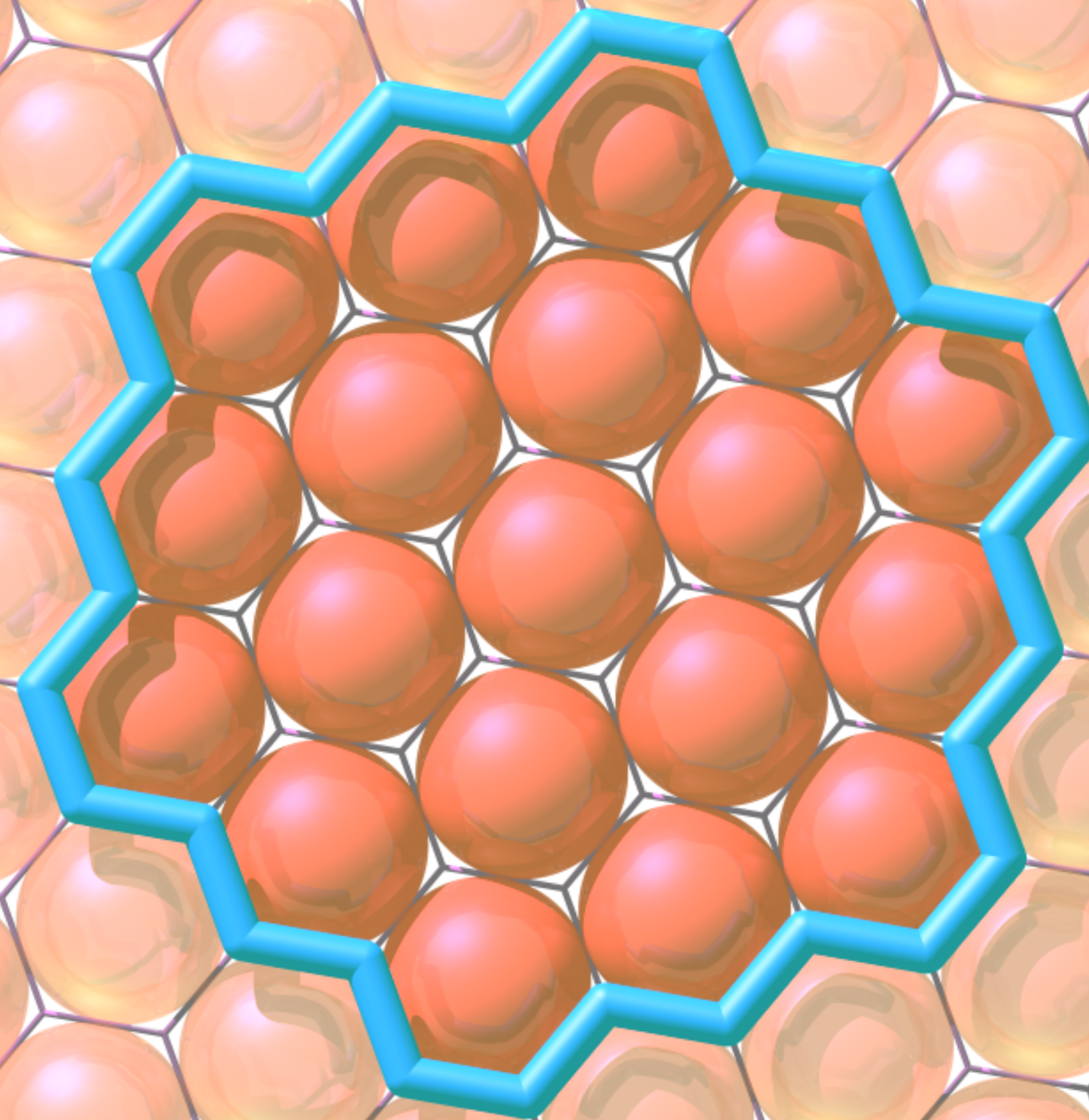
$$V_c = \frac{d^2 \sqrt{3}}{2}$$

Examining particles within test box



Consider all of the 19
particles and Voronoi
cells within a box

Examining particles within test box



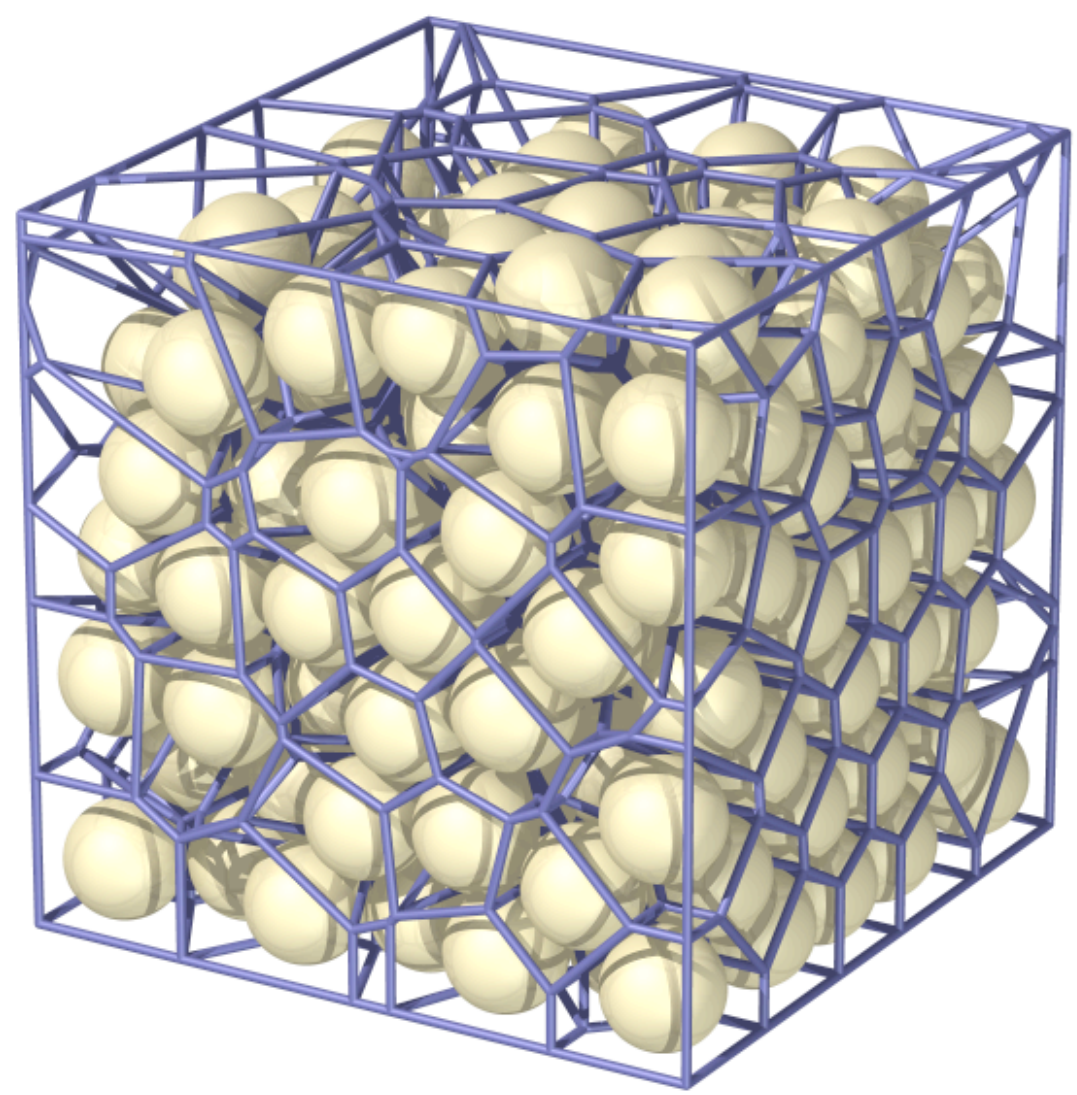
Calculate packing fraction based on

$$\phi = \frac{19V_p}{19V_c} = \frac{\pi}{2\sqrt{3}} = 90.69\%$$

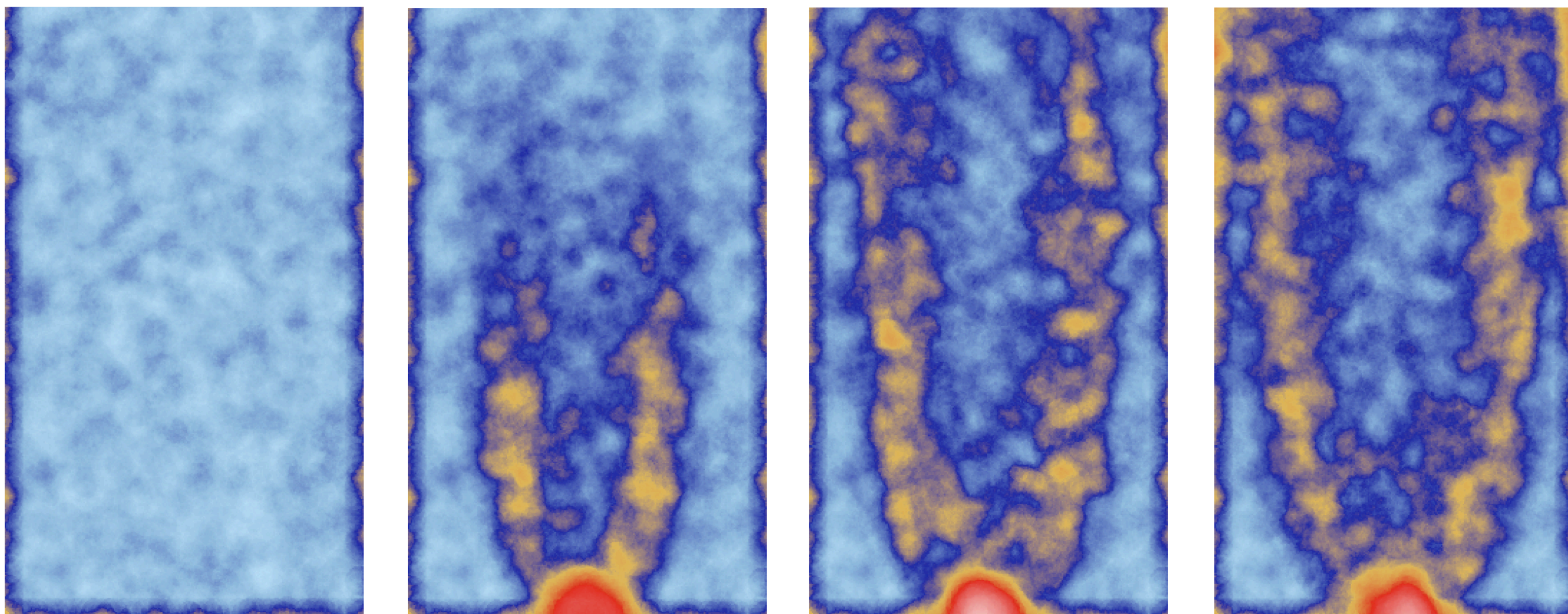
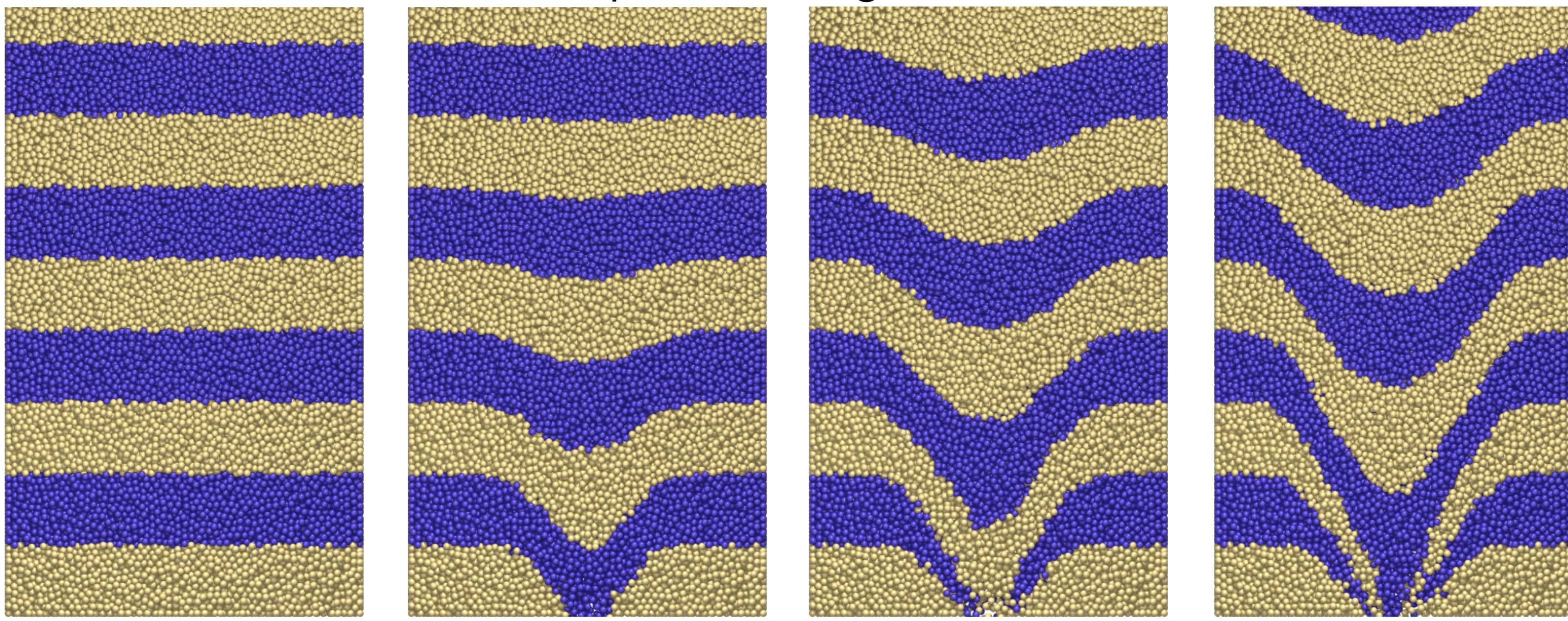
Even based on a single particle, this precisely matches the asymptotic value for large regions

Density variations

based on 3D Voronoi cell volumes in local patches



DEM snapshots through central slice



Local packing fraction

→ increasing time

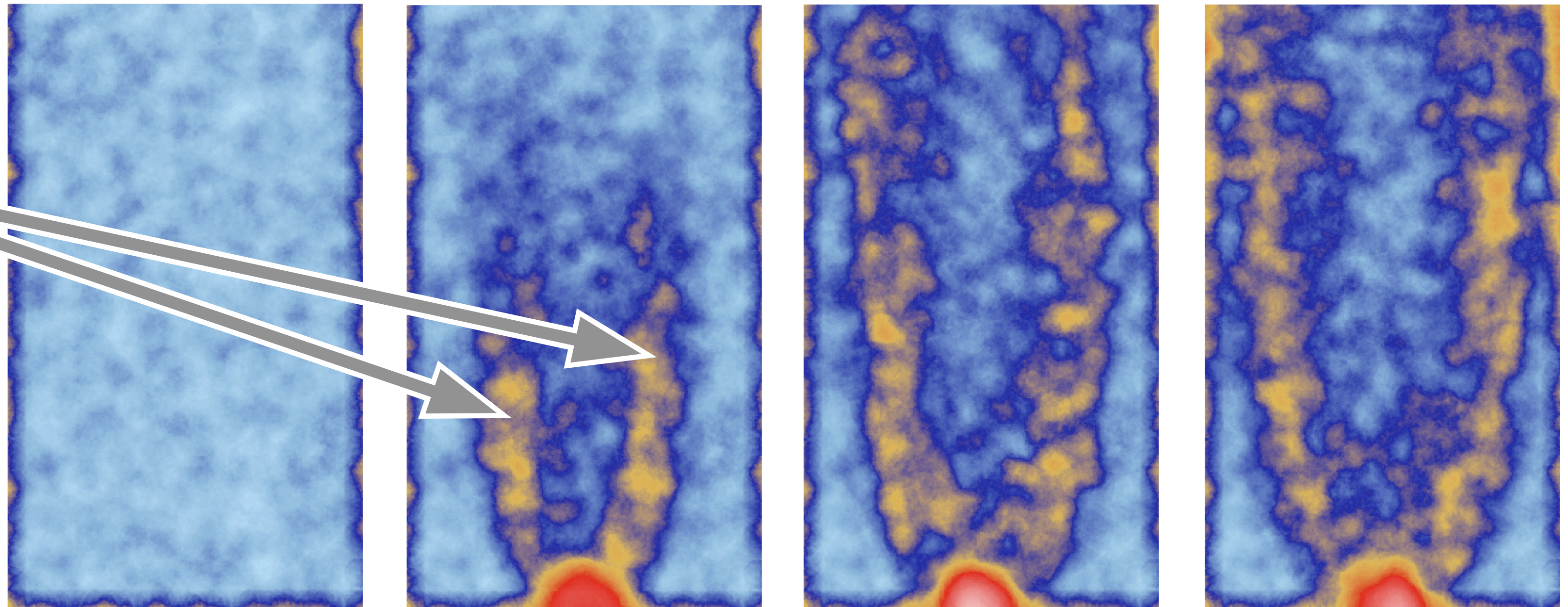
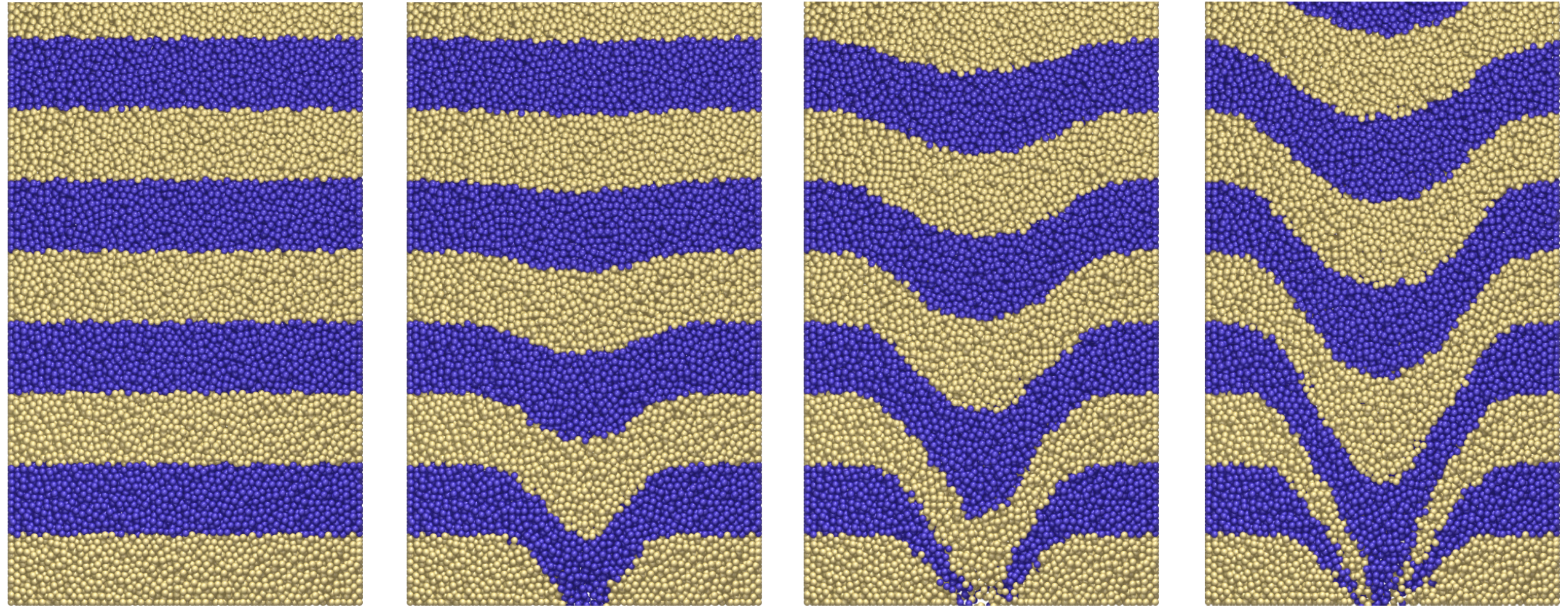
Density variations

based on 3D Voronoi cell volumes in local patches

Packing fraction is reduced in regions of **highest shear**, not highest velocity

Makes physical sense, since particles need room to flow past each other

DEM snapshots through central slice



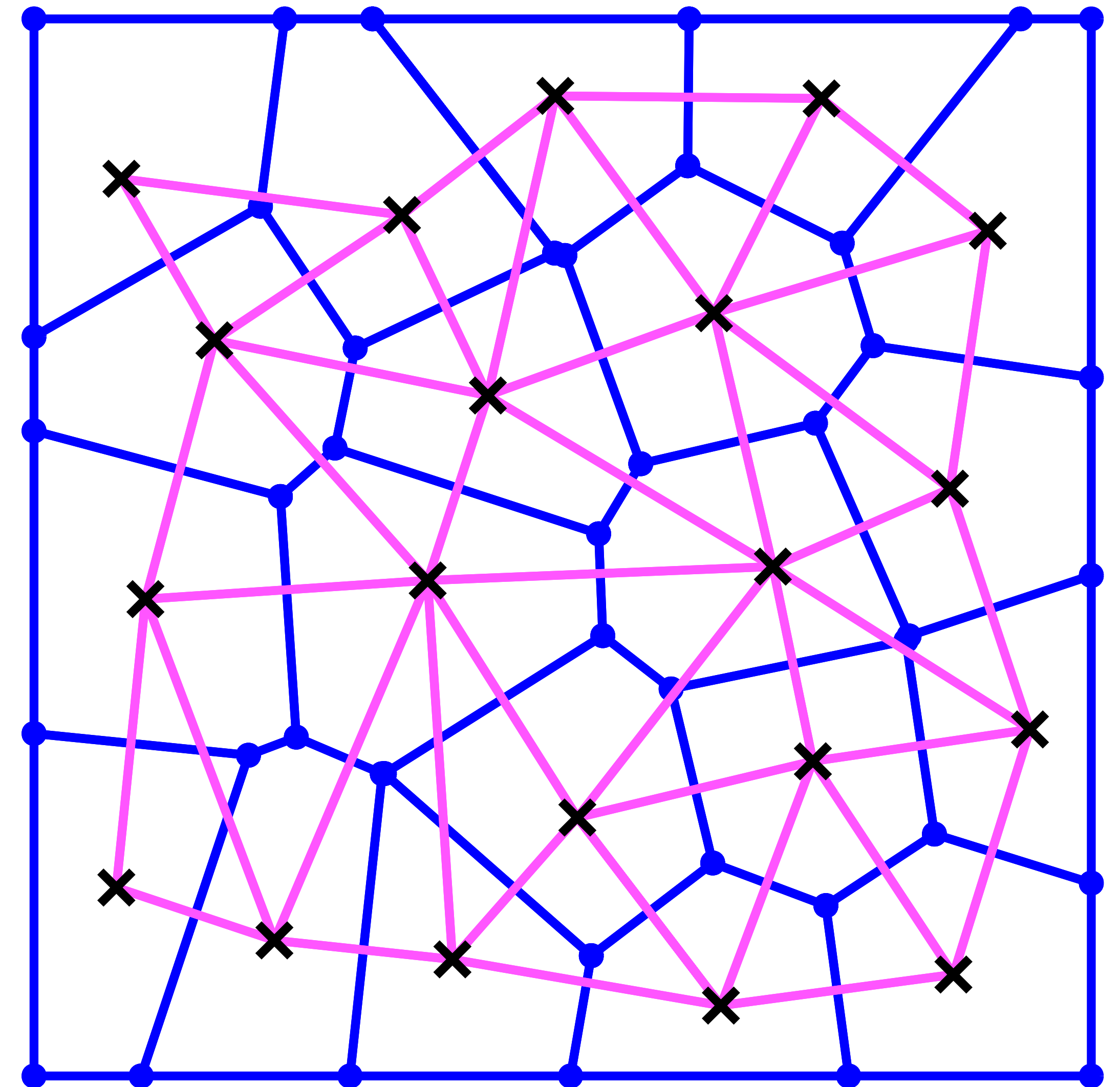
Local packing fraction

→ increasing time



A topological definition of neighbors

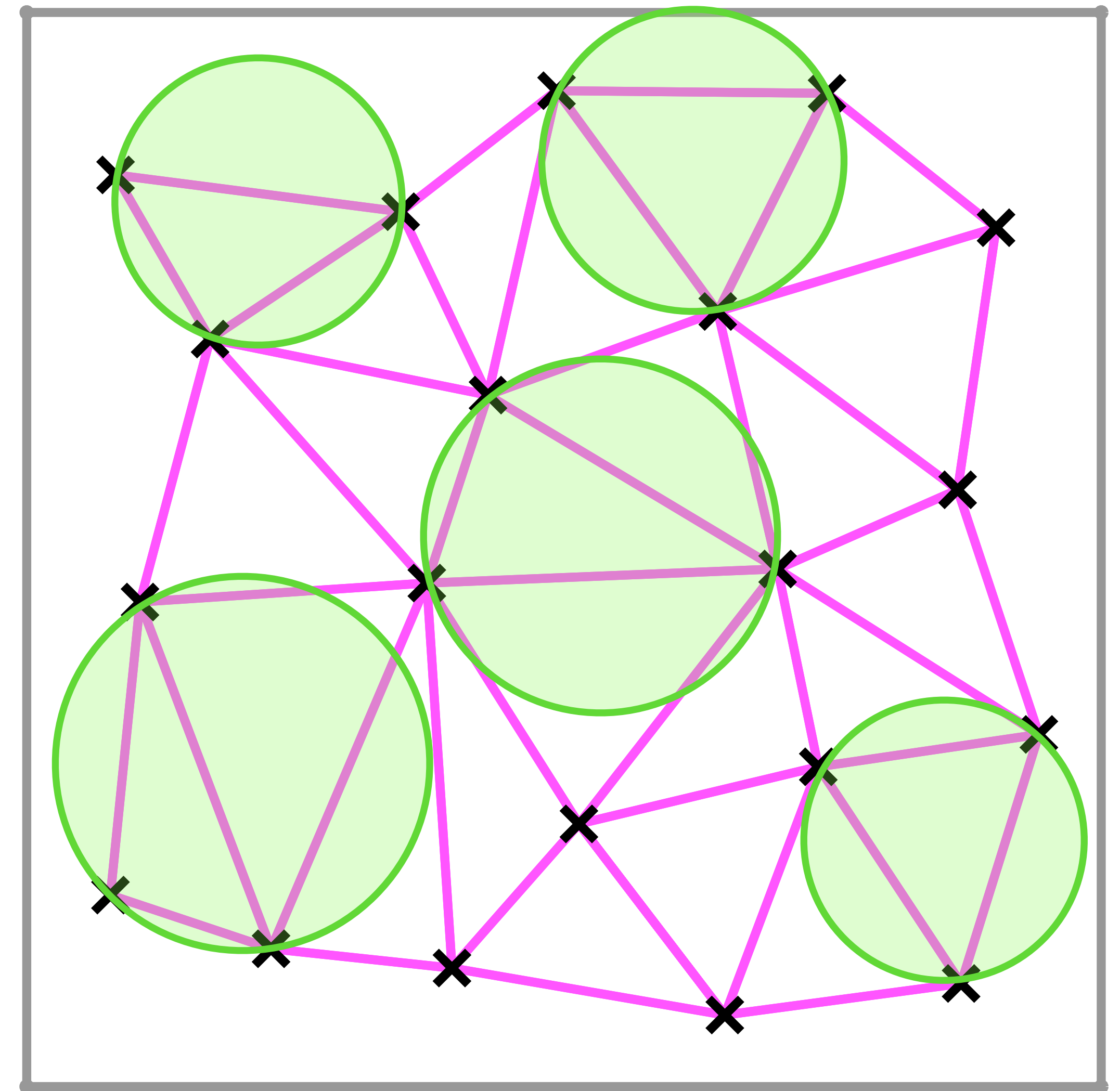
- We can define two particles as neighbors if their Voronoi cells share a face
- Connecting all these neighbors gives the **Delaunay triangulation**
- Each Delaunay edge corresponds to a Voronoi face; this is called **Voronoi–Delaunay duality**

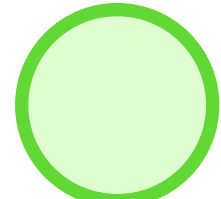


———— Delaunay triangulation

The Delaunay triangulation

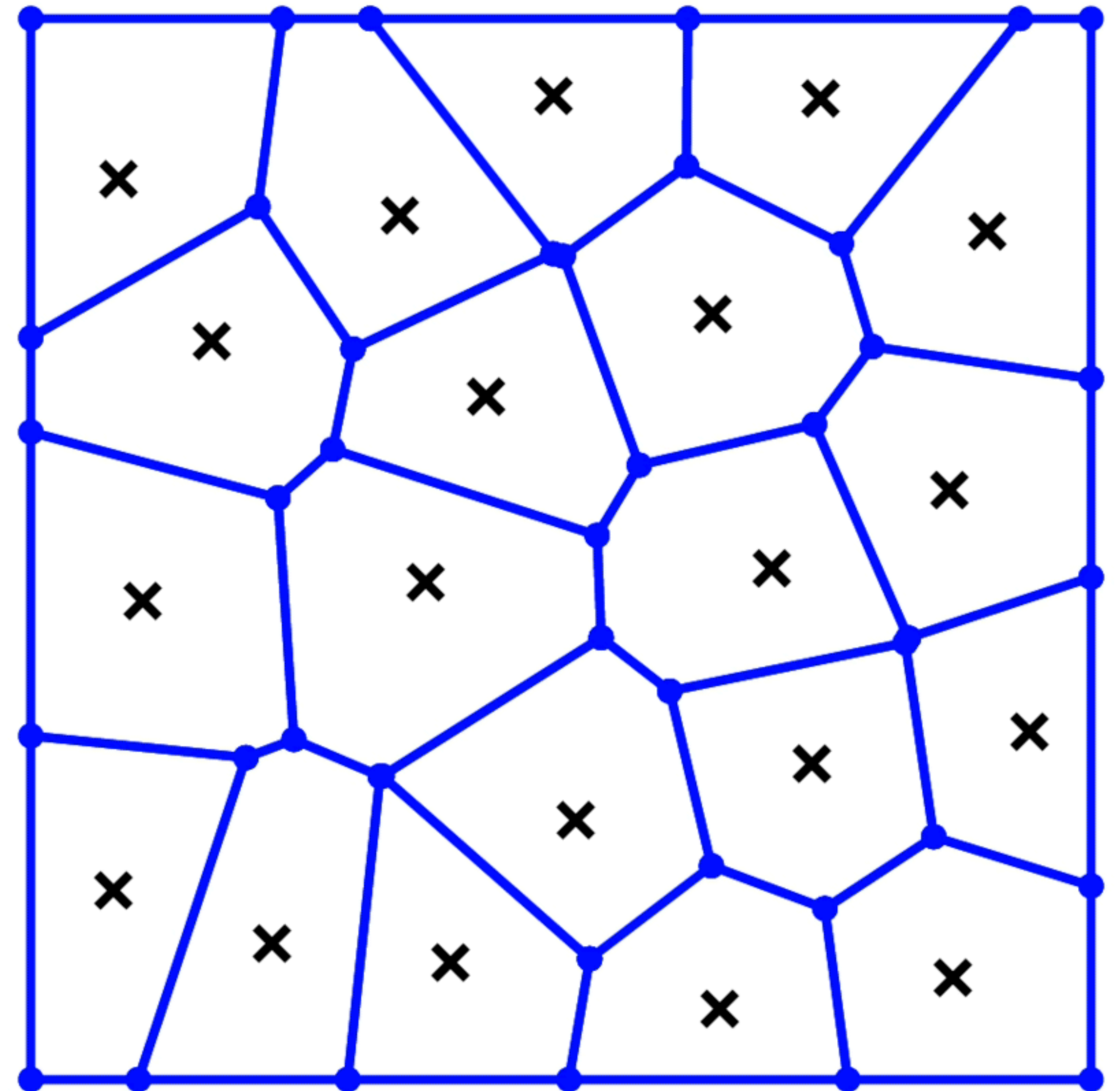
- The Delaunay triangulation is interesting in its own right, and can be defined independently of the Voronoi tessellation
- It is the unique triangulation where the circumcircle of each triangle contains no other particle
- Useful basis for triangulation



 Example circumcircles

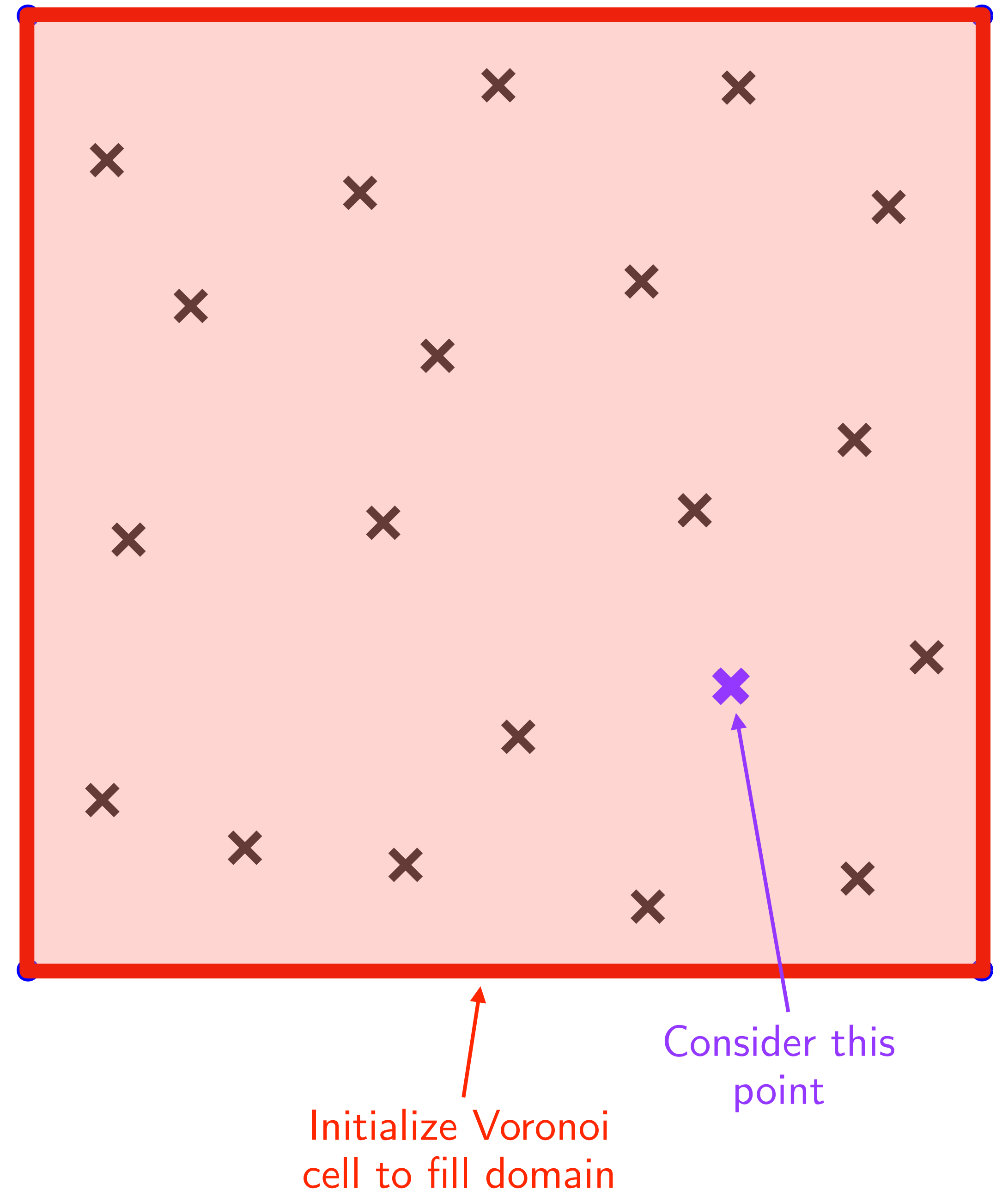
Calculating an individual Voronoi cell

- There are a number of algorithms for computing the Voronoi tessellation:
 - **Incremental approach:** add particles one-by-one and recompute the vertices each time
 - **Fortune's sweeping algorithm:** builds the tessellation by sweeping an advancing front across the domain
 - **Quickhull algorithm:** finding the tessellation in $\mathbb{R}^n$ can be converted into computing a convex hull in $\mathbb{R}^{n+1}$; used by the *Qhull* software package



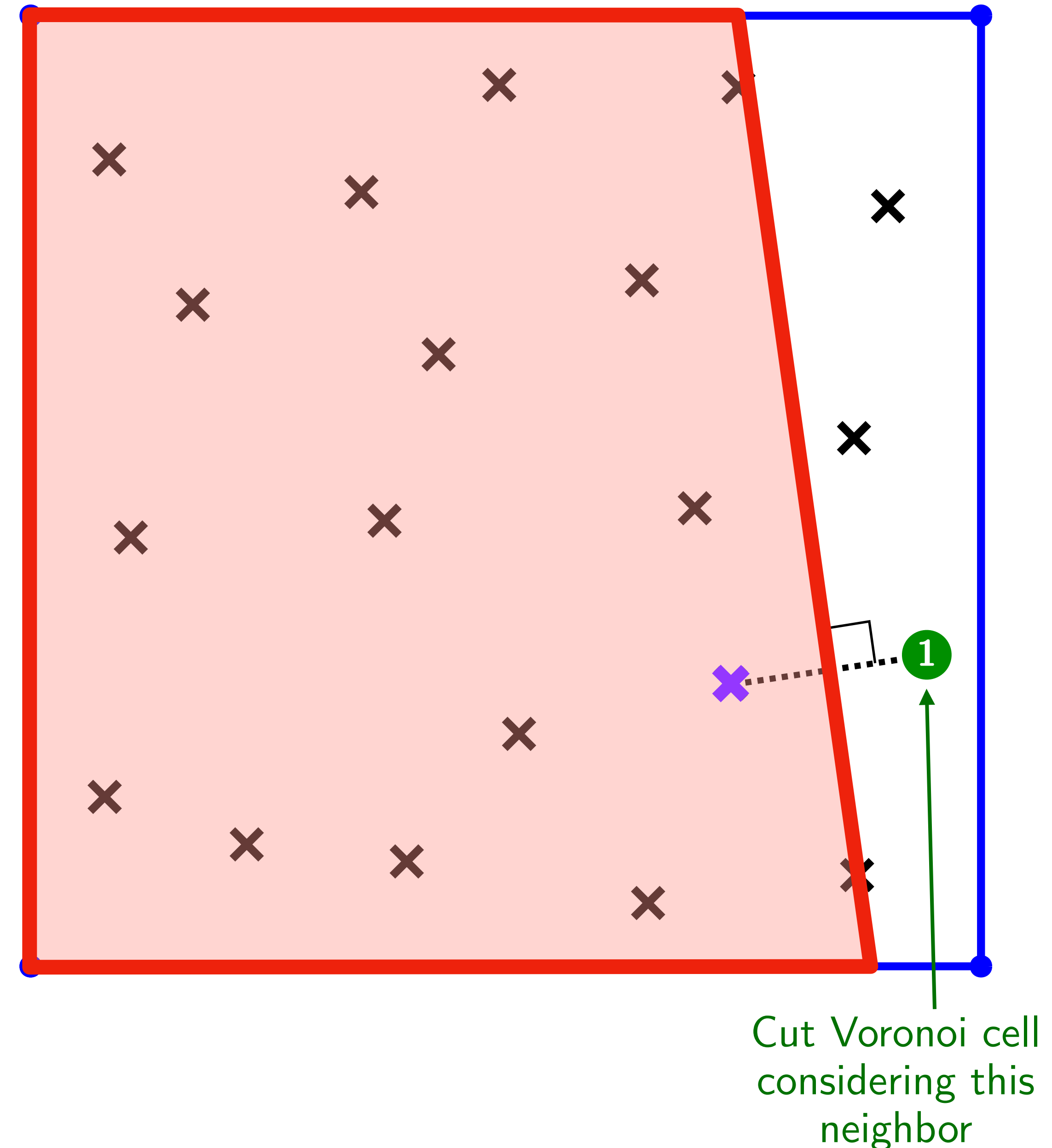
Calculating an individual Voronoi cell

- An alternative method is to compute each Voronoi cell separately
- Initialize Voronoi cell to fill Ω
- Each neighboring particle cuts the Voronoi cell by a plane that is a perpendicular bisector
- After enough nearby particles are considered, the Voronoi cell is complete



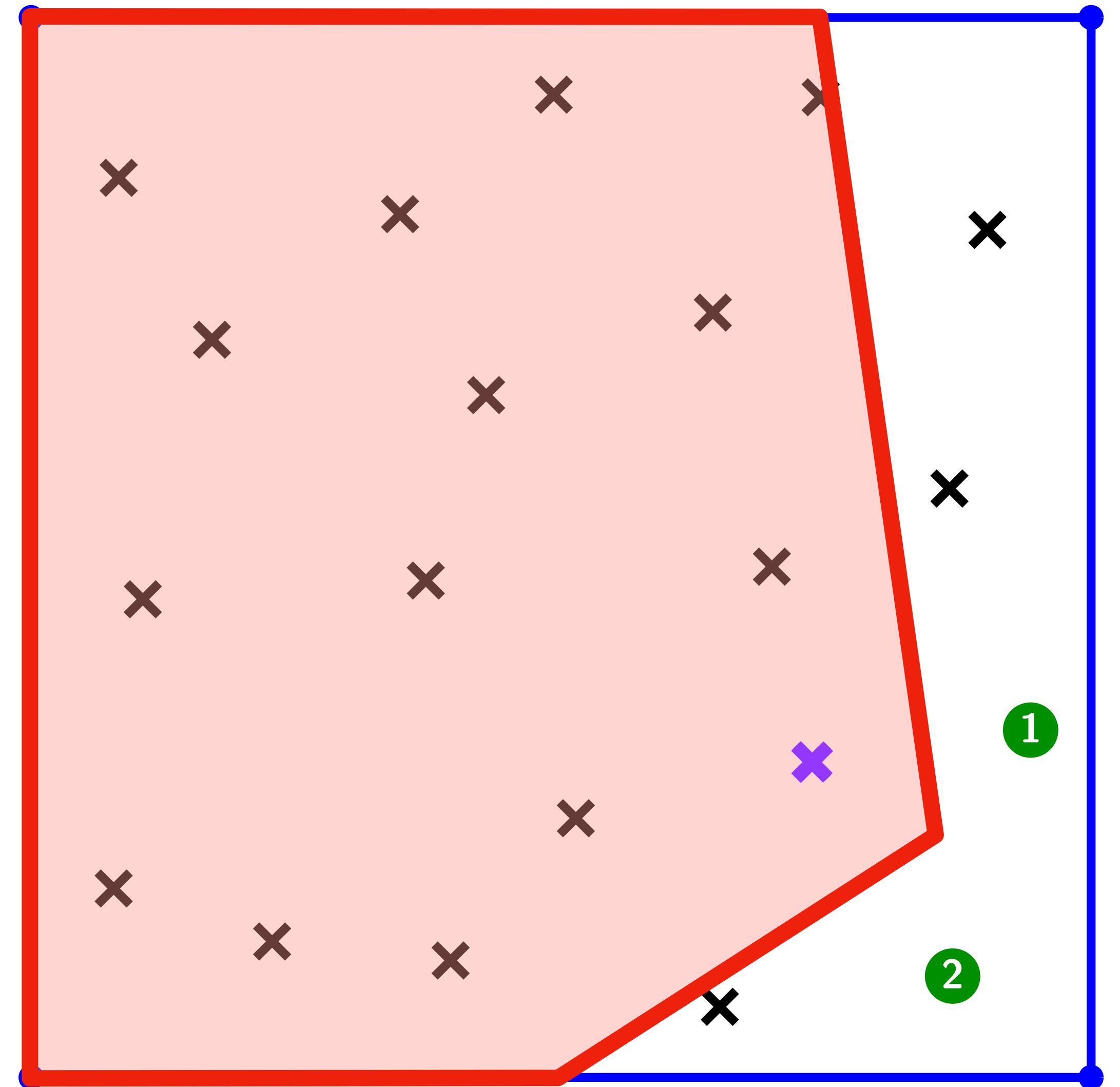
Calculating an individual Voronoi cell

- An alternative method is to compute each Voronoi cell separately
- Initialize Voronoi cell to fill Ω
- Each neighboring particle cuts the Voronoi cell by a plane that is a perpendicular bisector
- After enough nearby particles are considered, the Voronoi cell is complete



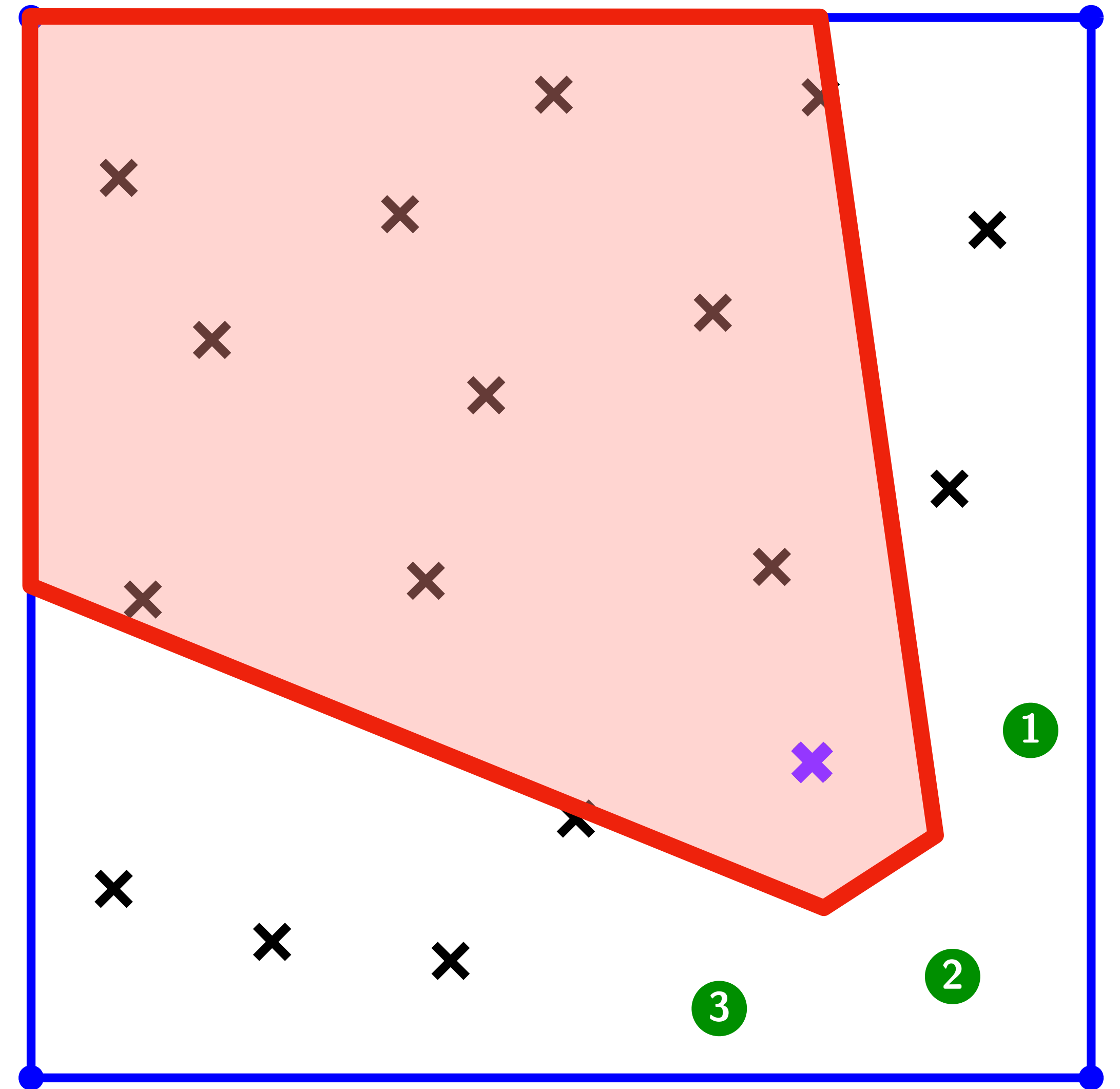
Calculating an individual Voronoi cell

- An alternative method is to compute each Voronoi cell separately
- Initialize Voronoi cell to fill Ω
- Each neighboring particle cuts the Voronoi cell by a plane that is a perpendicular bisector
- After enough nearby particles are considered, the Voronoi cell is complete



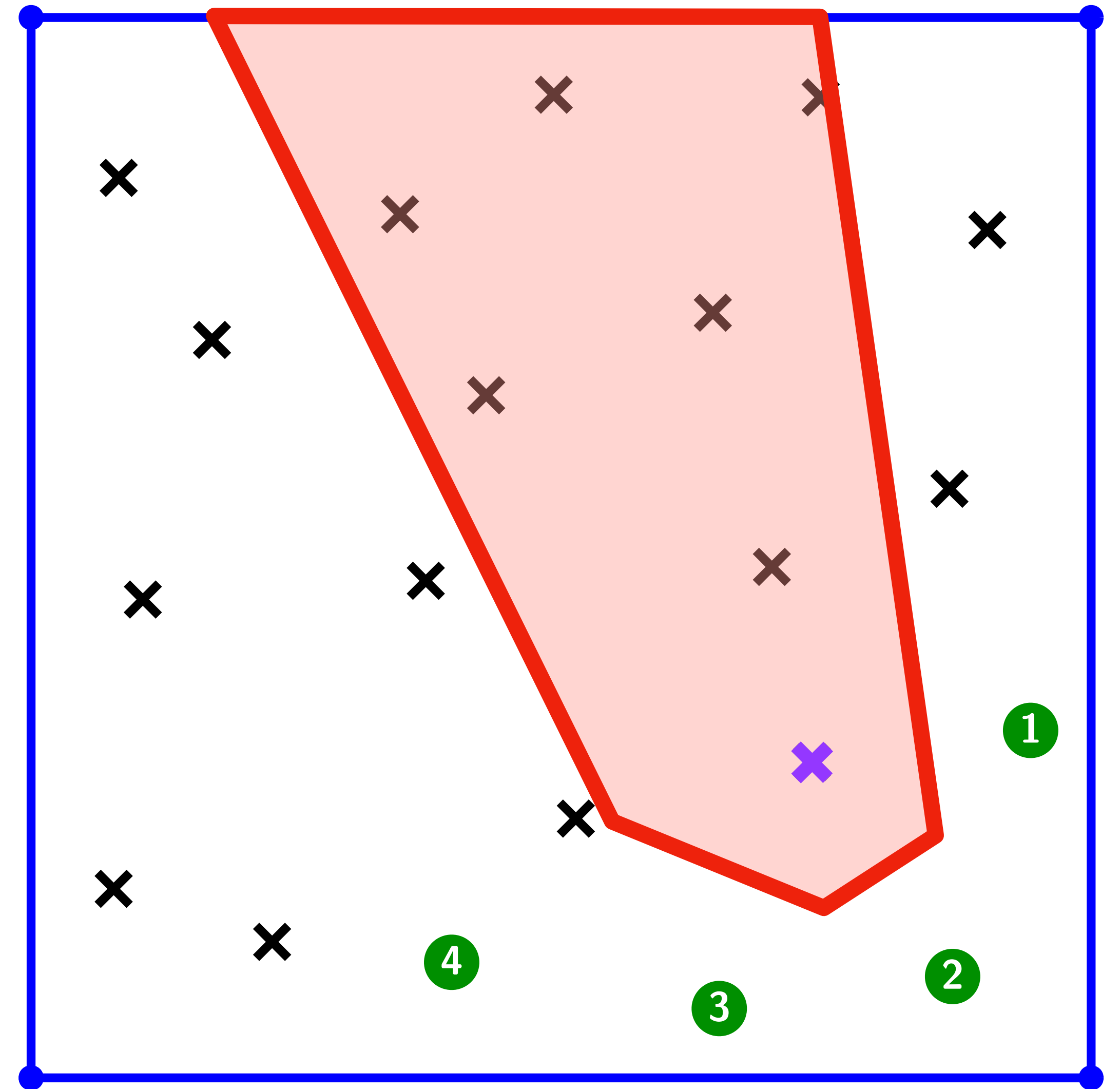
Calculating an individual Voronoi cell

- An alternative method is to compute each Voronoi cell separately
- Initialize Voronoi cell to fill Ω
- Each neighboring particle cuts the Voronoi cell by a plane that is a perpendicular bisector
- After enough nearby particles are considered, the Voronoi cell is complete



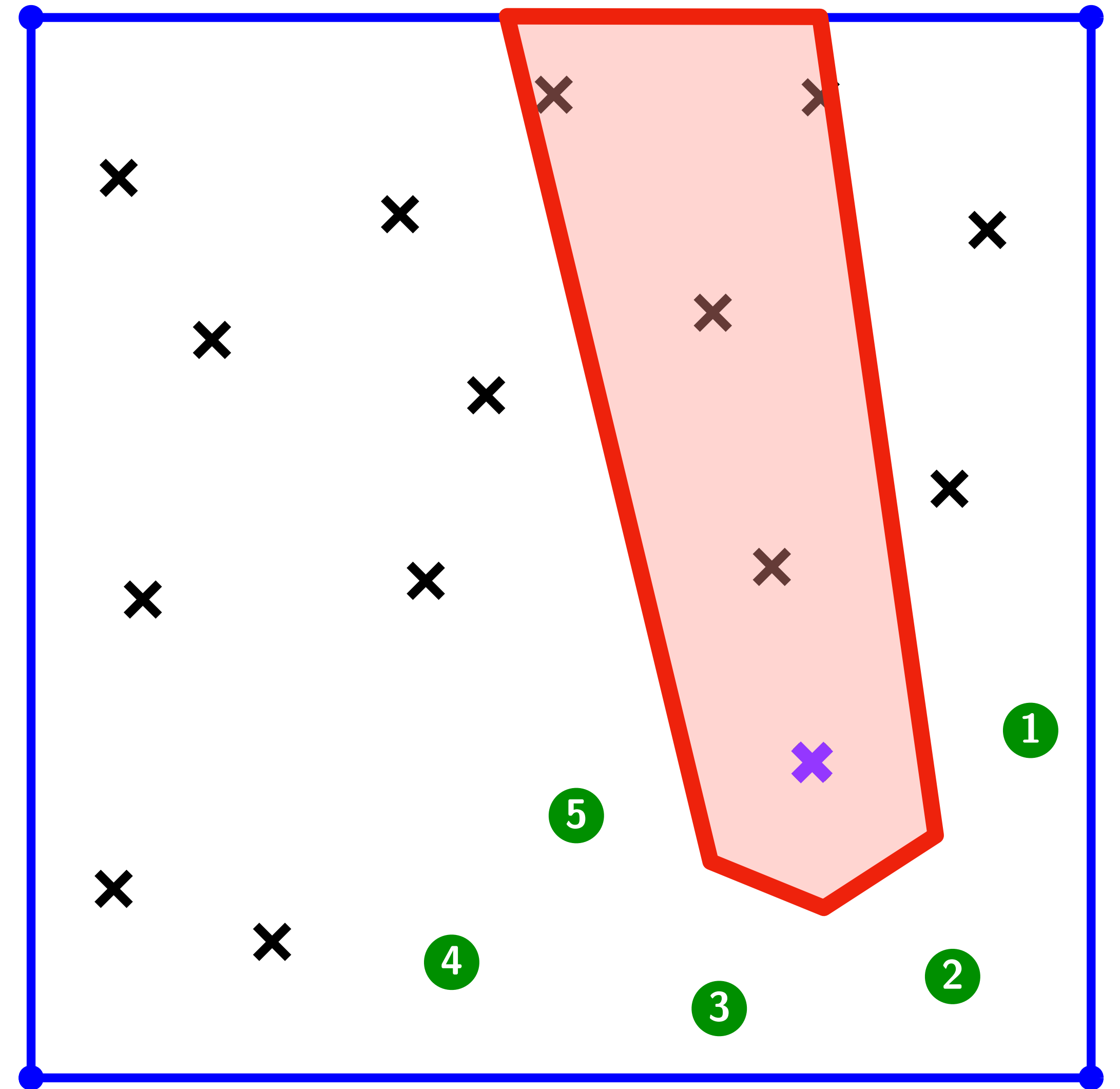
Calculating an individual Voronoi cell

- An alternative method is to compute each Voronoi cell separately
- Initialize Voronoi cell to fill Ω
- Each neighboring particle cuts the Voronoi cell by a plane that is a perpendicular bisector
- After enough nearby particles are considered, the Voronoi cell is complete



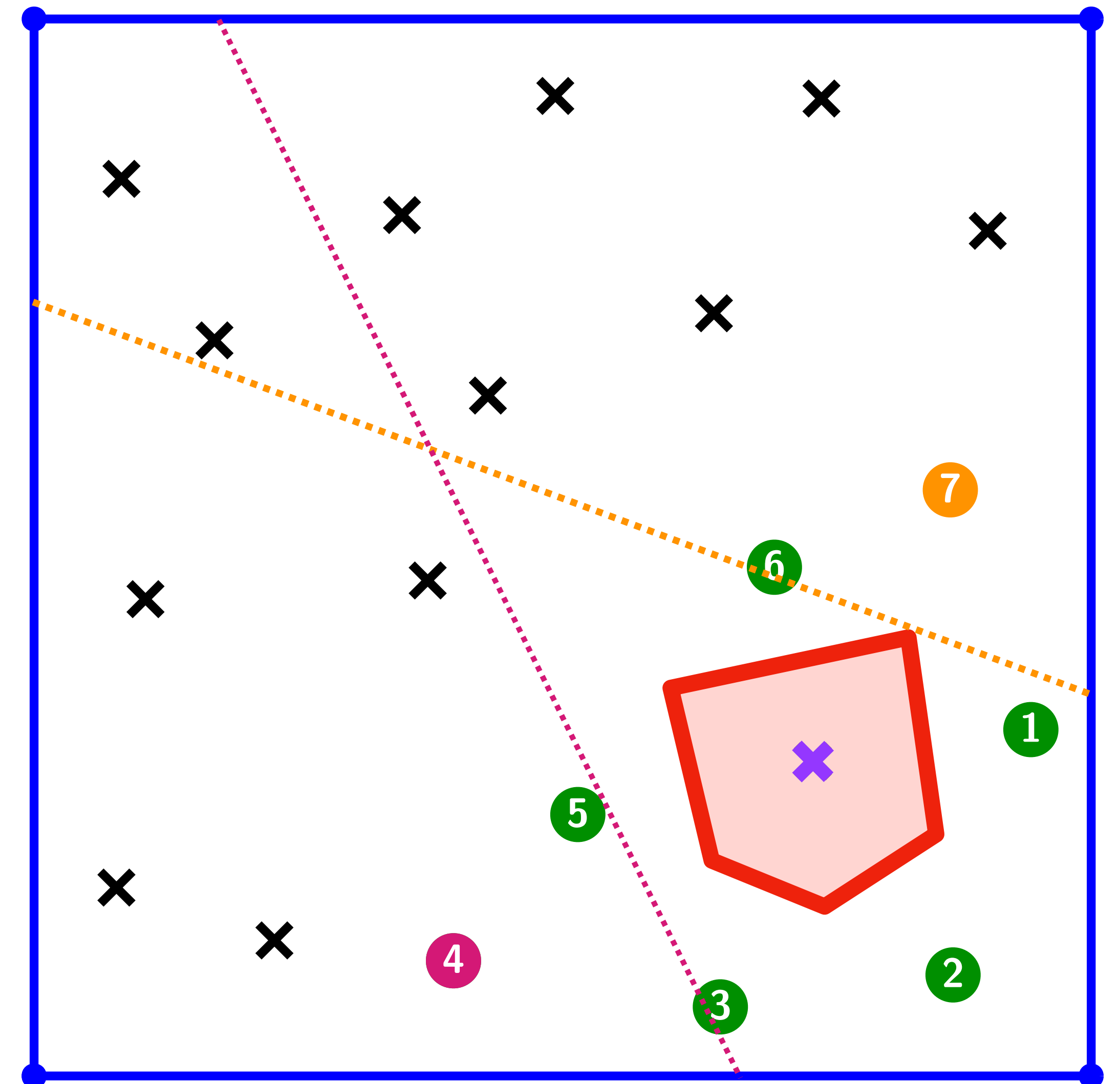
Calculating an individual Voronoi cell

- An alternative method is to compute each Voronoi cell separately
- Initialize Voronoi cell to fill Ω
- Each neighboring particle cuts the Voronoi cell by a plane that is a perpendicular bisector
- After enough nearby particles are considered, the Voronoi cell is complete



Calculating an individual Voronoi cell

- An alternative method is to compute each Voronoi cell separately
- Initialize Voronoi cell to fill Ω
- Each neighboring particle cuts the Voronoi cell by a plane that is a perpendicular bisector
- After enough nearby particles are considered, the Voronoi cell is complete



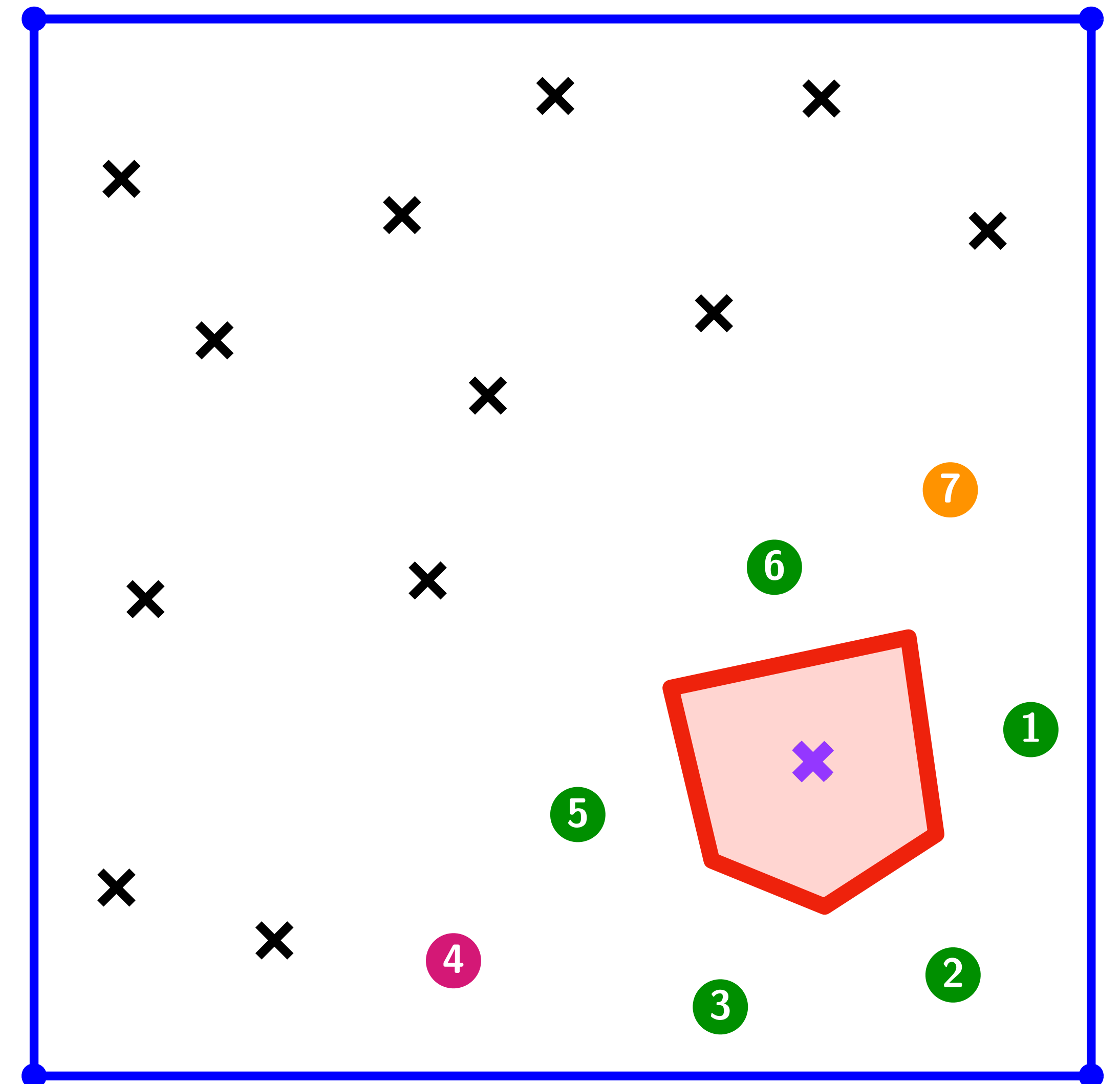
7 : considered, but didn't cut cell

4 : cut cell, but ultimately not needed

1,2,3,5,6 : required

Computational considerations

- Each cell can be computed independently; has advantages for parallel computation
- The algorithm needs to consider nearby particles efficiently—we want to find the **green** particles, and minimize choosing **orange/magenta** particles
- The algorithm has some drawbacks—we will return to this later



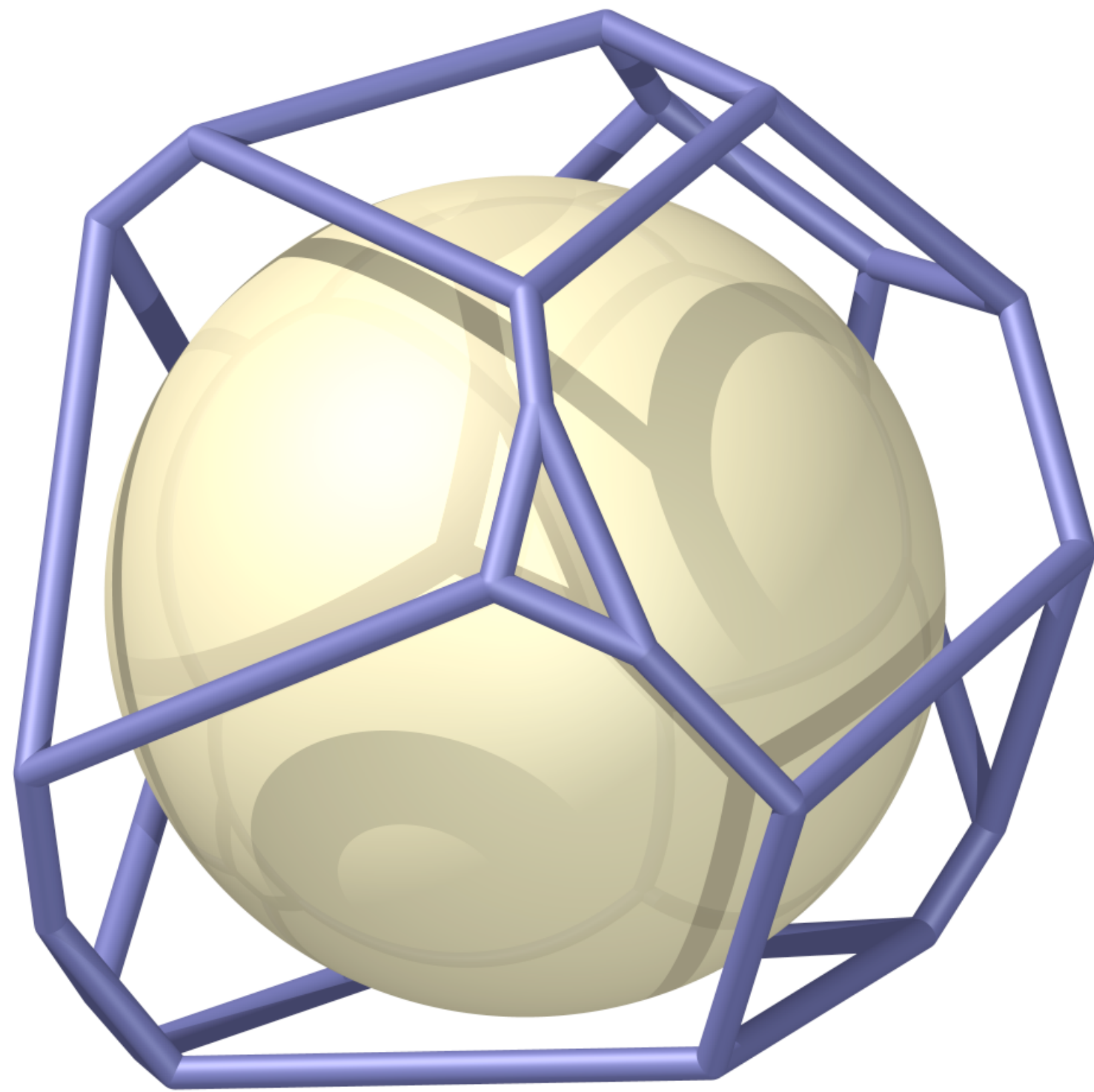
7 : considered, but didn't cut cell

4 : cut cell, but ultimately not needed

1,2,3,5,6 : required

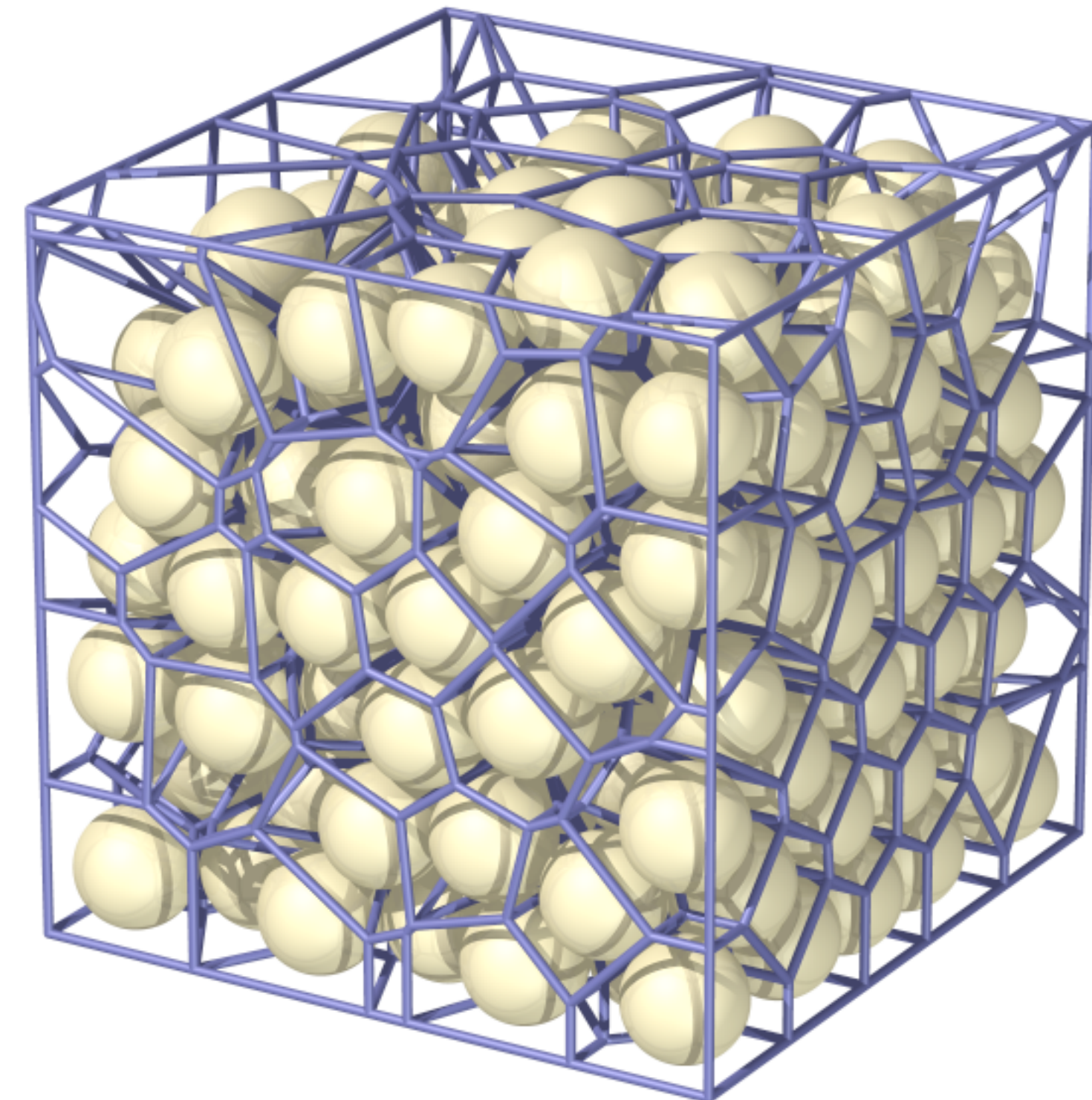
Voro++, a software library for cell-based Voronoi calculations

- Primary audience: diagnostics for particle-based simulation
- Good performance (135,000 cells per second on a typical desktop, 3x faster than Qhull)
- Written in C++ around several *classes*



voronoicell_3d class

(Code module for doing single-cell computations)

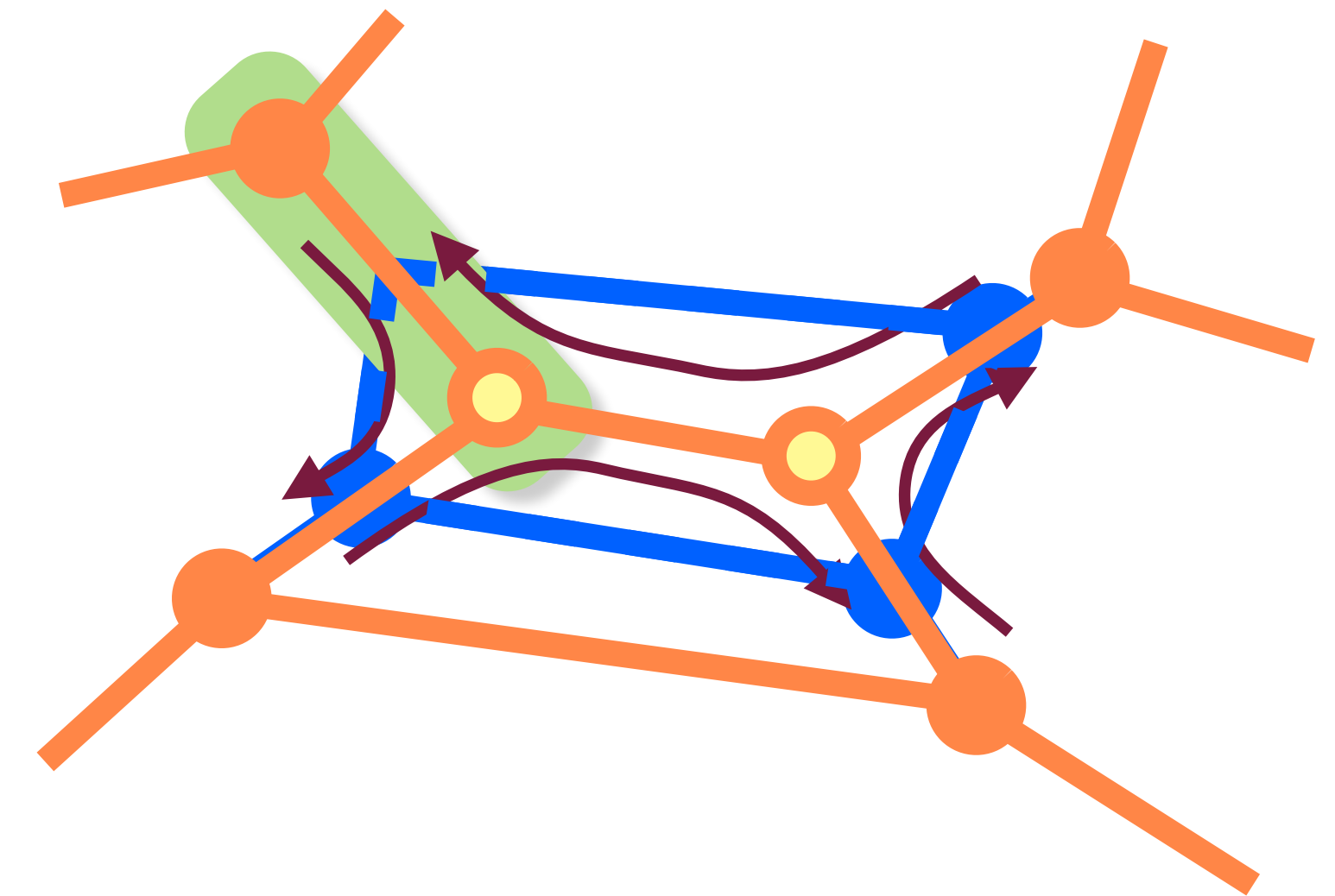
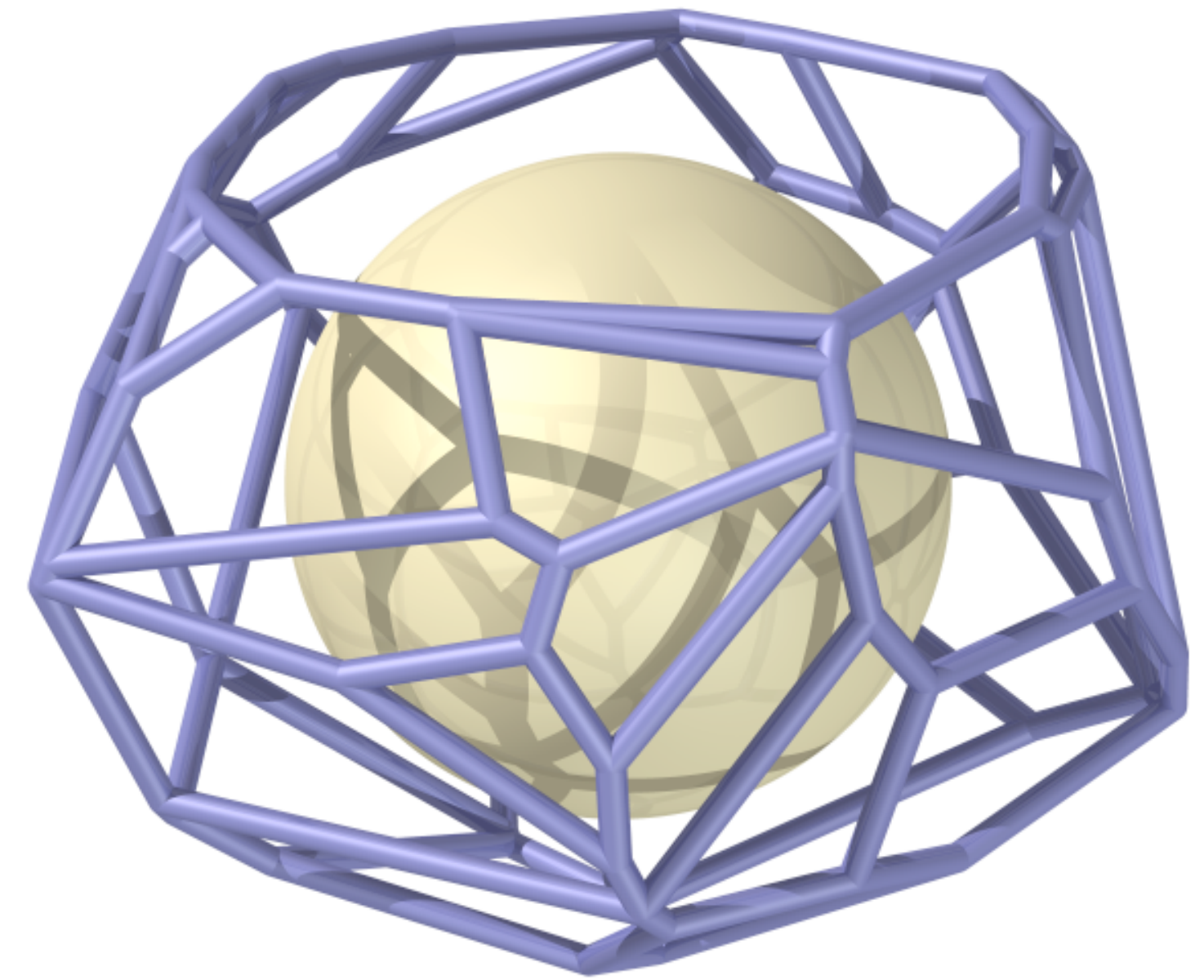


container_3d class

(Code module for Voronoi analysis of particle arrangements in a domain)

The voronoicell_3d class

- An irregular convex polyhedron described by:
 - Vertex positions
 - Table of edges
- Primary operation: recompute vertices when cut by a plane
- Diagnostic routines:
 - Number of planes, edges
 - Volume, centroid, *etc.*

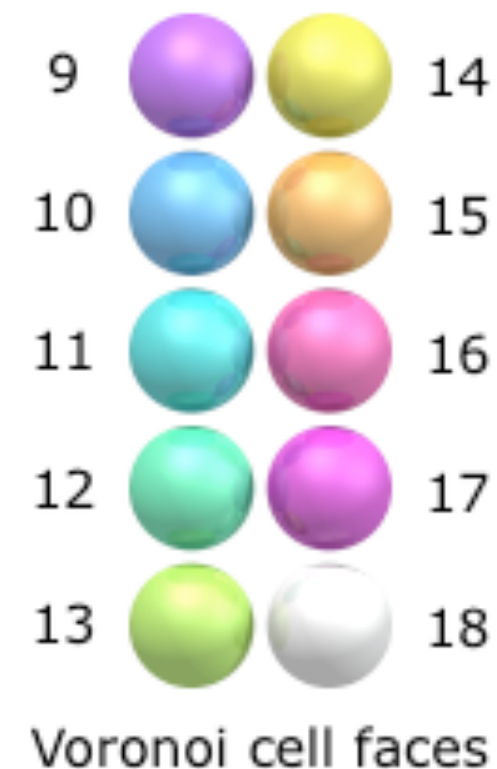
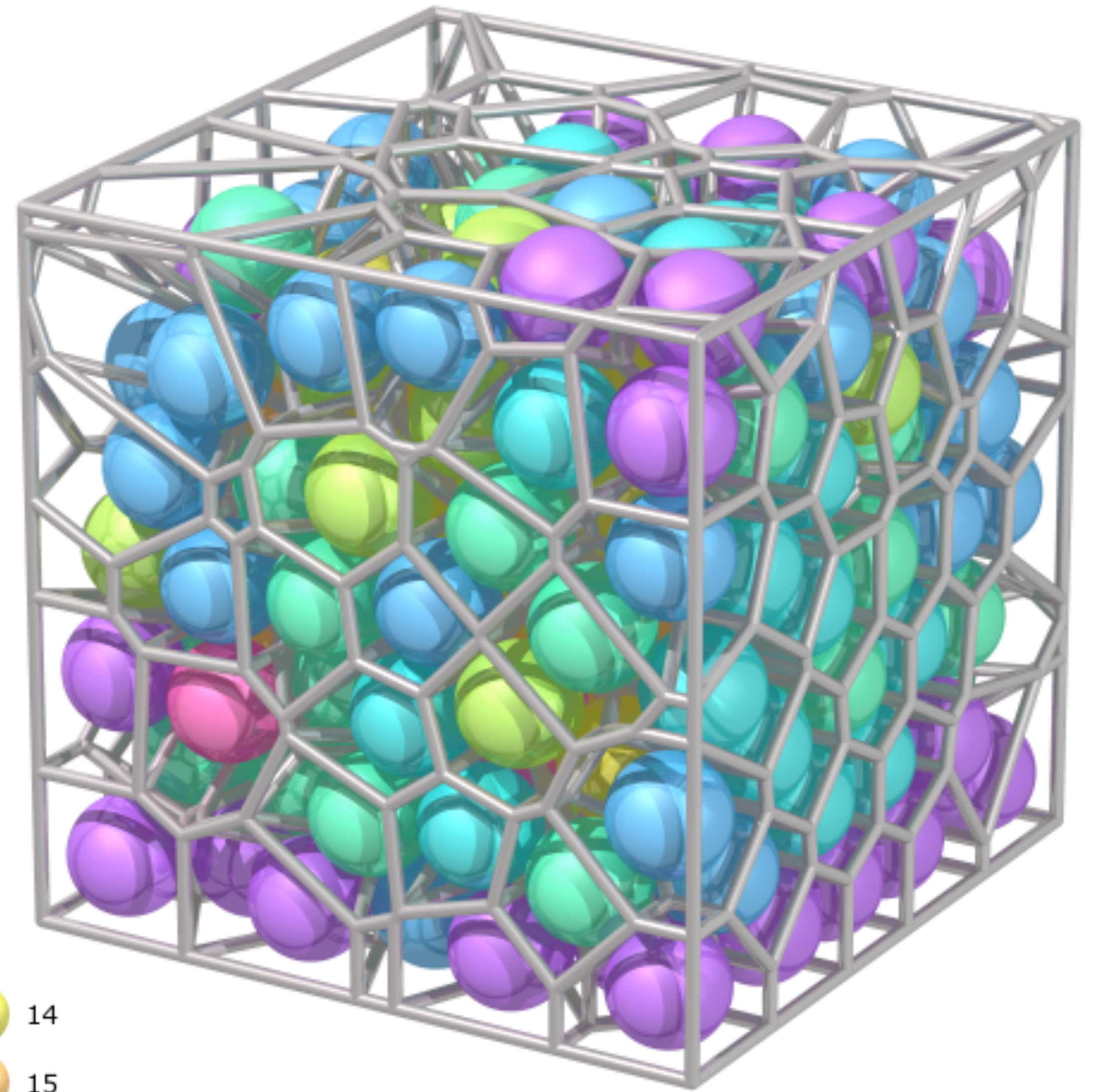


● : inside

● : outside

The container_3d class

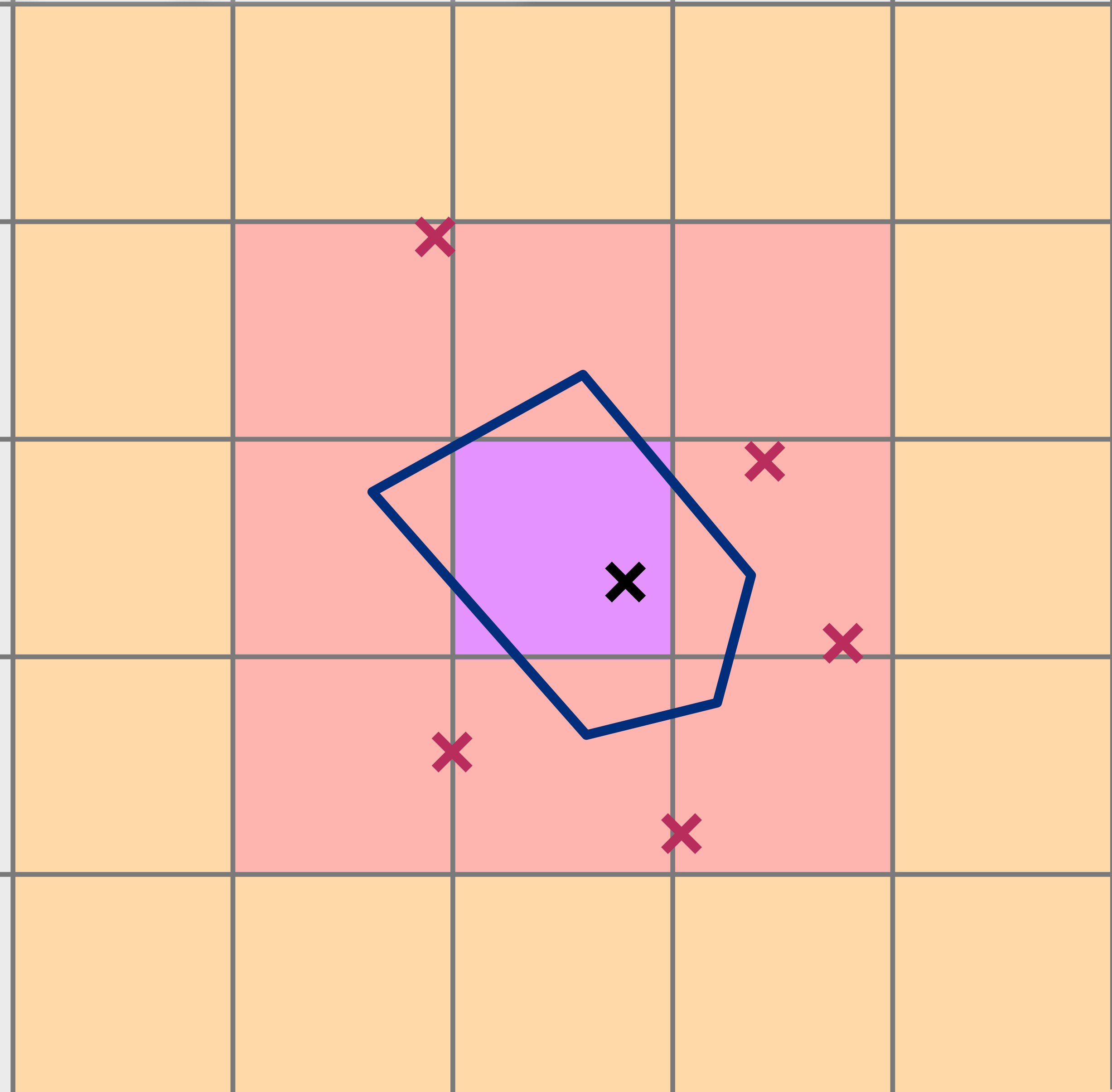
- Sorts particles into rectangular grid of blocks
- Uses `voronoicell` class to construct individual cells by testing over the blocks
- Can carry out a variety of calculations on the computed cells
- Library and example programs available for download from <http://math.lbl.gov/voro++/>



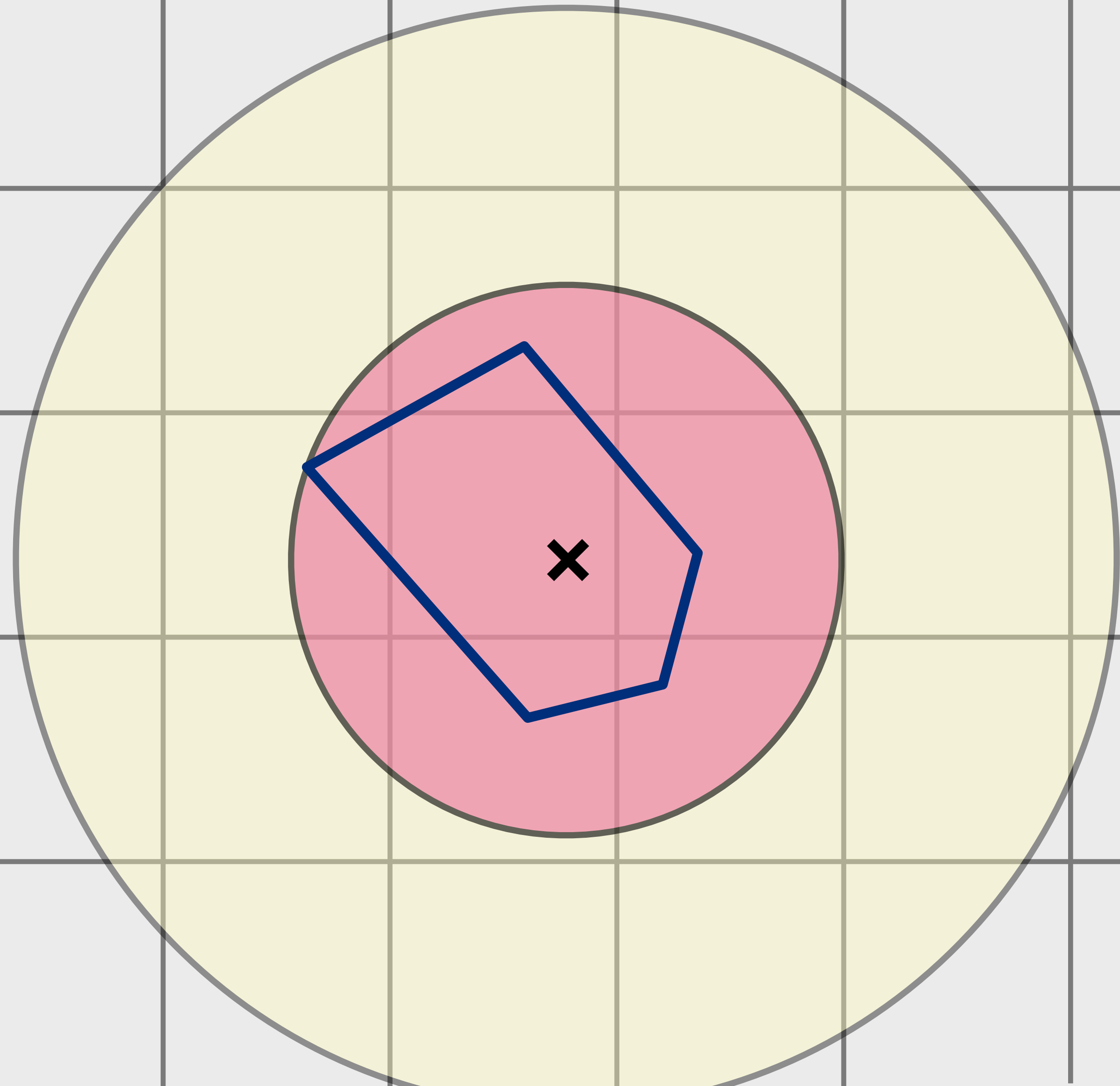
Voronoi tessellation with
216 particles in a cube of side length 6,
generated by DEVSIM simulation
the Voronoi cell faces

A hybrid searching algorithm

Basic approach: sweep outwards and apply plane cuts for particles in those blocks



A hybrid searching algorithm



Sphere bounding: only need to test blocks up to twice the maximum distance to a vertex

A hybrid searching algorithm

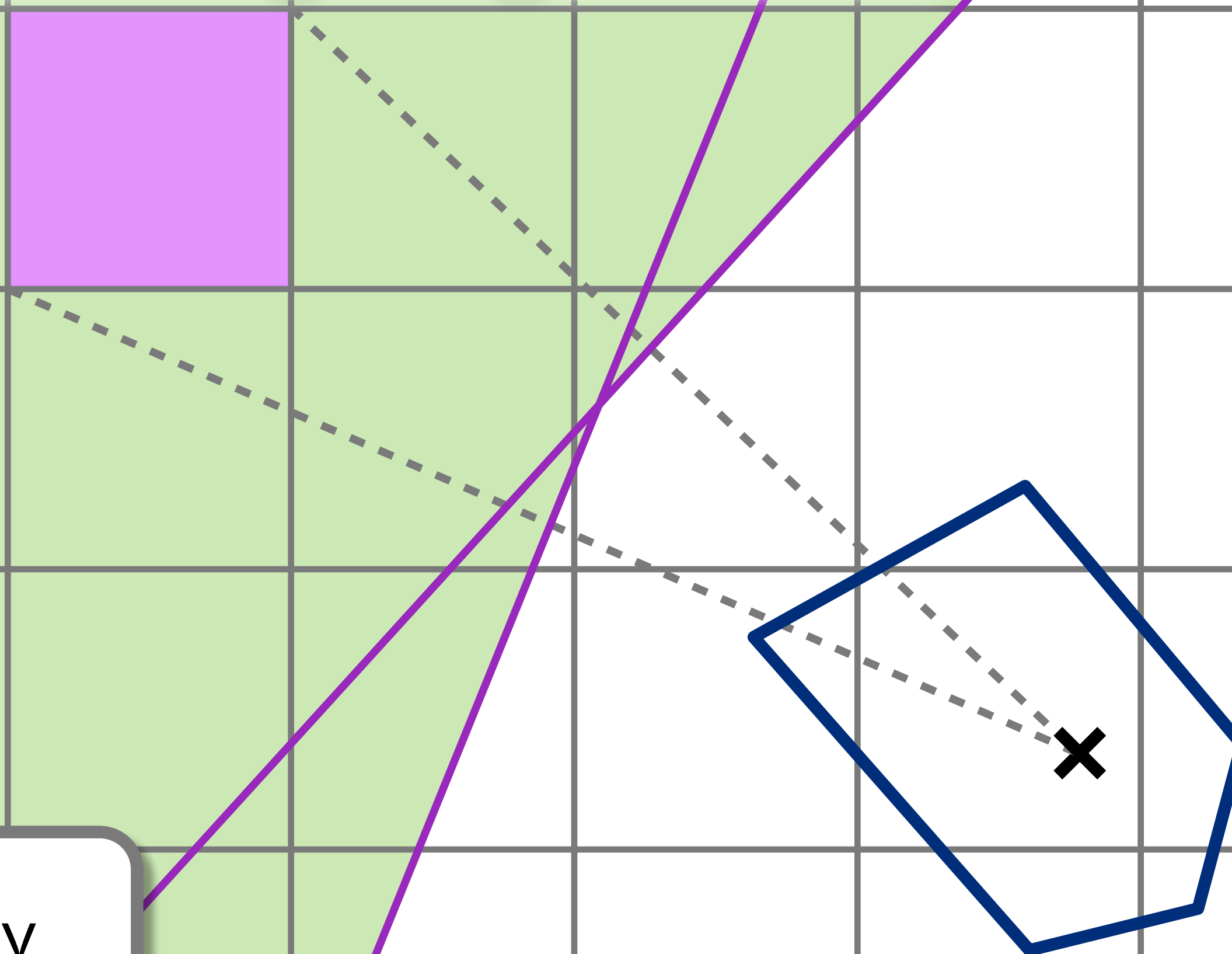


Advantage: very fast

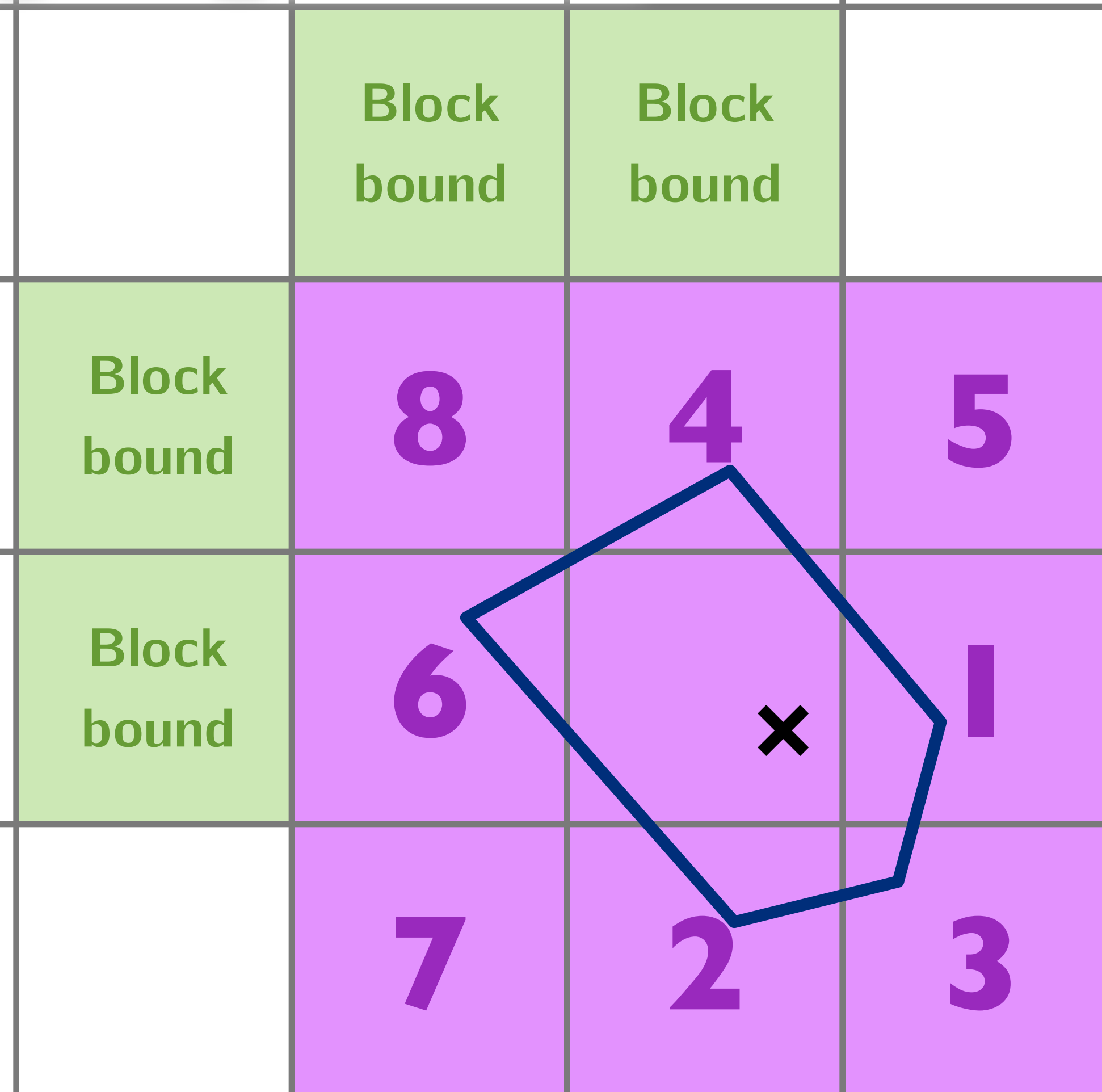
Disadvantage: works poorly
for cells that extend far in one
direction

A hybrid searching algorithm

Block bounding: apply plane tests to ascertain if a block could possibly cut the cell



A hybrid searching algorithm

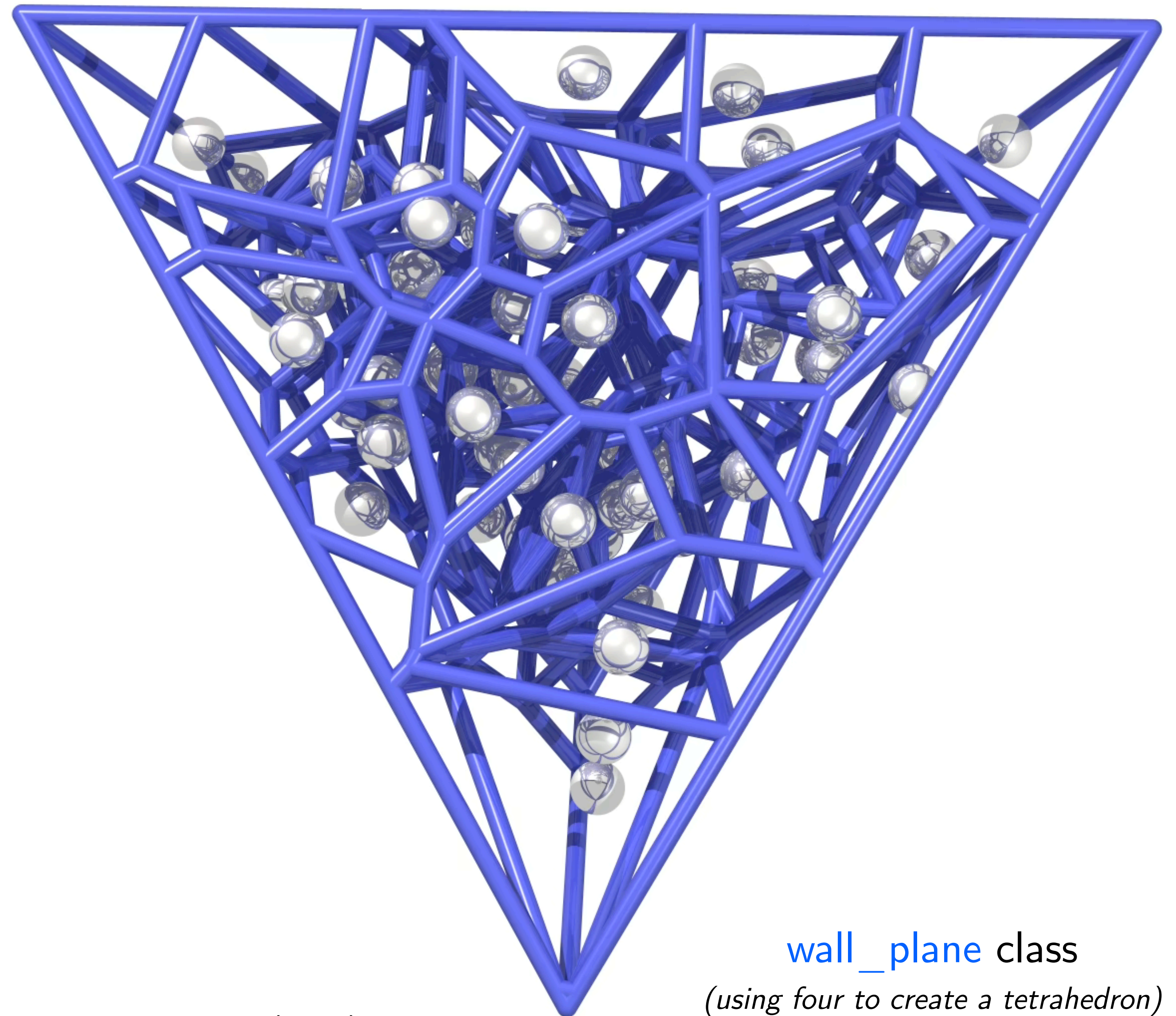


Hybrid approach:

1. Apply a precomputed list of nearby blocks, using sphere bounding
2. Flood fill outwards using block bounding

Classes for wall computations

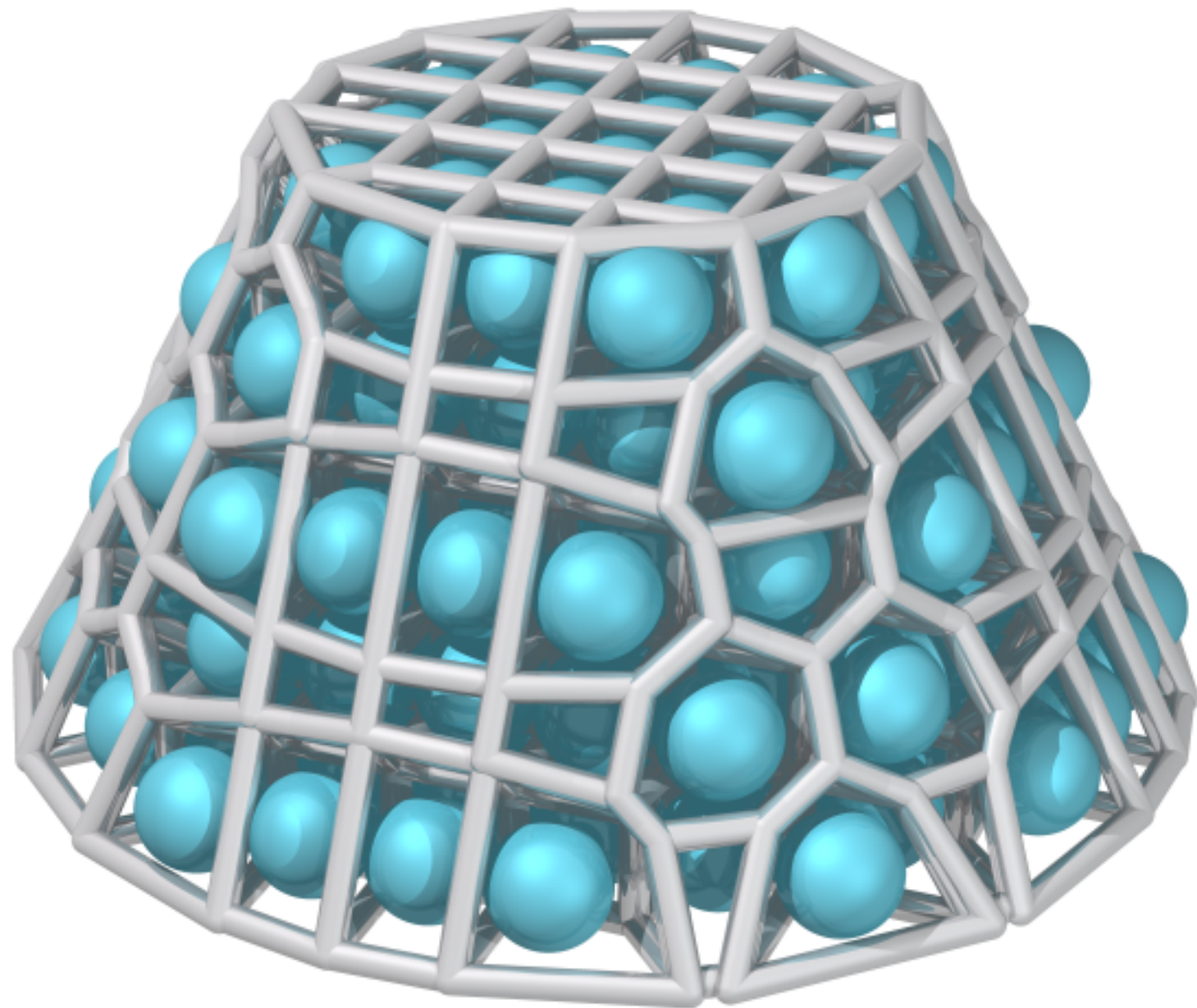
- Wall classes can be added to the container class
- They apply additional plane cuts during cell construction
- Gives perfect results within convex polygonal shapes



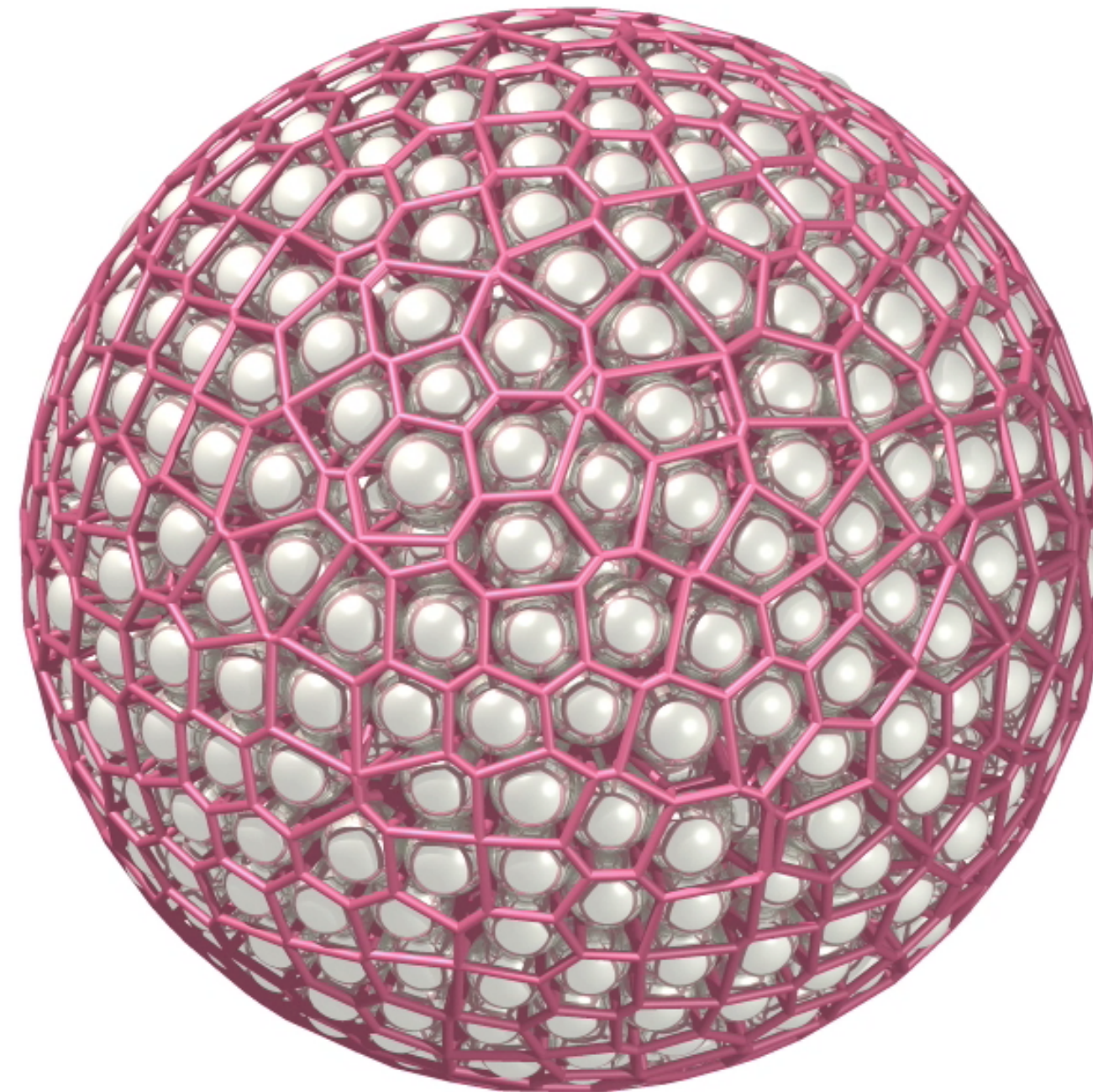
`wall_plane` class
(using four to create a tetrahedron)

Walls with curved boundaries

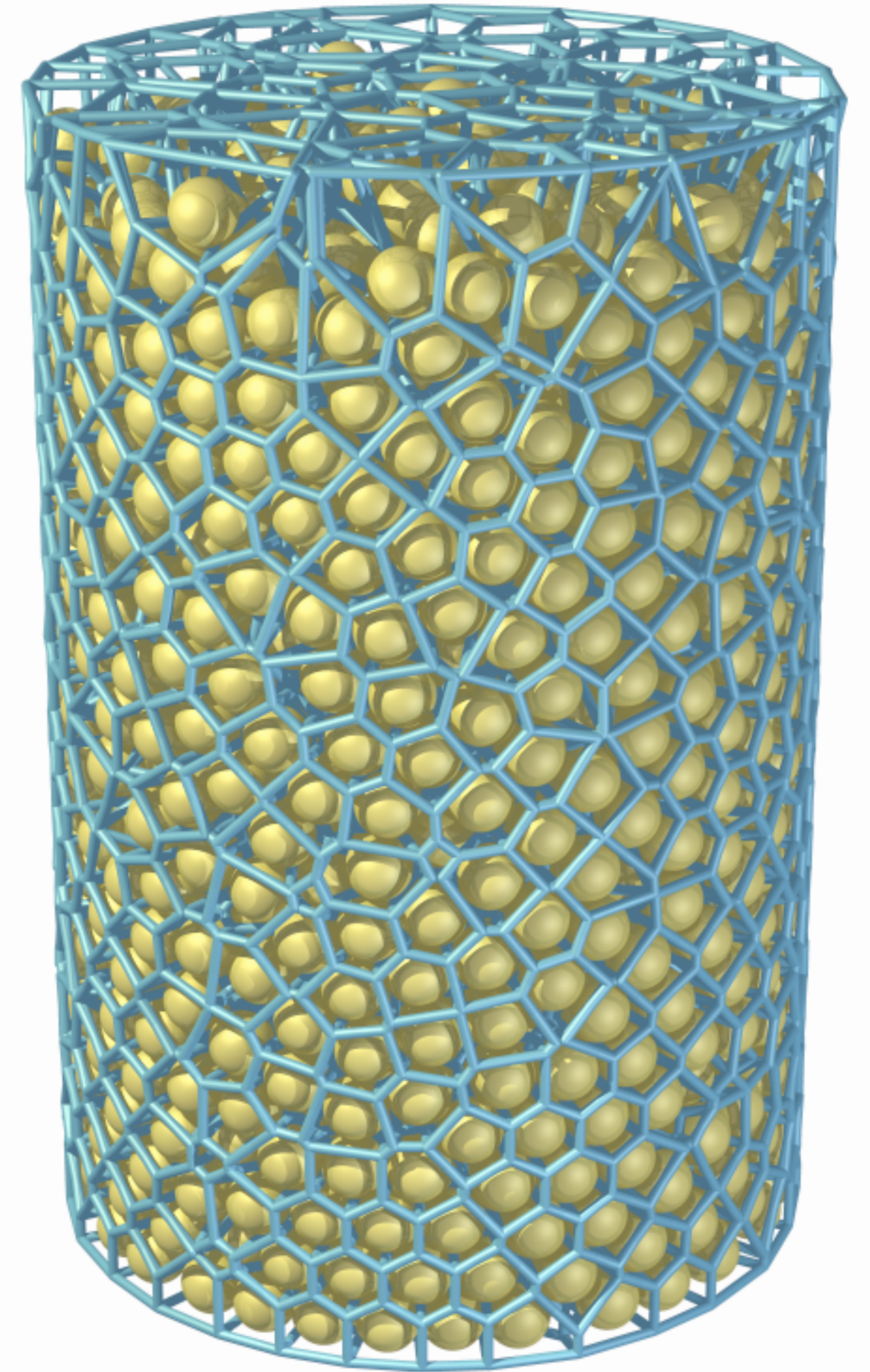
- Curved boundaries can be approximated with plane cuts
- Has small inaccuracies from approximating the curved surface



`wall_cone` class
(to make a frustum)



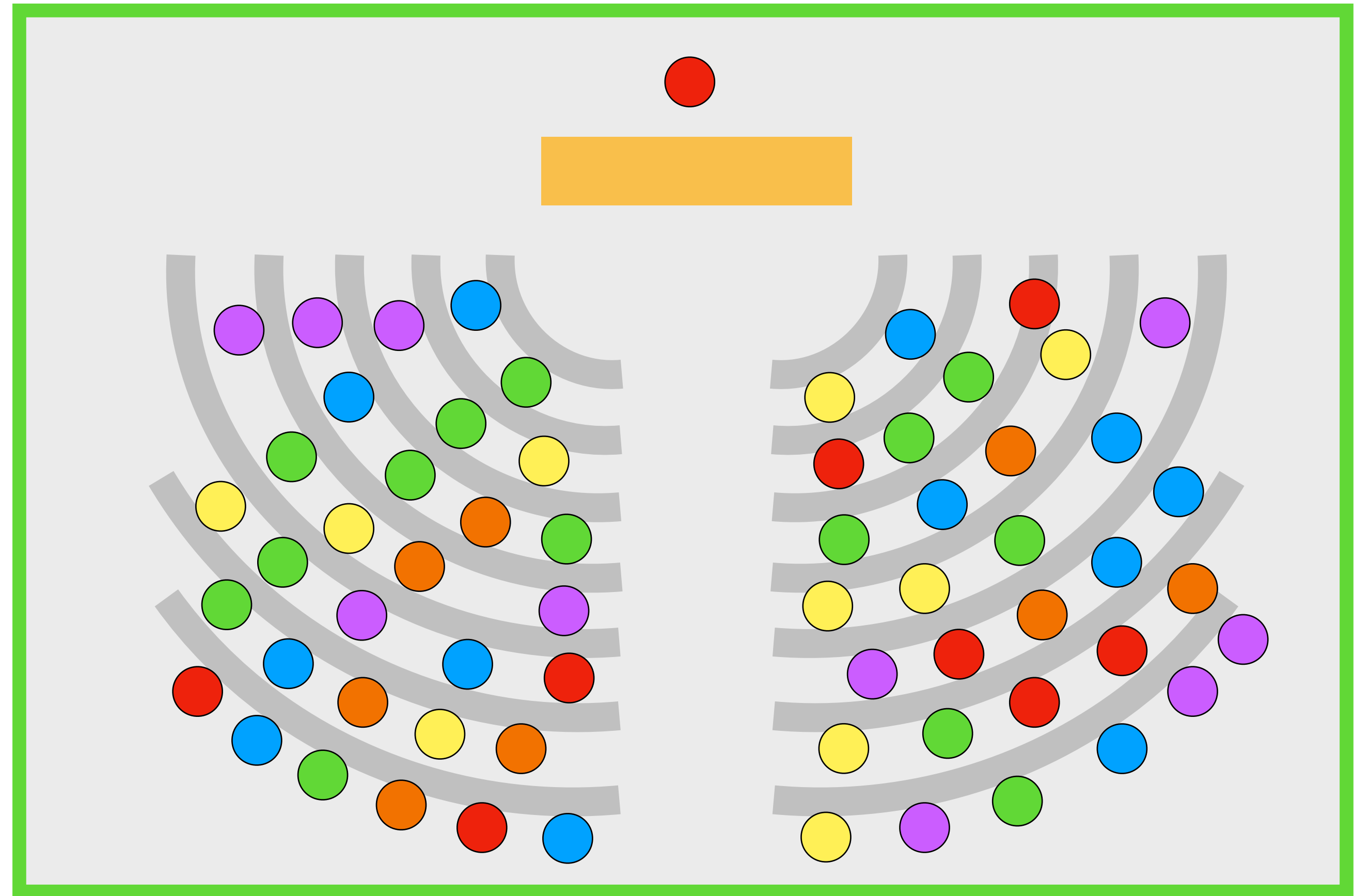
`wall_sphere` class



`wall_cylinder` class

Neighbor relations test

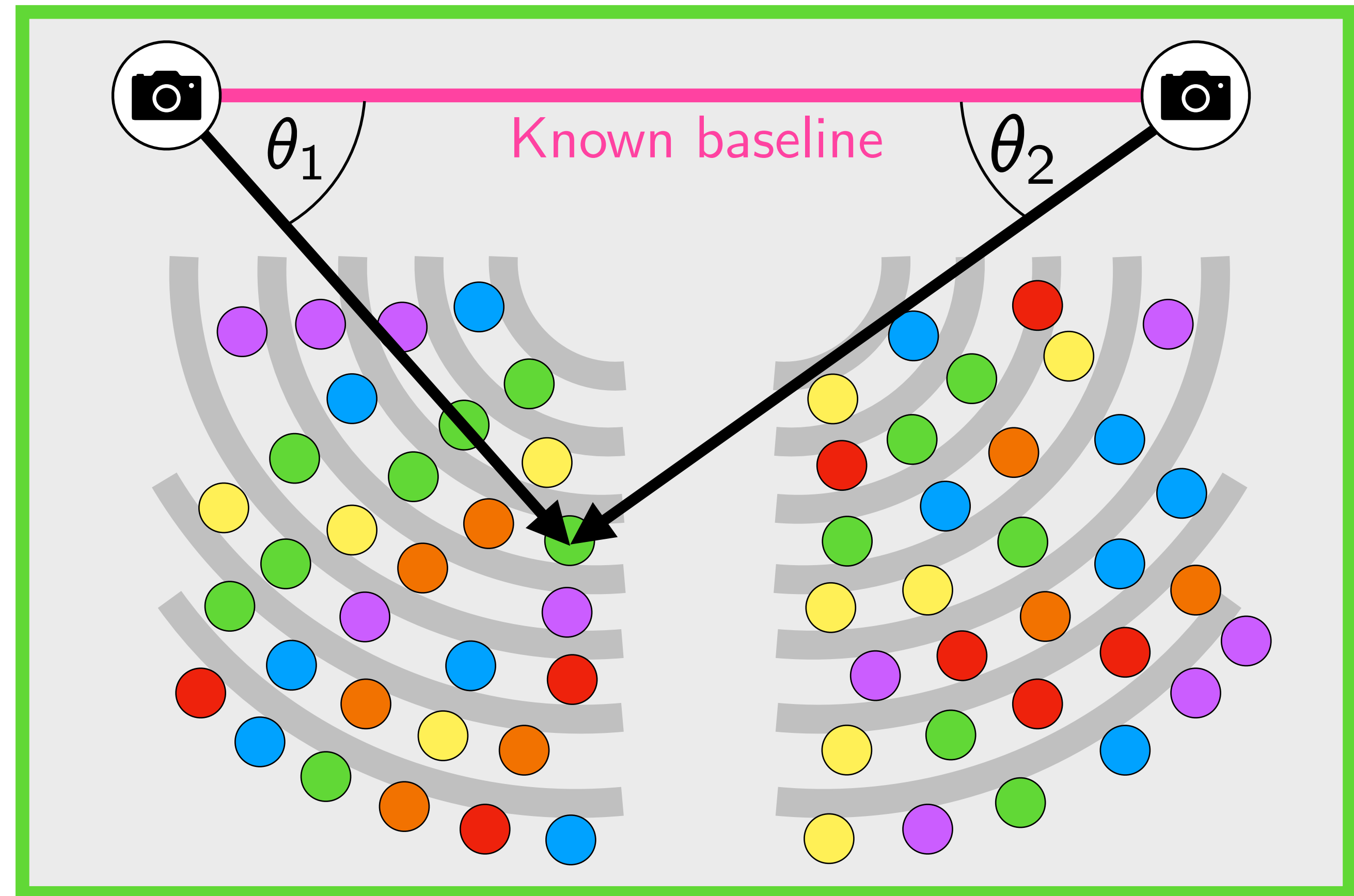
- Suppose everyone in the room is a particle
- Who would you define as your particle neighbors?
- I will give everyone a rainbow-colored ID number



The lecture room

Triangulation

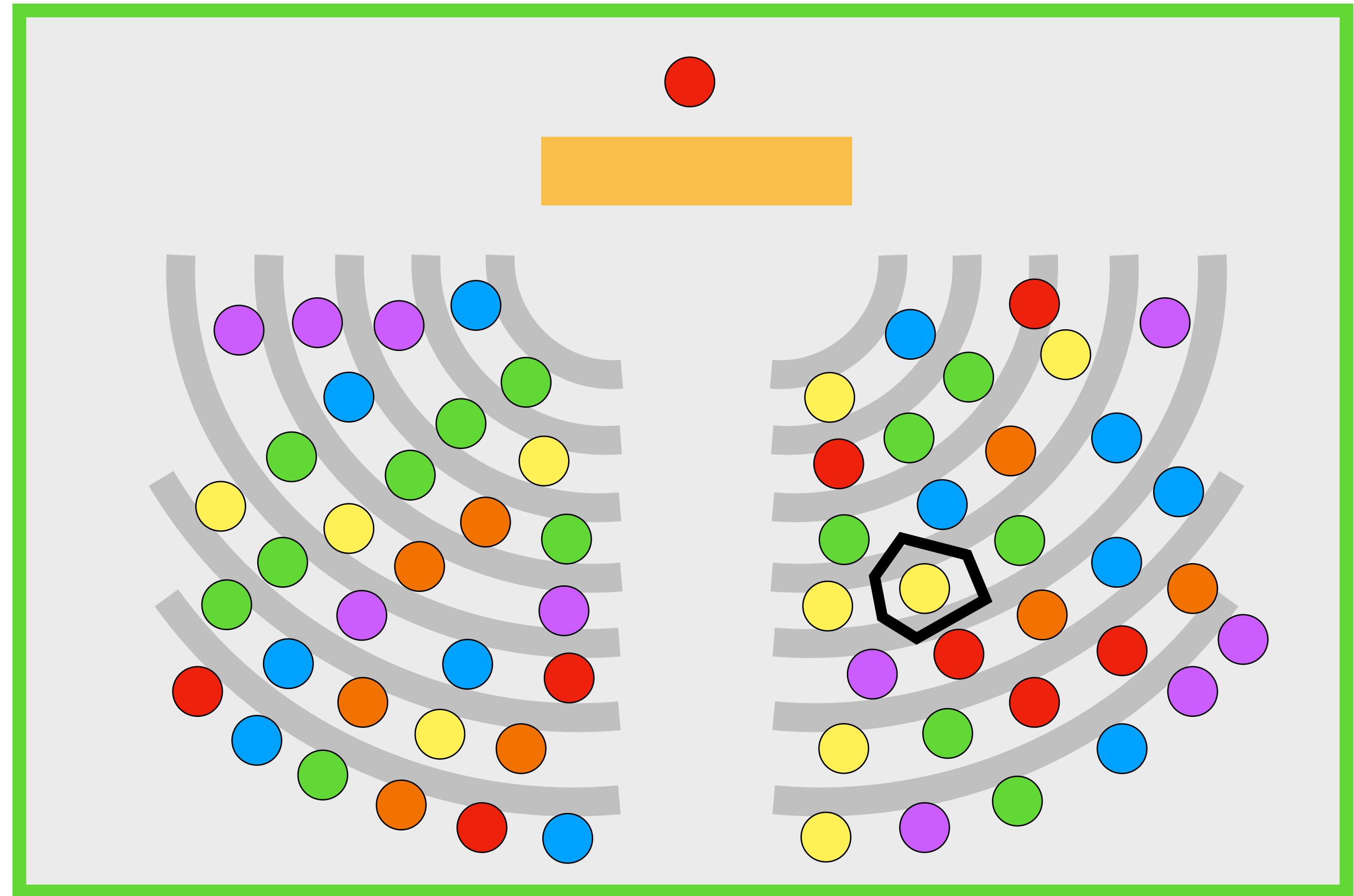
- I am going to build a map of particle positions using triangulation
- From photos at two ends of a known baseline, I can determine everyone's position
- I will build a map visualizing the neighbor relations



The lecture room

Neighbor relations test (with Voronoi cells)

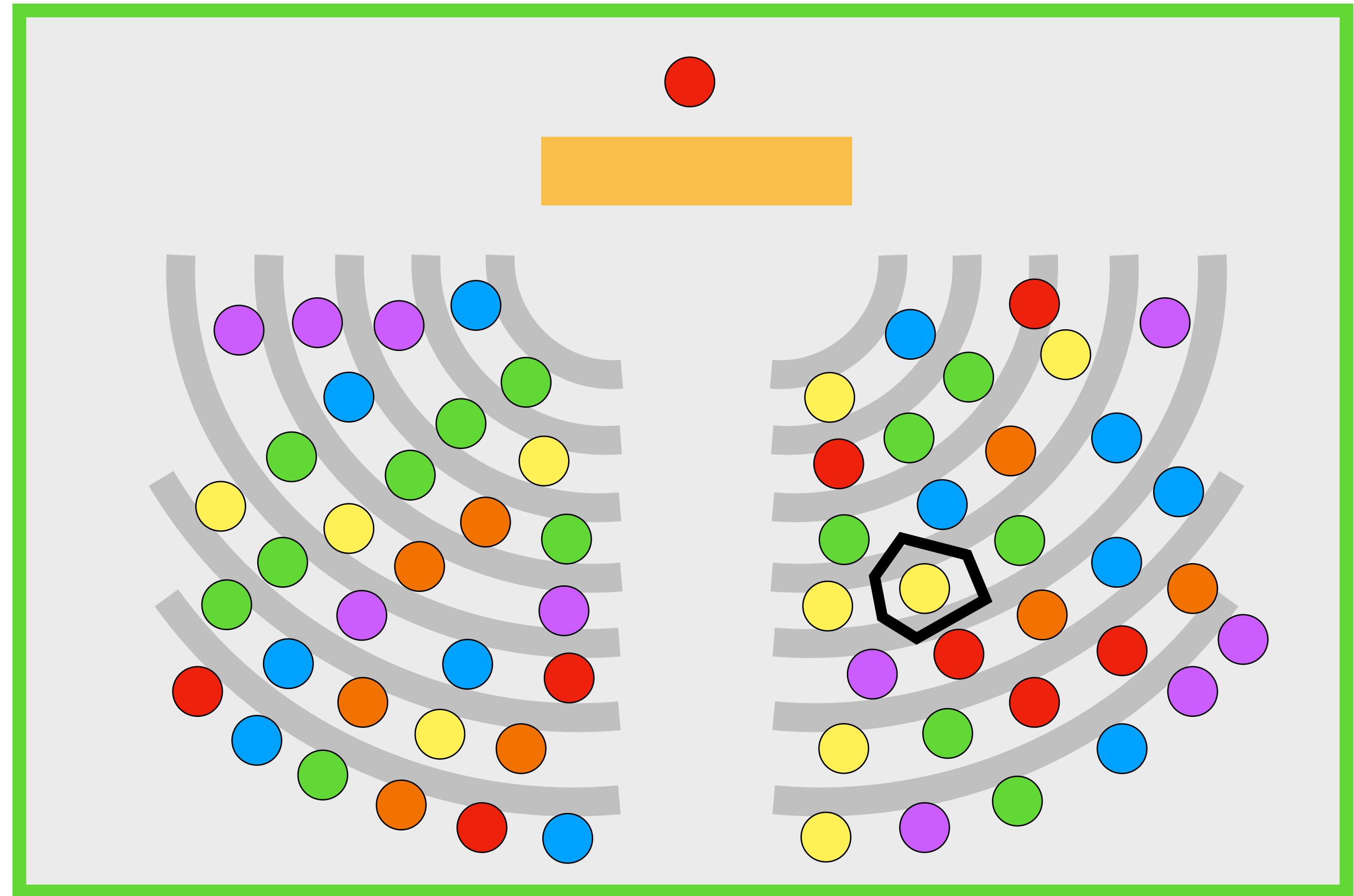
- I want you to visualize the Voronoi cell around you, and write down in **A** your neighbors that share Voronoi face



The lecture room

Neighbor relations test (with Voronoi cells)

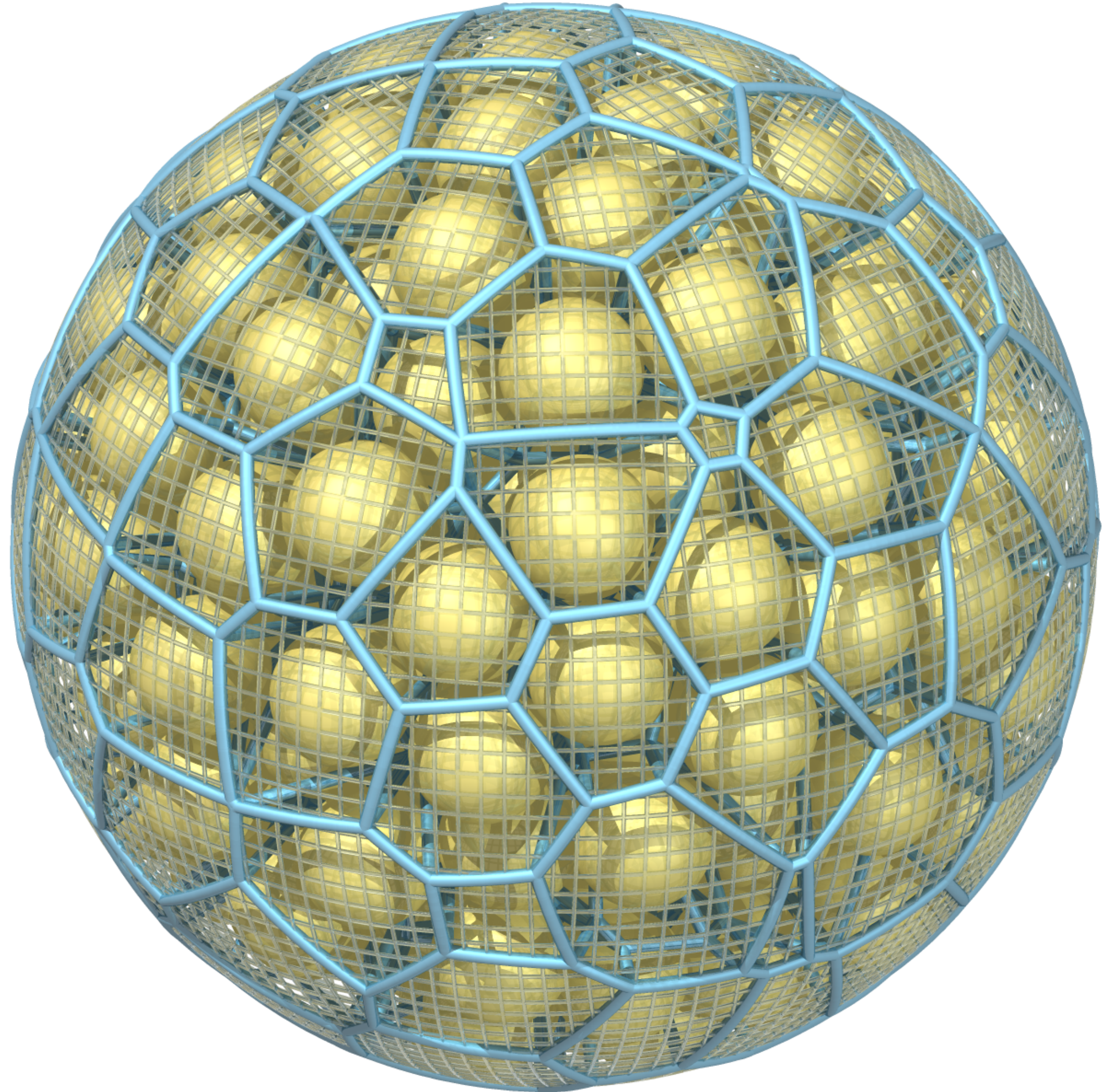
- I want you to visualize the Voronoi cell around you, and write down in **A** your neighbors that share Voronoi face
- Extra credit in **B**: estimate the area of your Voronoi cell (include the unit, e.g. m² or ft²)



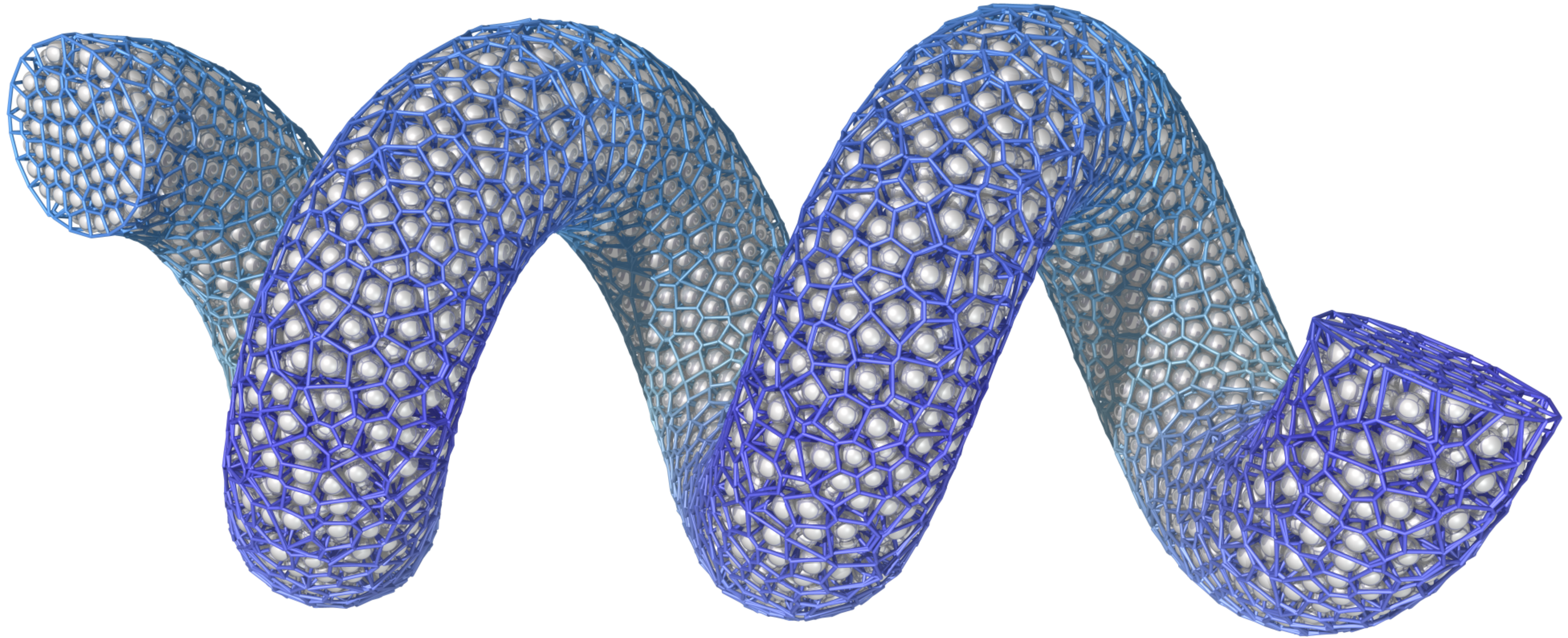
The lecture room

Higher accuracy

- Multiple plane cuts can better approximate a curved surface
- In the example shown, a 2D grid of cutting planes is applied to every Voronoi cell
- The total Voronoi cell volume differs from the exact sphere volume by 0.039%

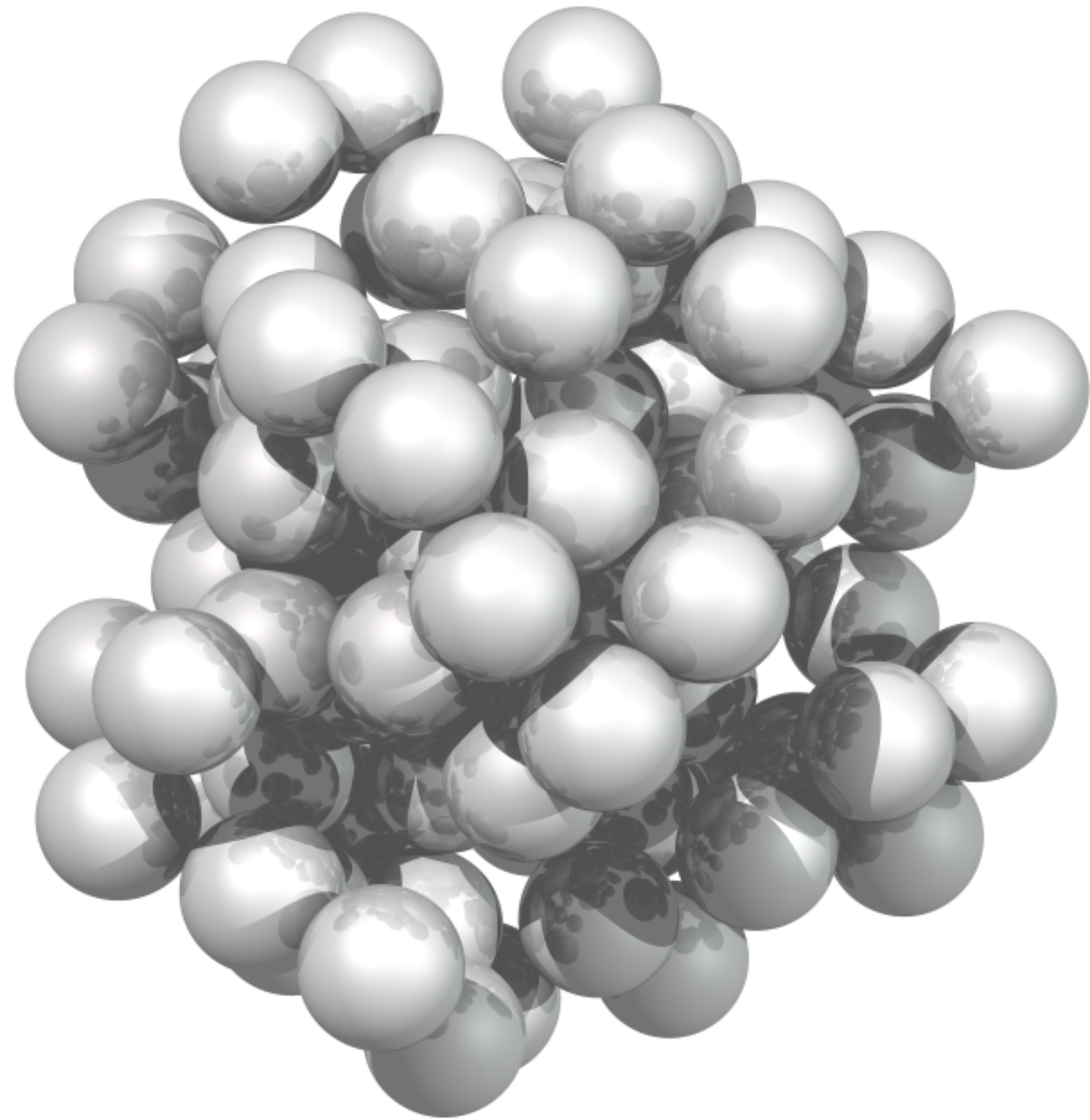


Custom wall shapes



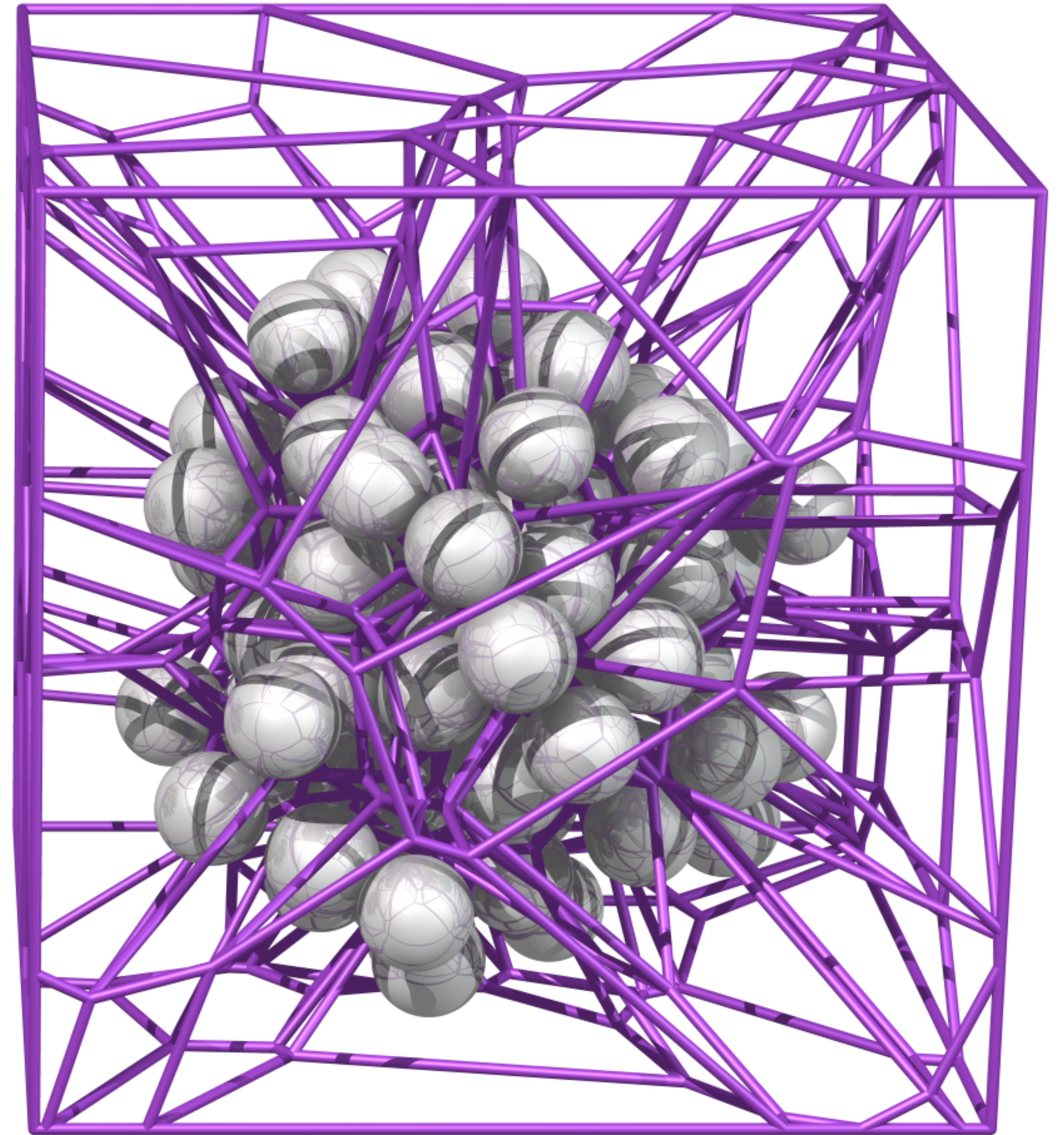
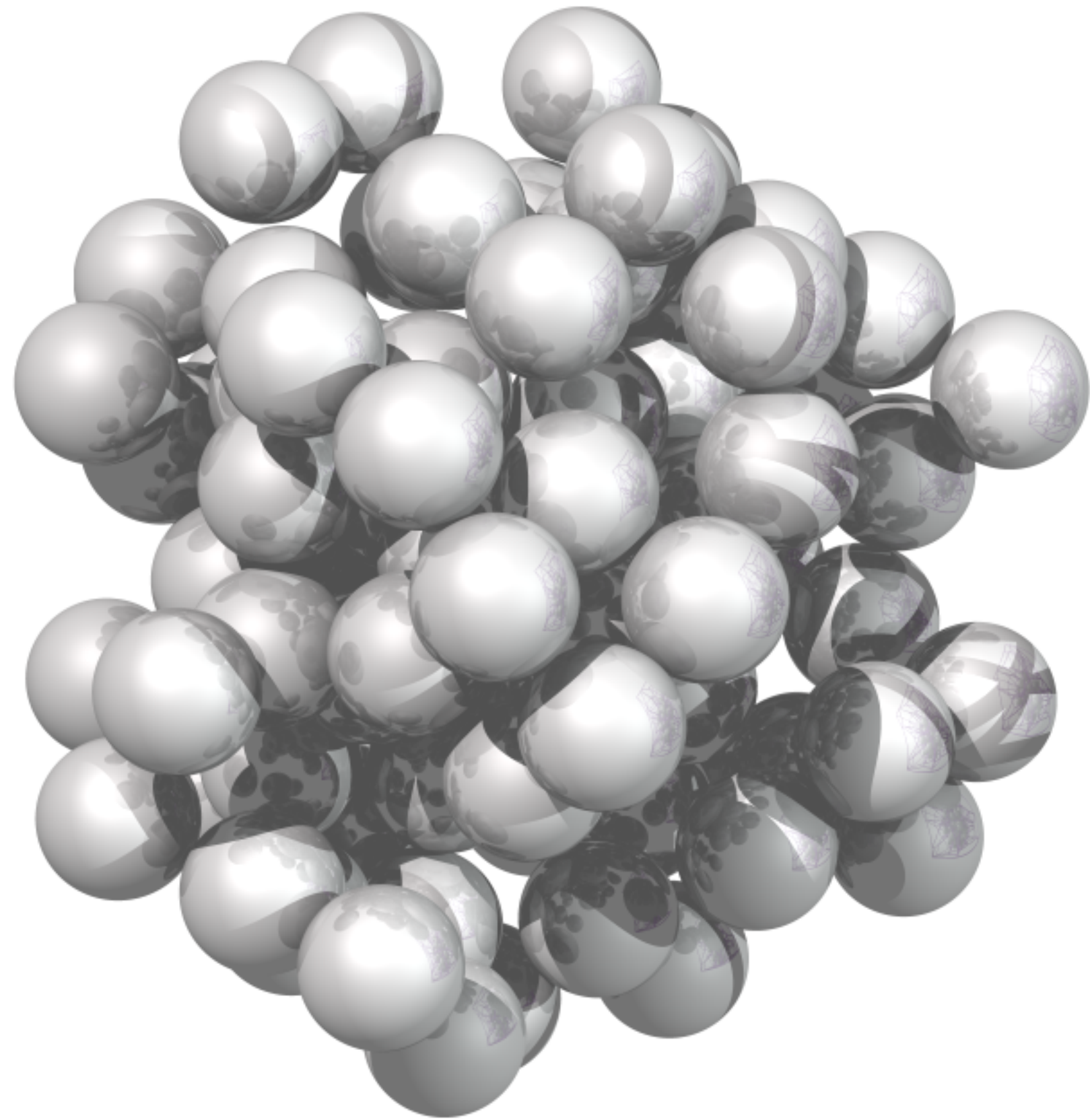
Additional wall shapes can be written by the user to handle custom domains

Irregular particle packings



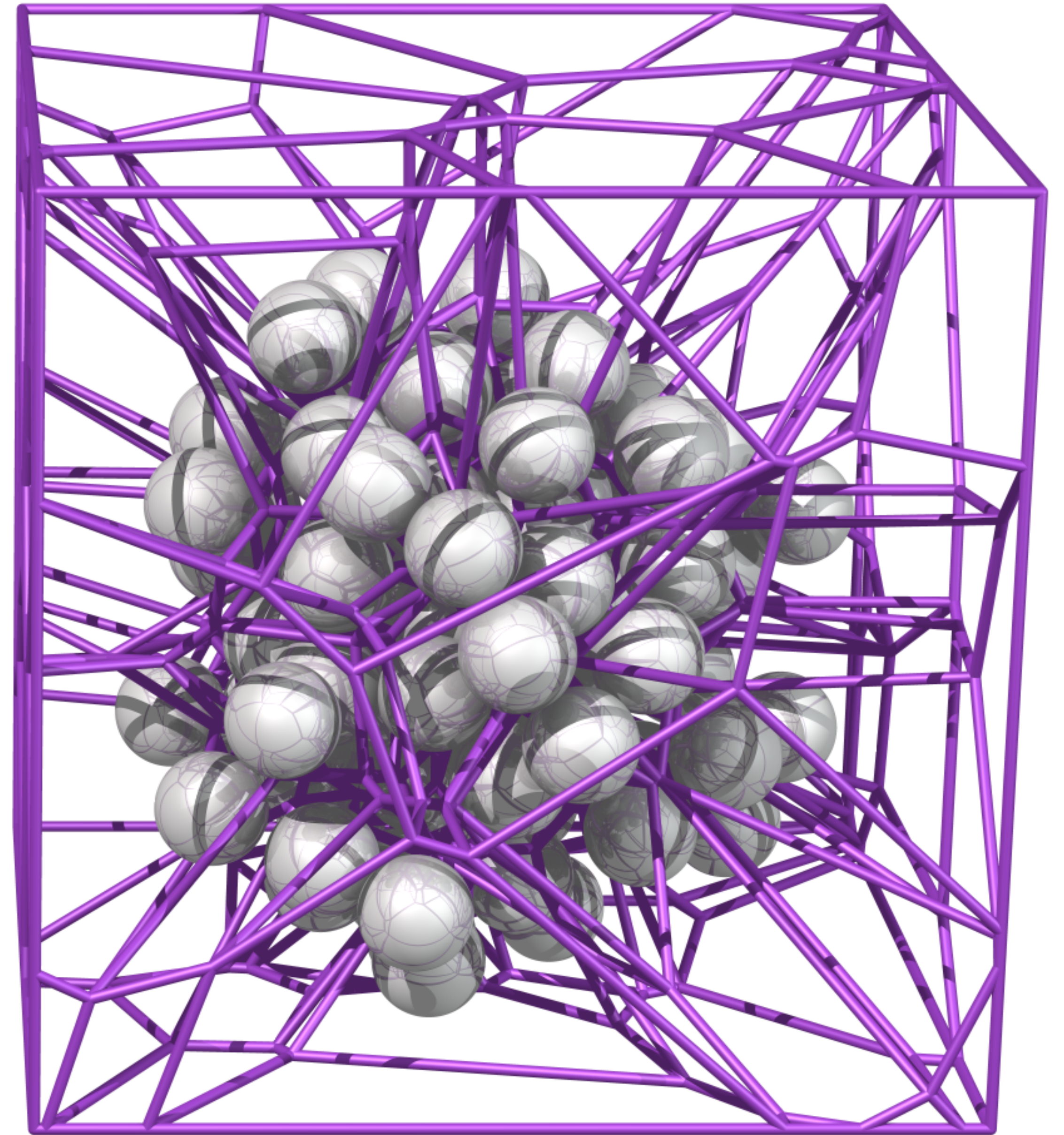
- Another common scenario is to have particles that form an irregular arrangement in space
- These may not conform to any particular domain
- Voronoi cells may extend much further than the arrangement itself

Irregular particle packings

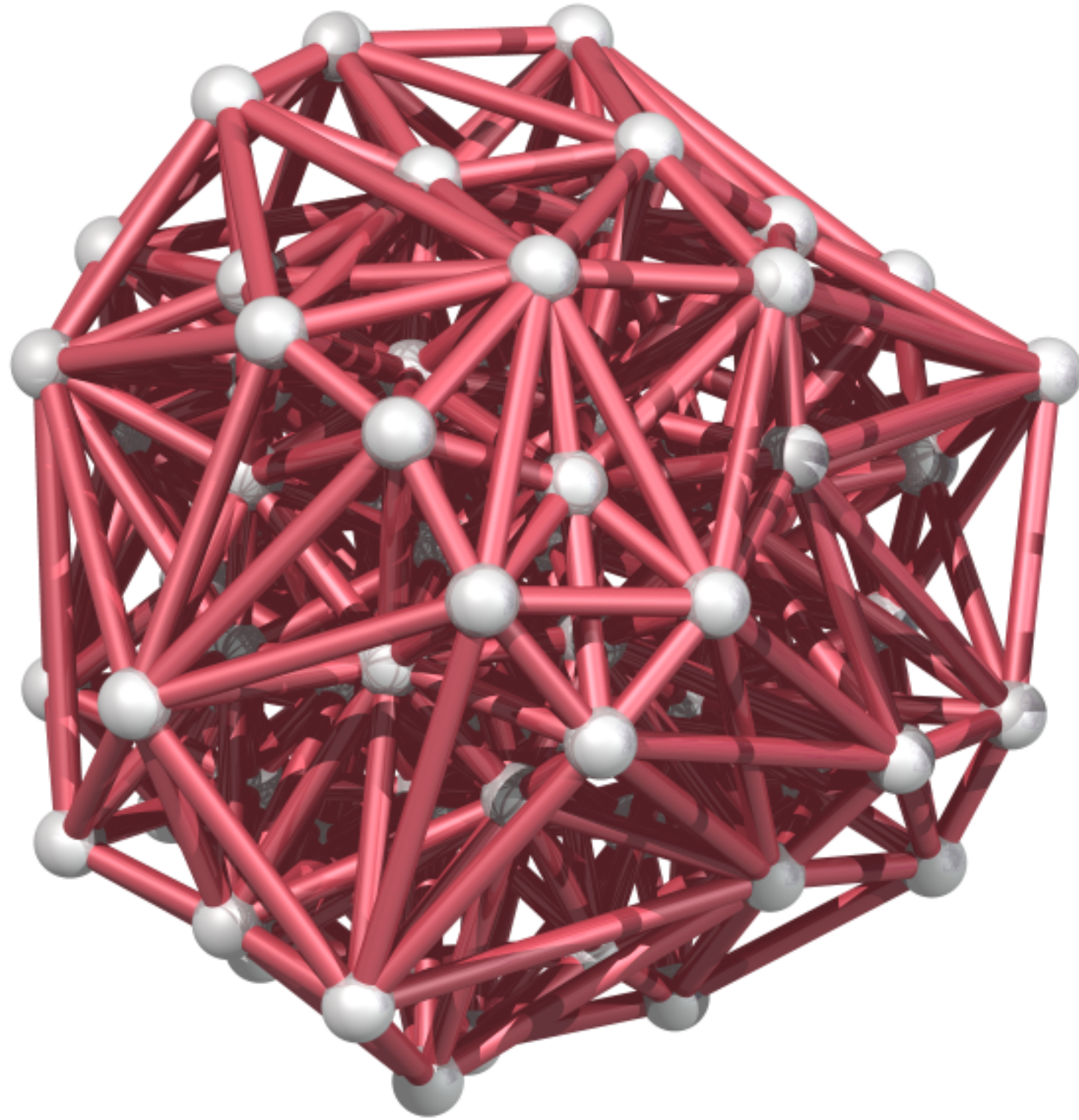


Irregular particle packings

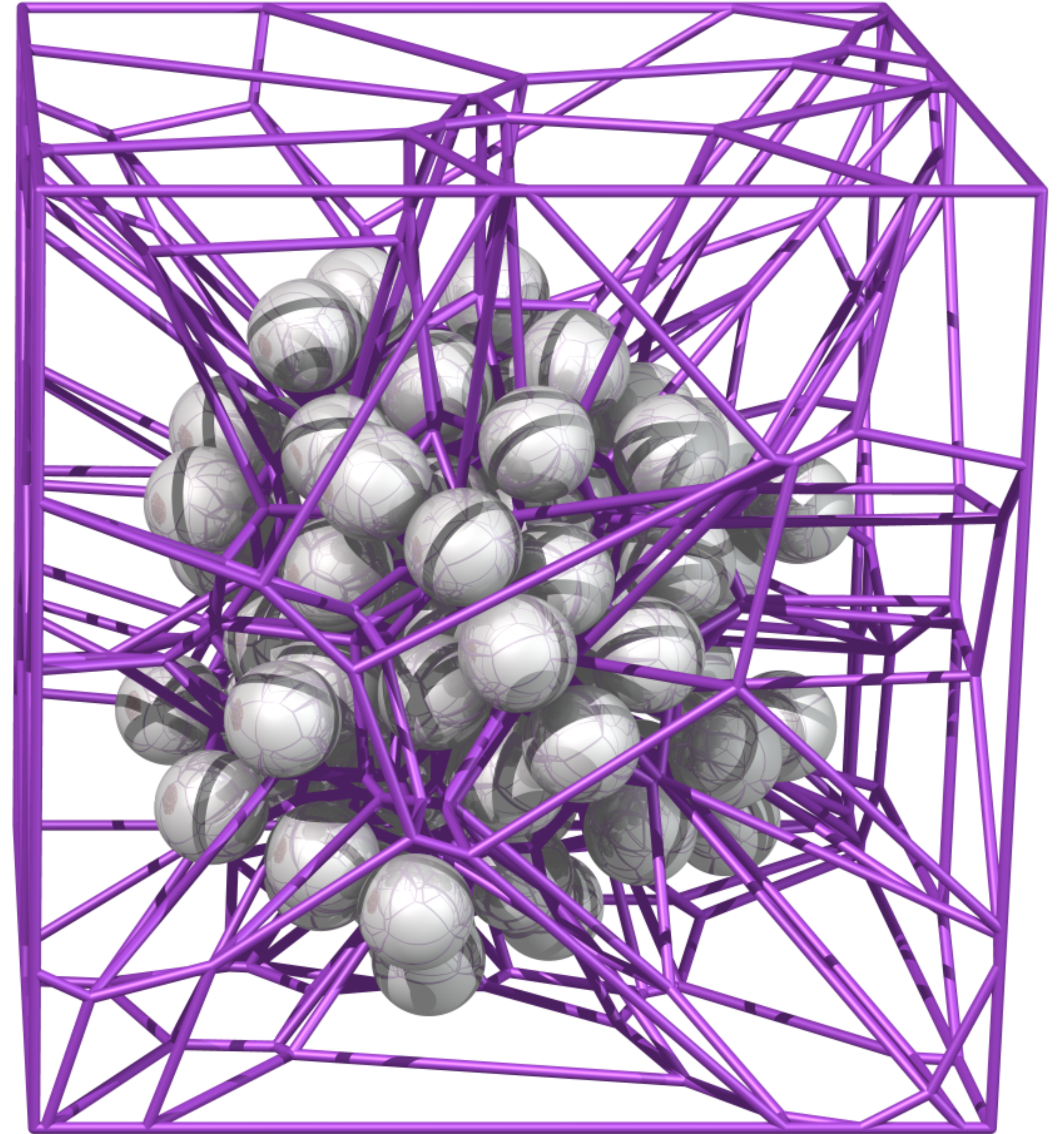
- May have negative consequences for analysis:
 - Very large Voronoi cell volumes for particles on the periphery
 - Large numbers of Voronoi faces, which may be difficult to interpret physically
 - Neighbor connections between particles that are far away



Irregular particle packings



Neighbor relations are
shown in red



Irregular particle packings

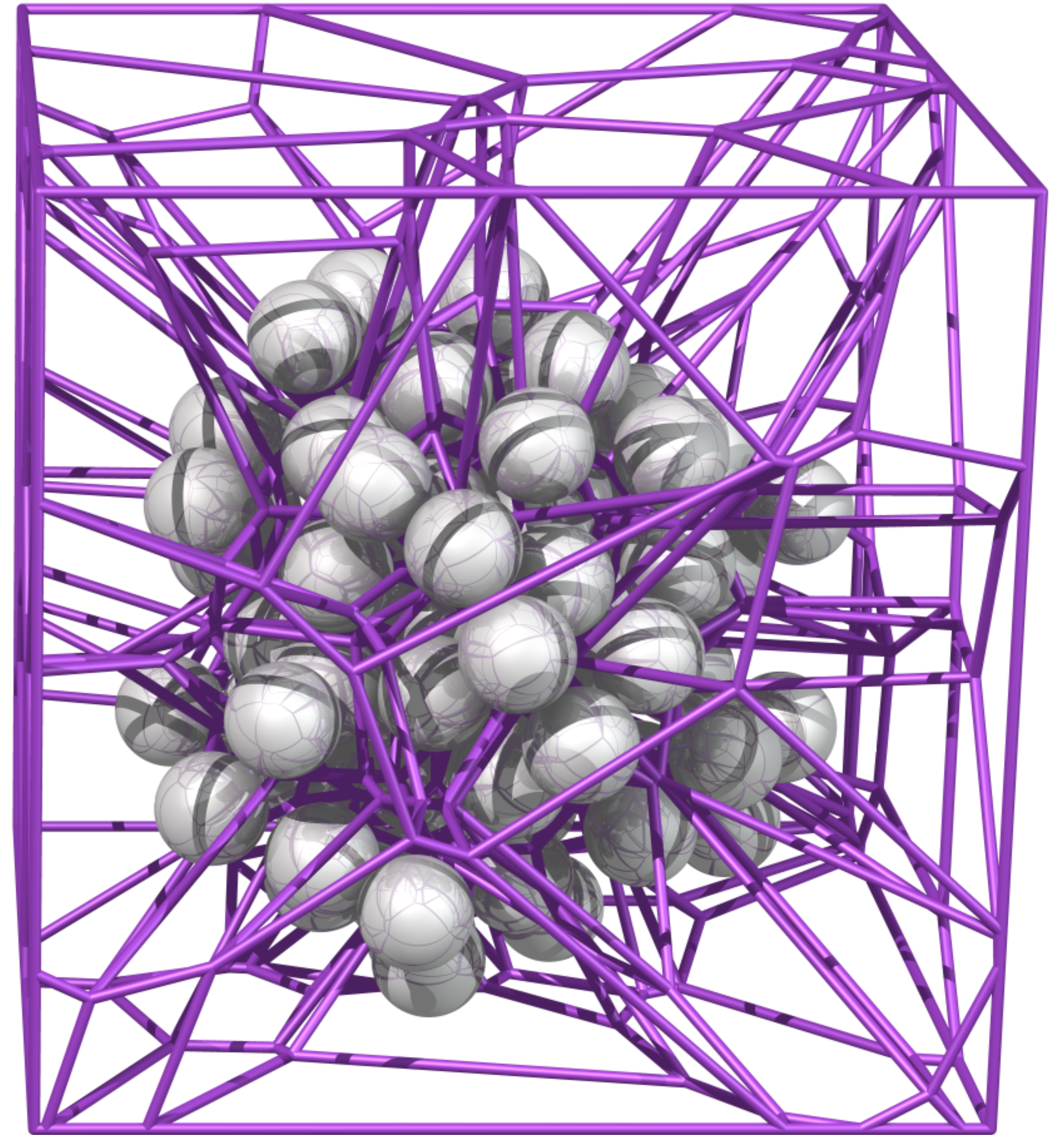
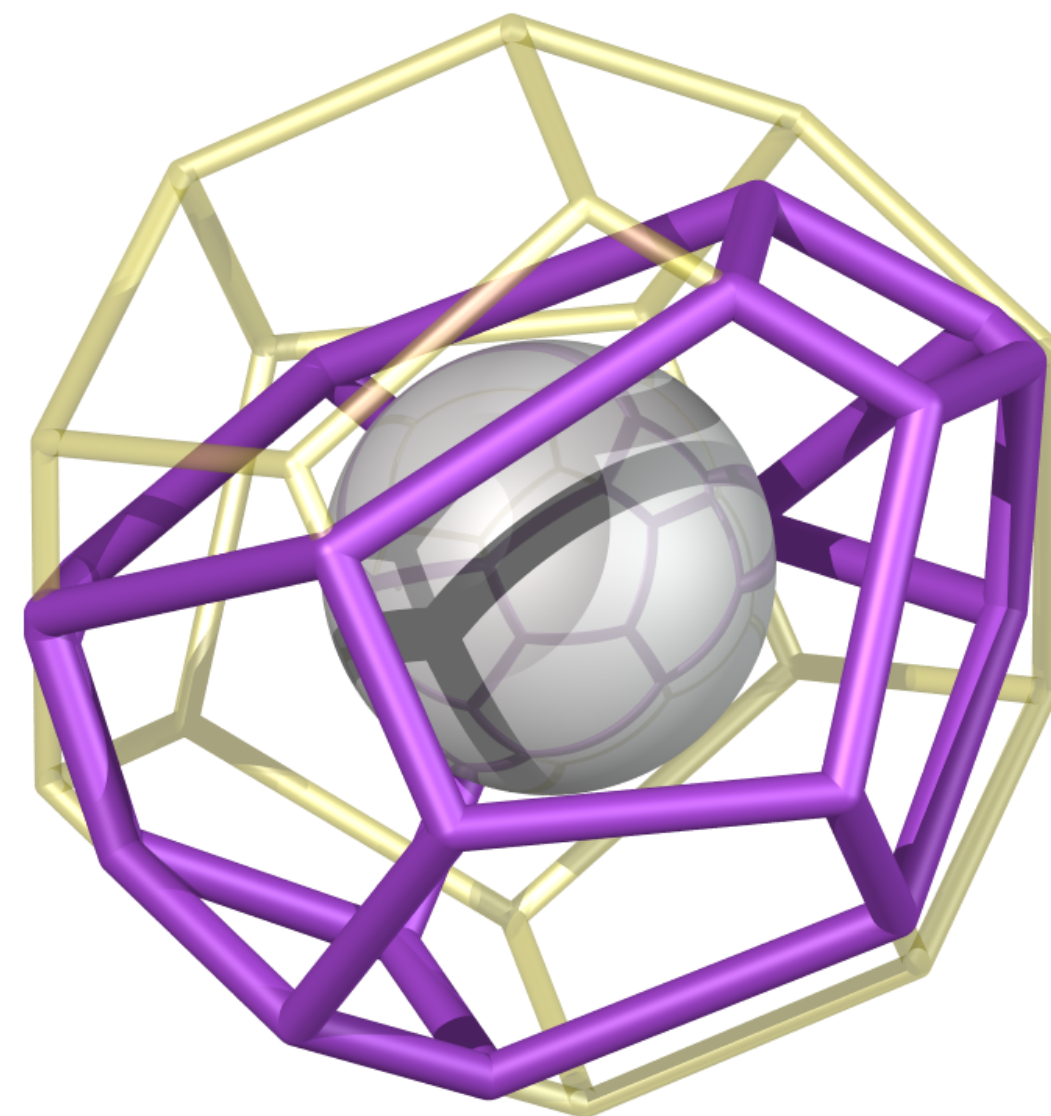
- Usually each Voronoi cell is initialized to fill the entire domain
- Alternative scenario: start each Voronoi cell as a reference shape (e.g. a dodecahedron)

Yellow lines

indicate initial shape

Purple lines

indicate Voronoi cell
after plane cuts



Irregular particle packings

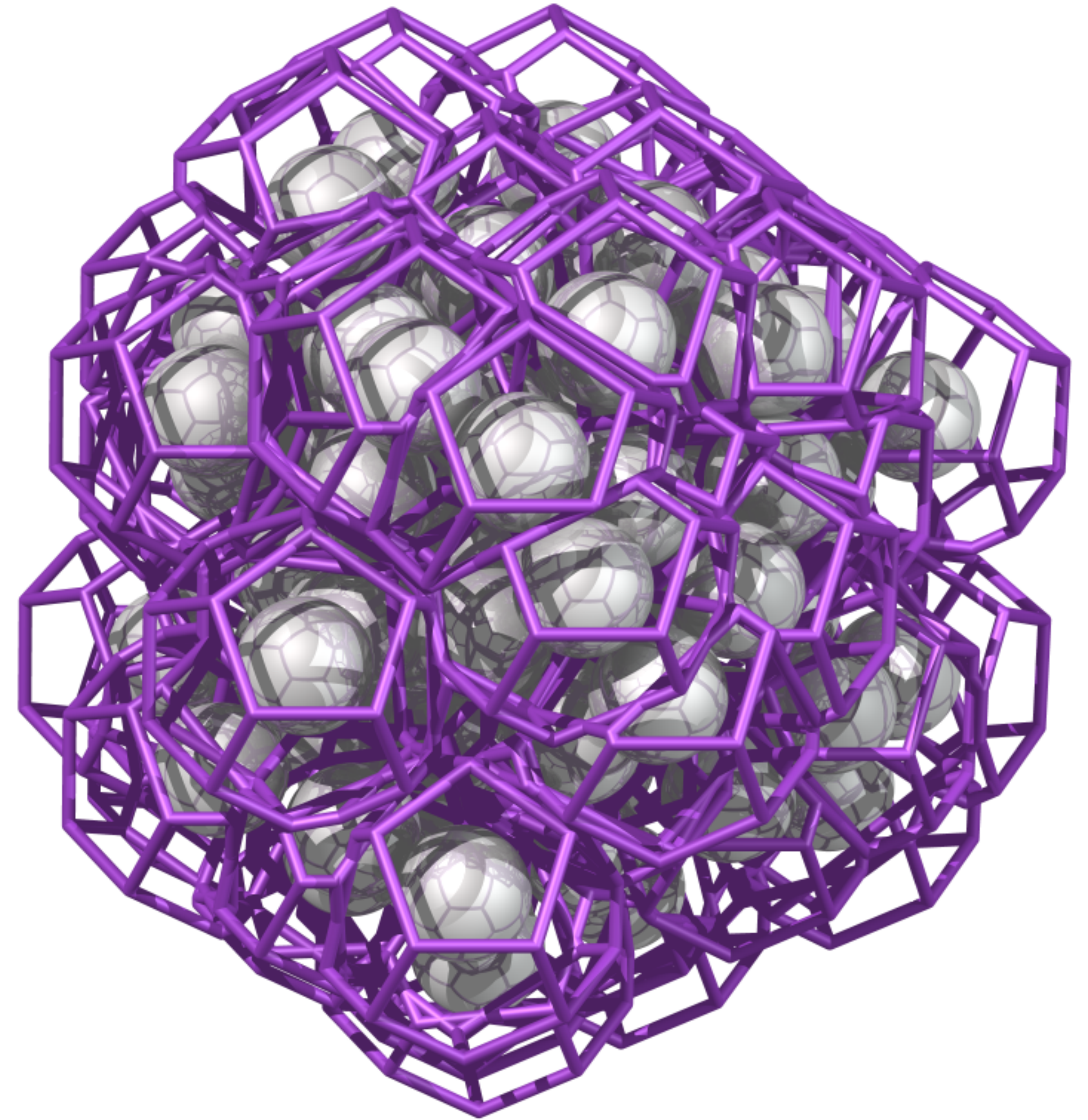
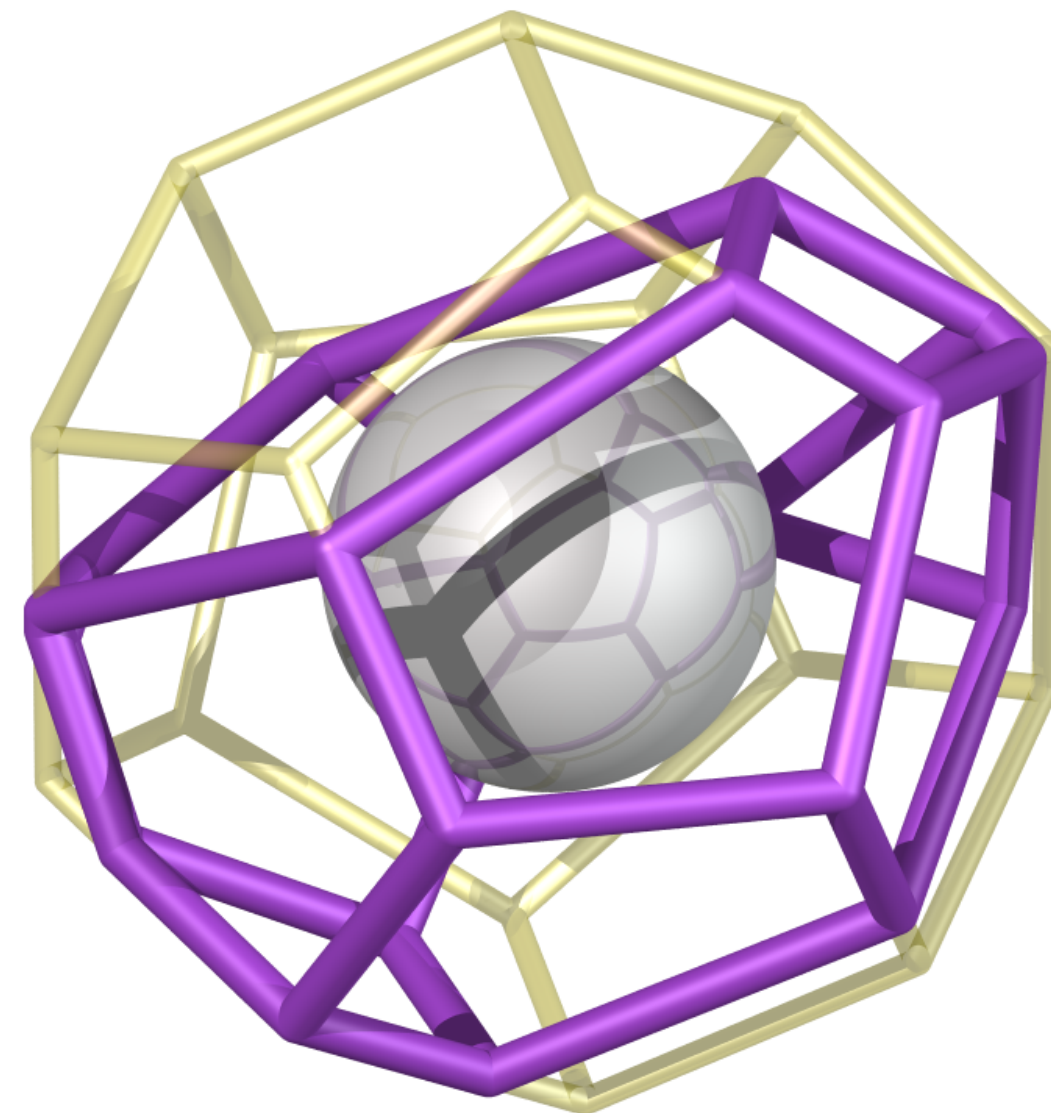
- With this method, the Voronoi cell structure conforms to the packing arrangement
- More efficient to compute, since small cells require less searching

Yellow lines

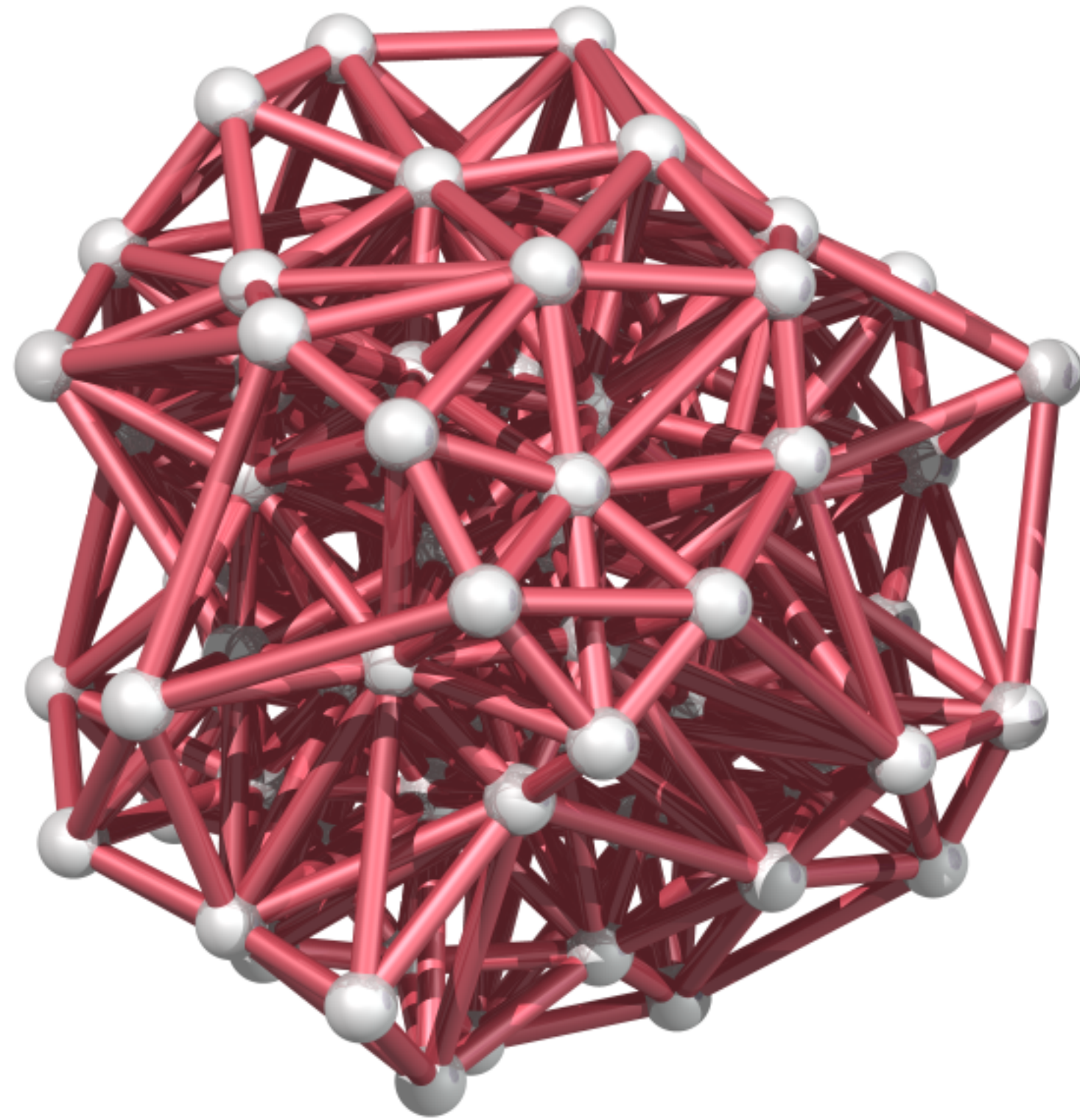
indicate initial shape

Purple lines

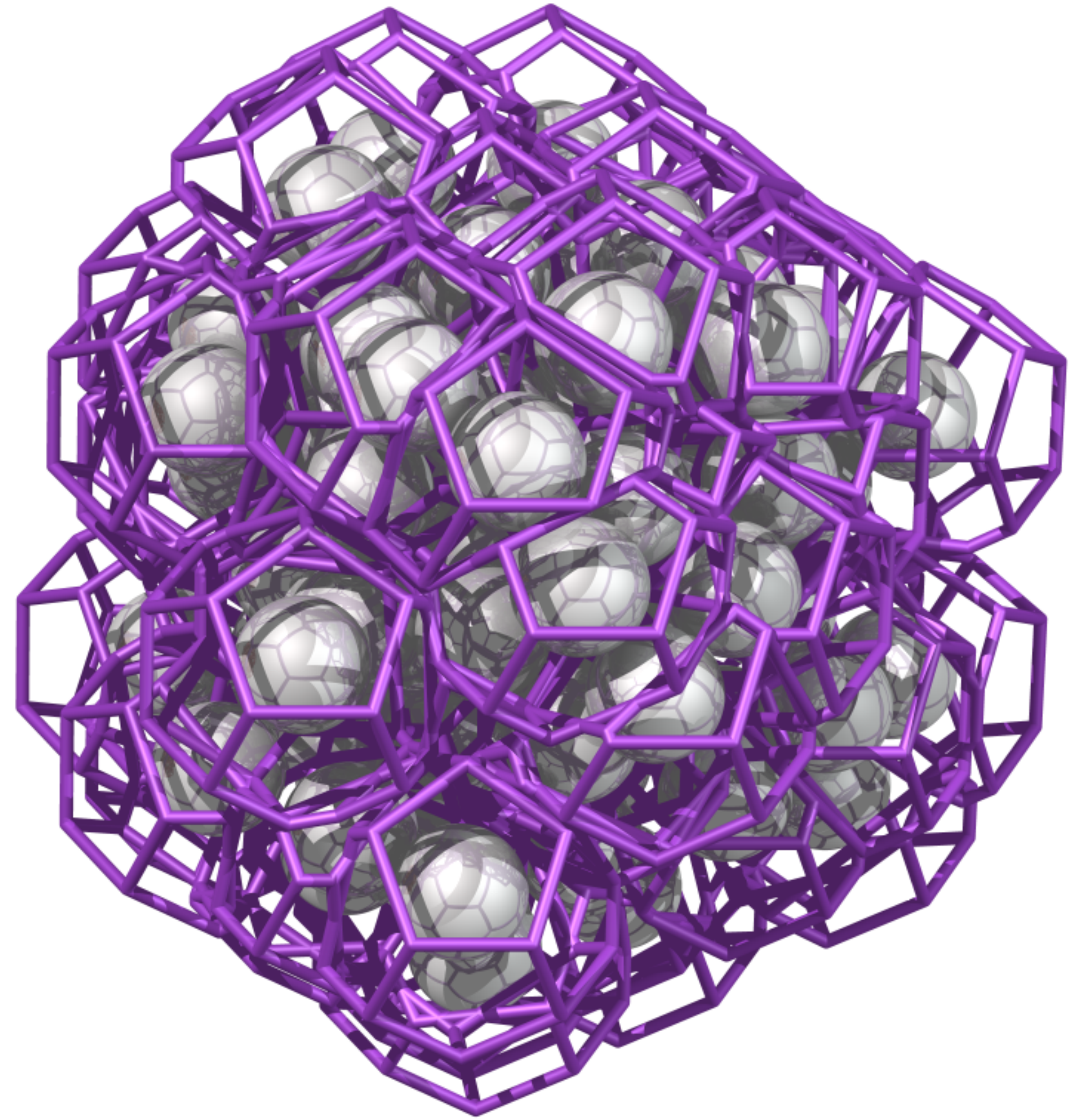
indicate Voronoi cell
after plane cuts



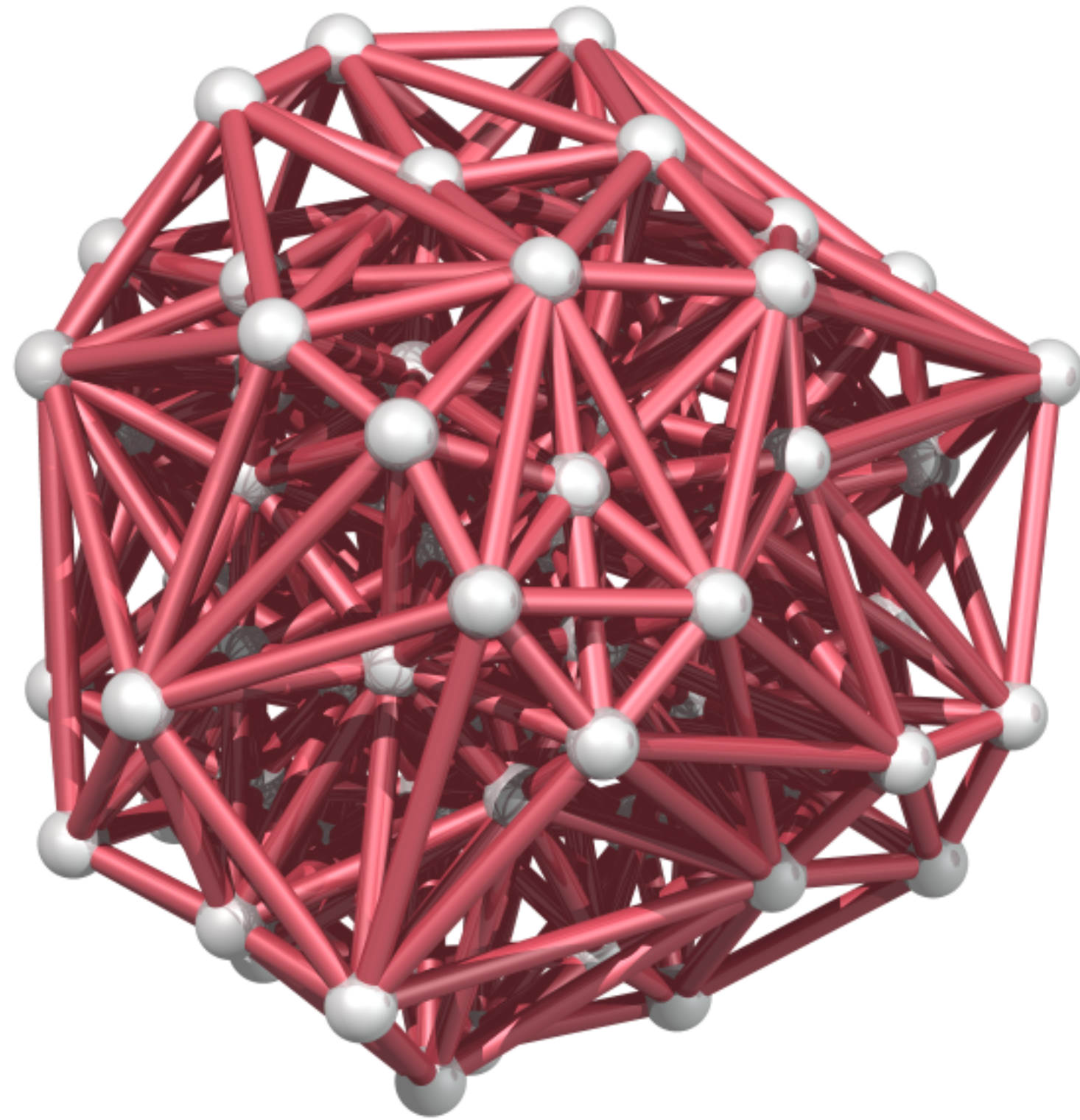
Irregular particle packings



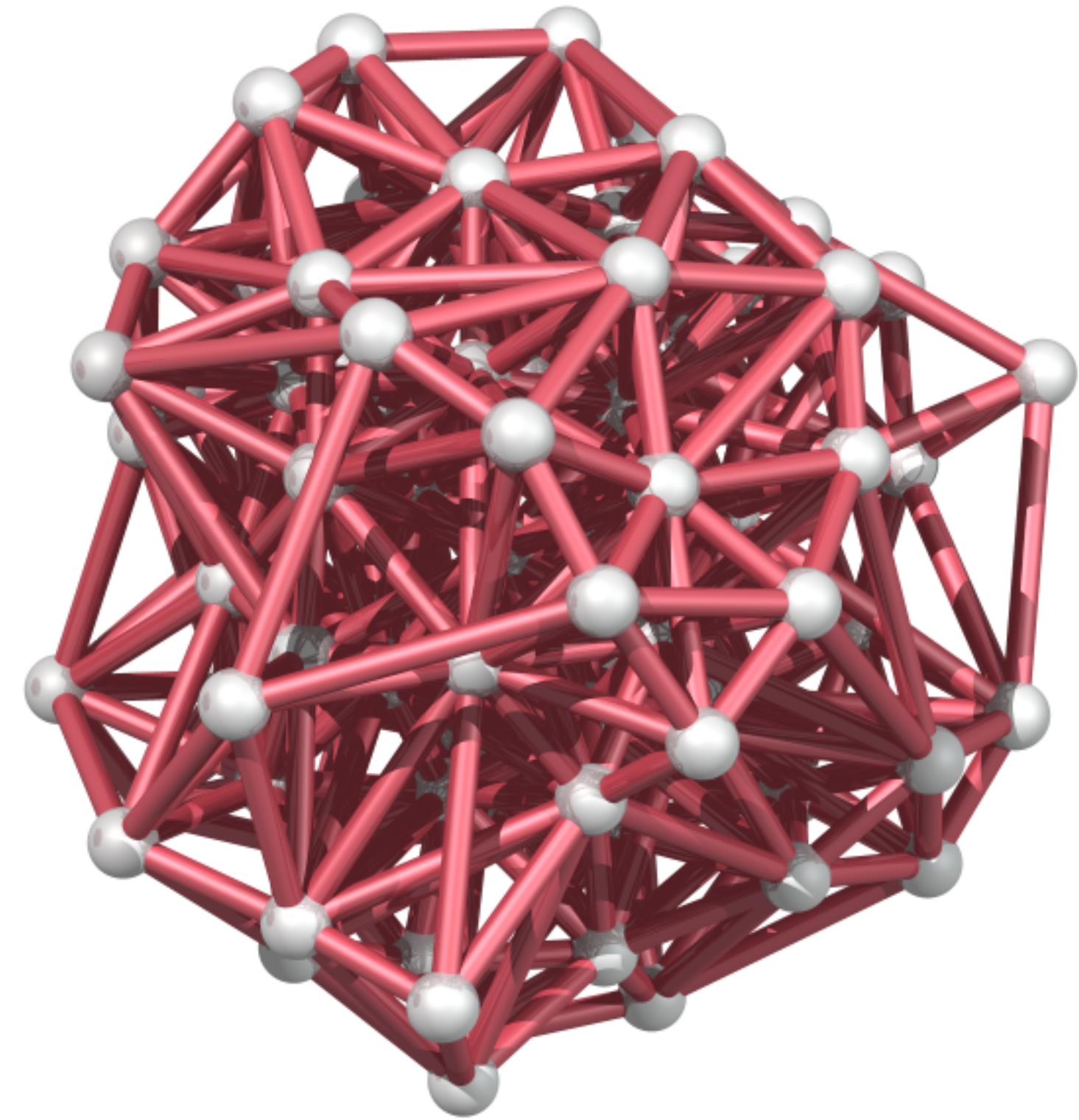
Neighbor relations are
shown in red



Irregular particle packings

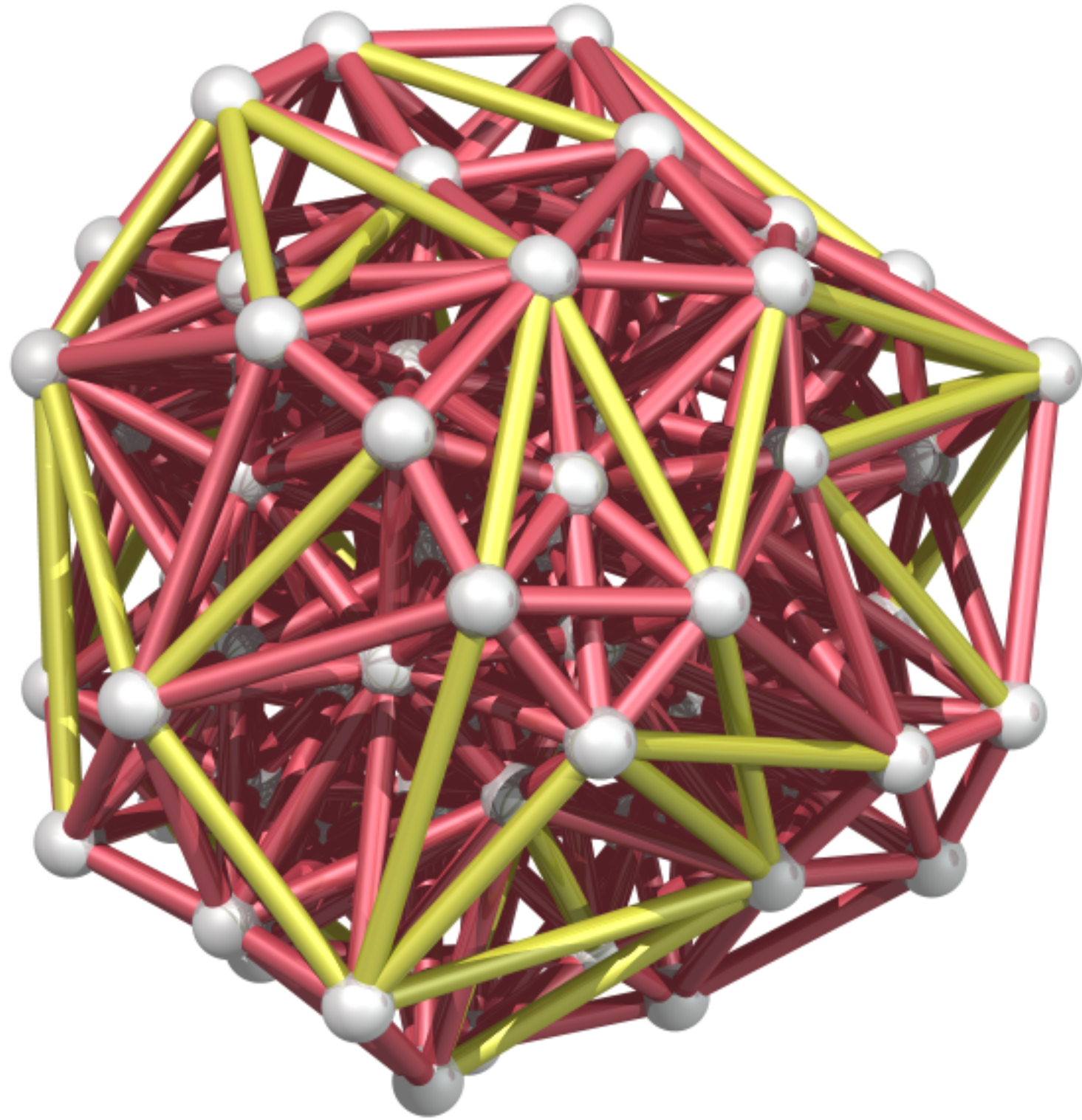


Neighbor relations from original
Voronoi tessellation



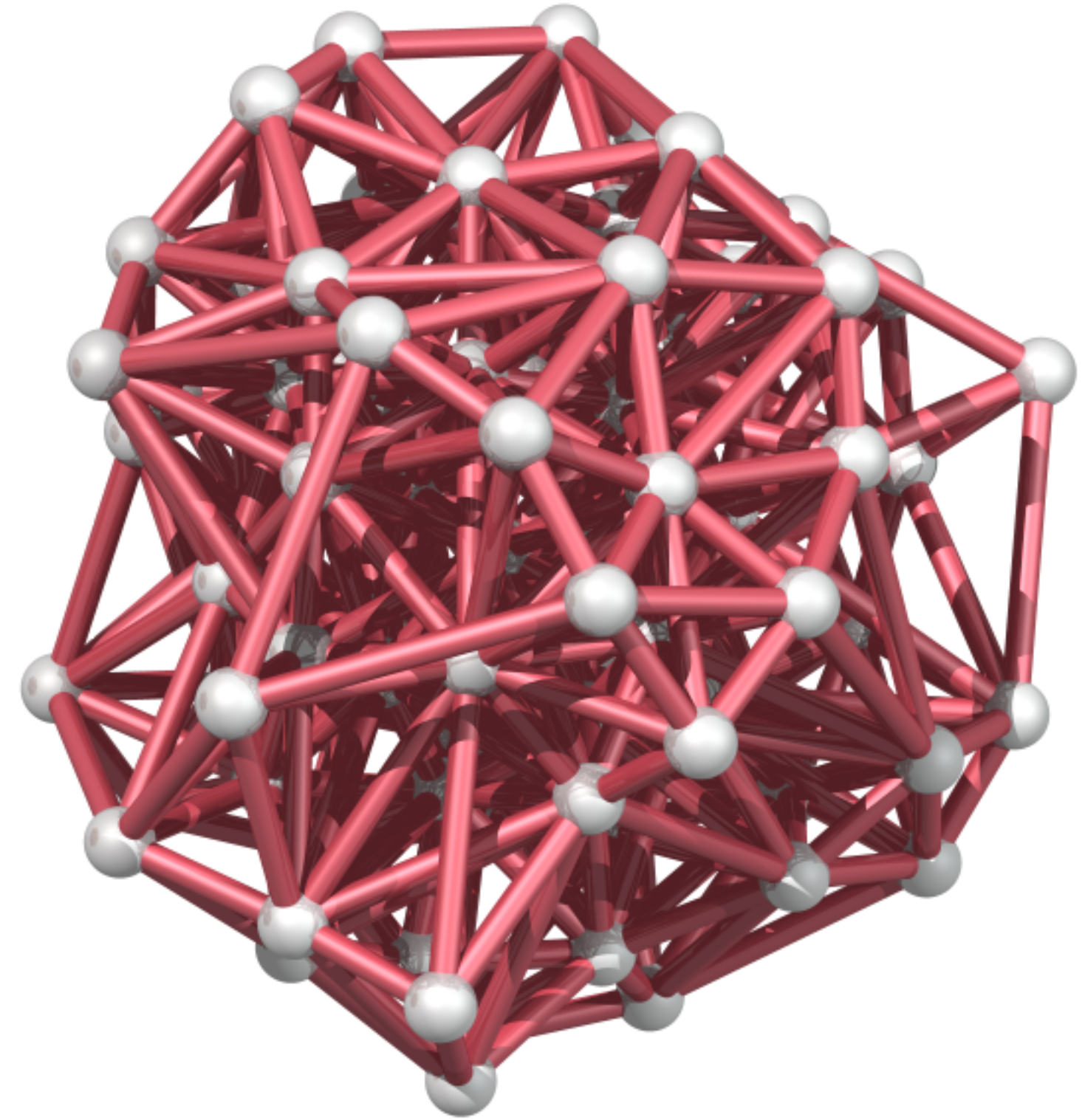
Neighbor relations from
truncated Voronoi tessellation

Irregular particle packings



Neighbor relations from original
Voronoi tessellation

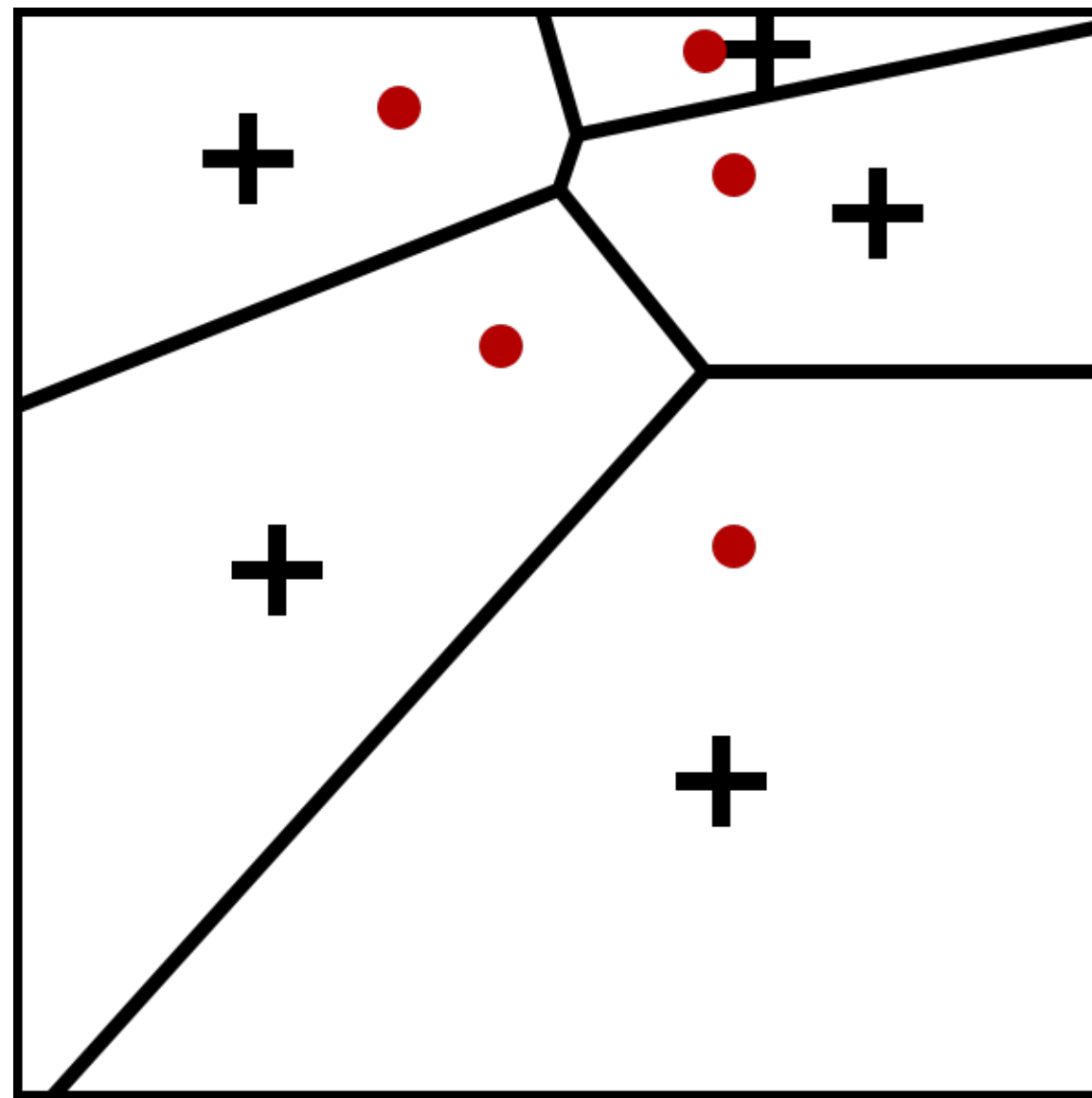
(extra connections shown in yellow)



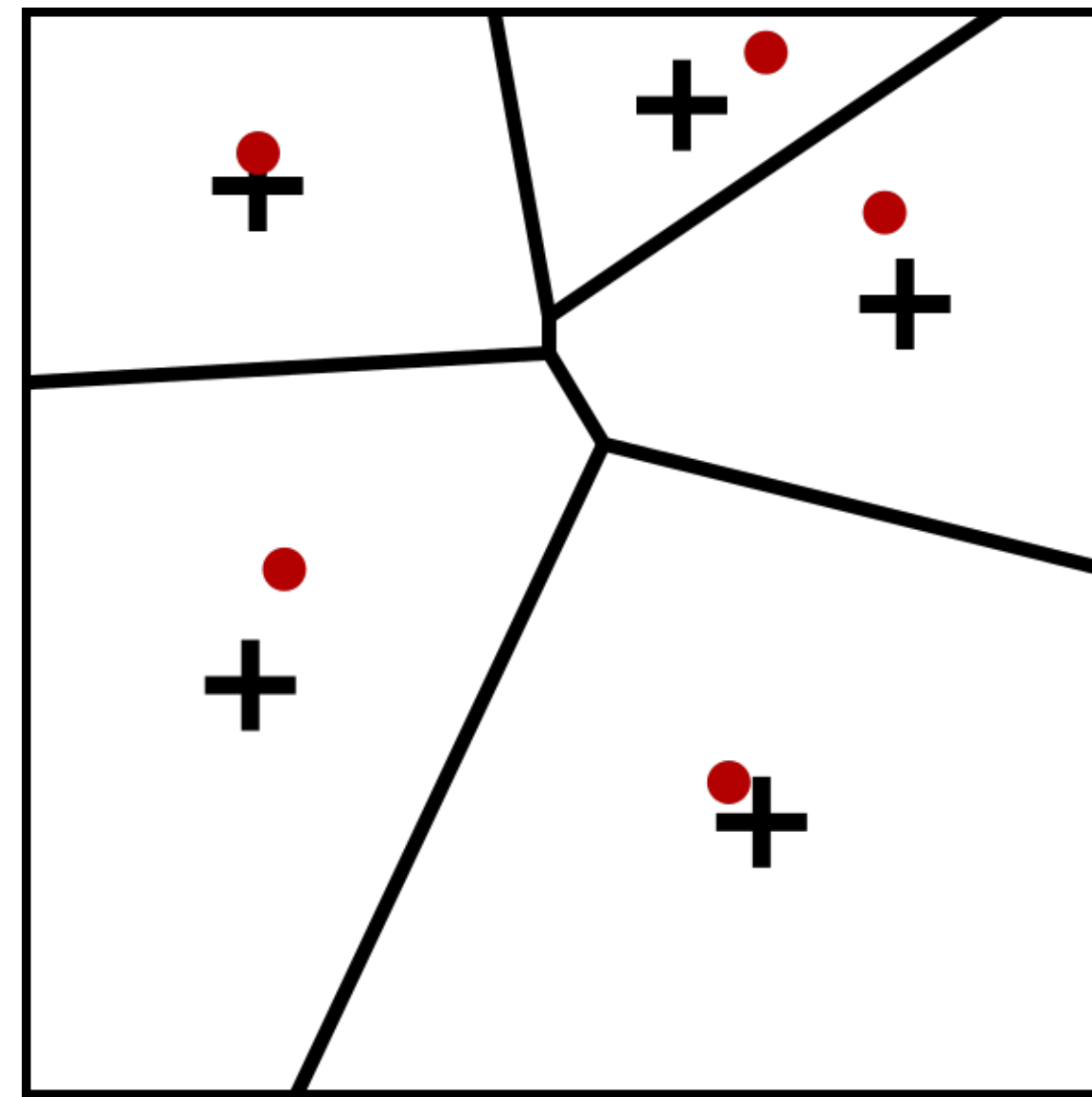
Neighbor relations from
truncated Voronoi tessellation

Regularizing point arrangements

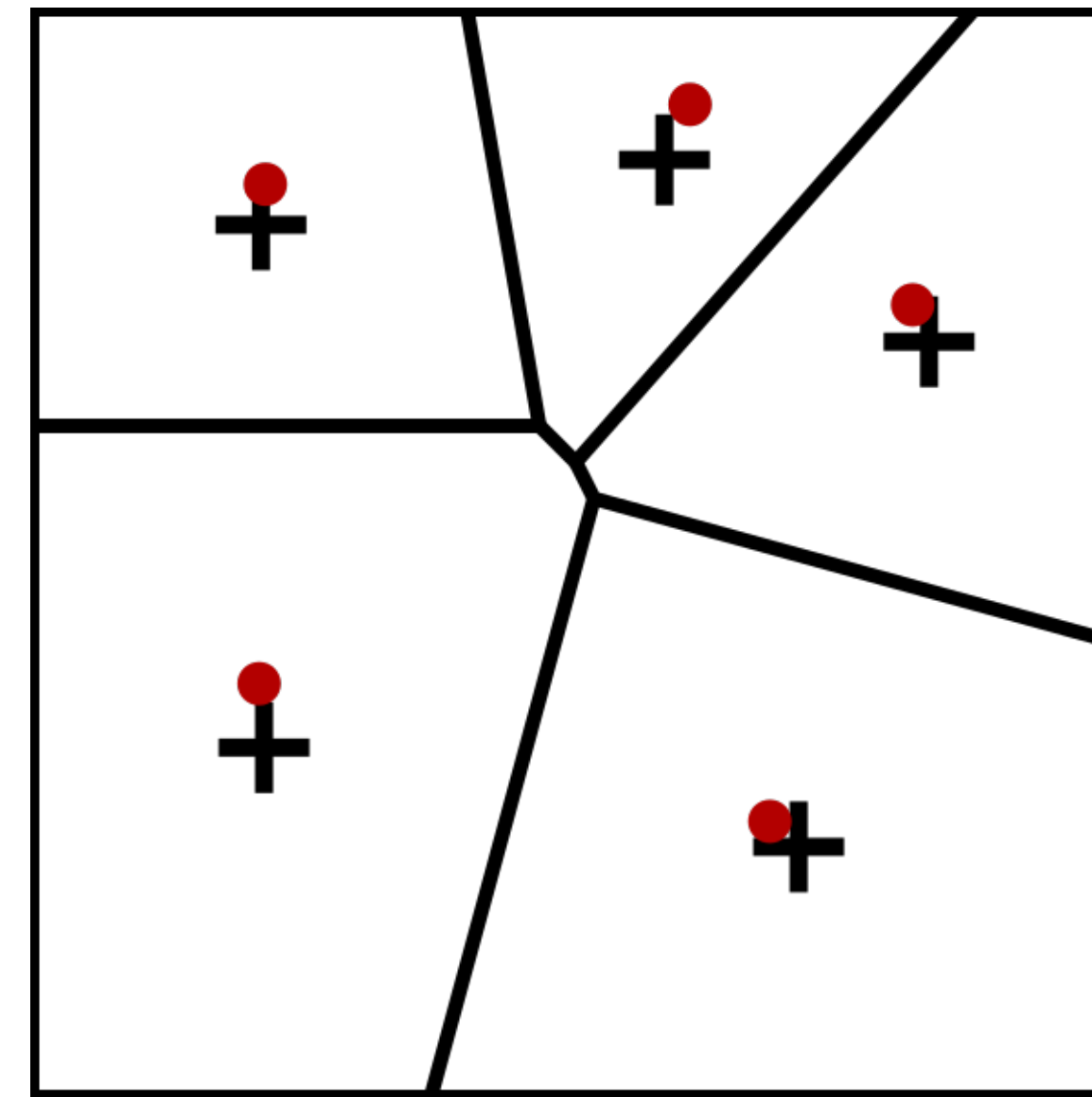
- Lloyd's algorithm: iteratively move points to the centroids of their Voronoi cells
- Mimics pattern formation PDEs and give direct description of region boundaries



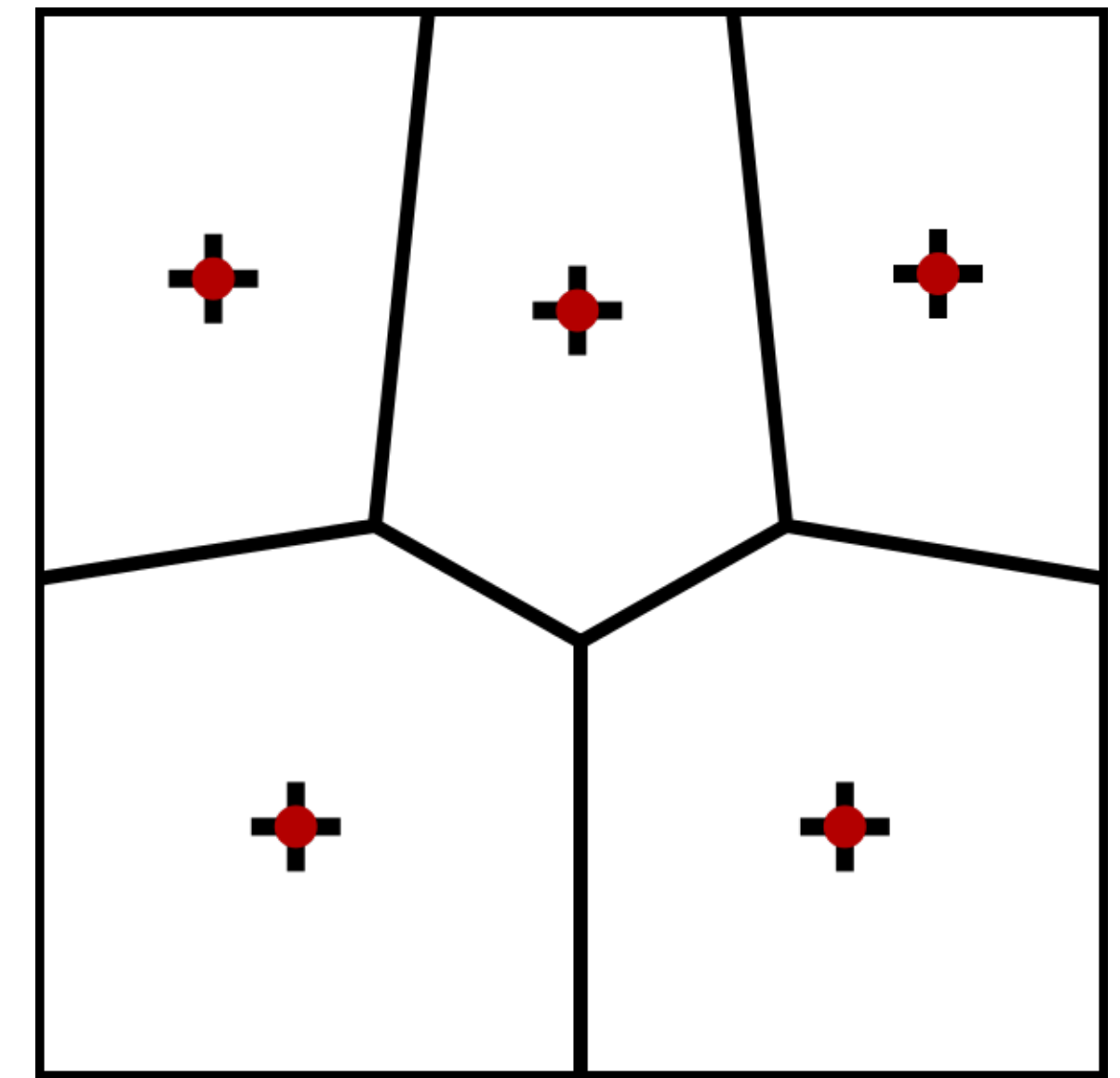
Iteration 1



Iteration 2



Iteration 3

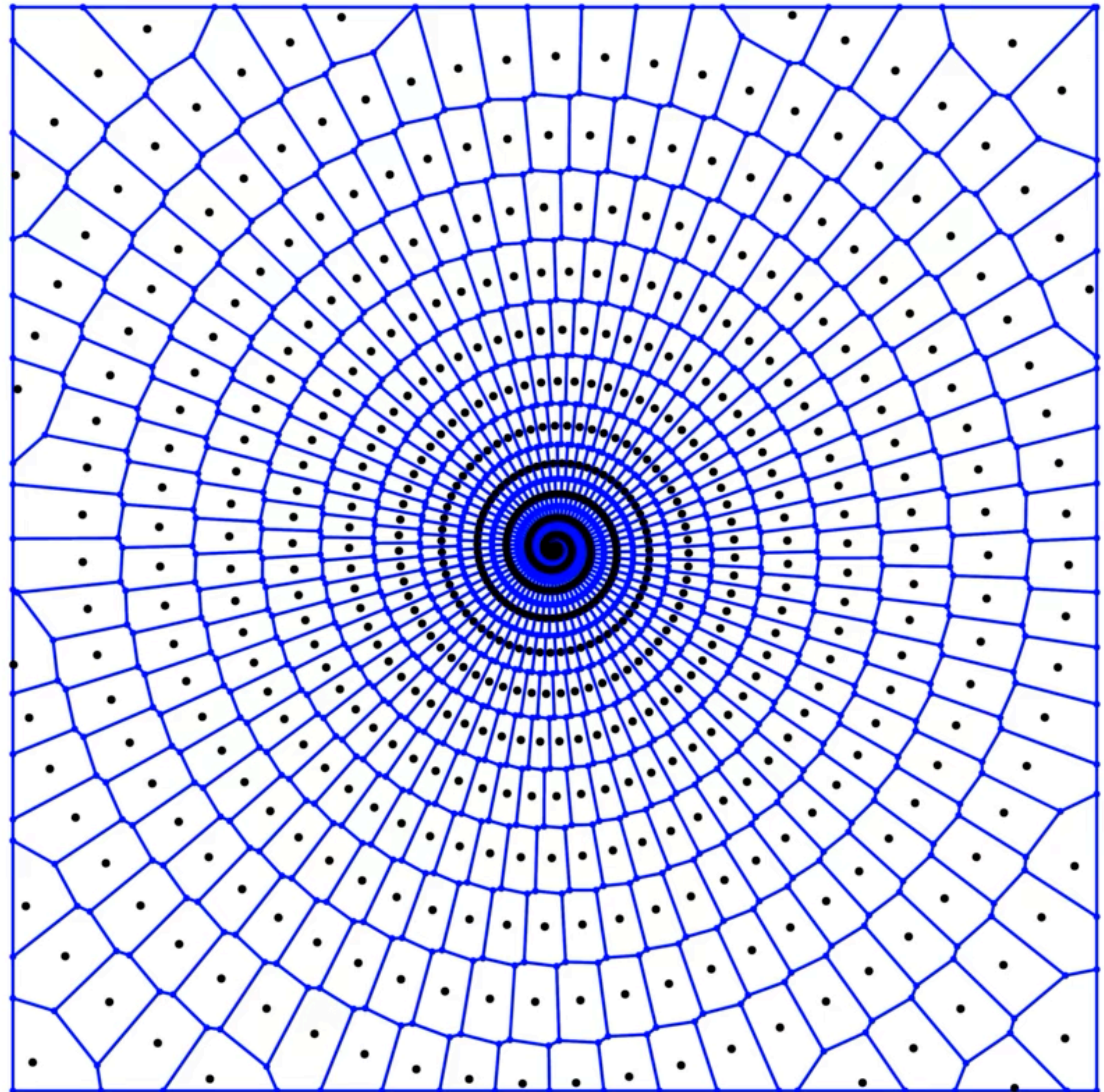


Iteration 15

Converges to **centroidal
Voronoi tessellation (CVT)**

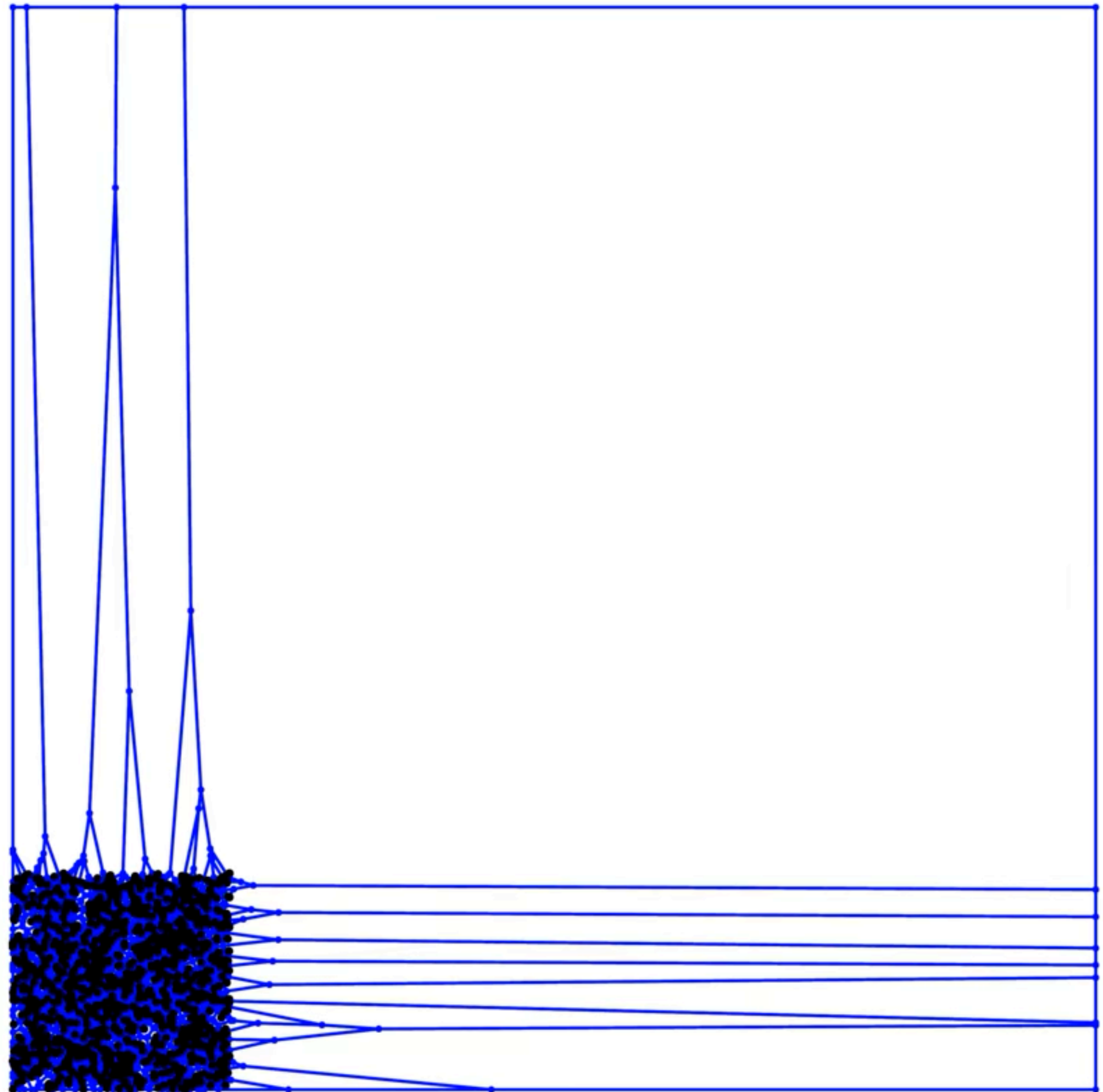
Lloyd's algorithm: example 2

- Start with 838 particles in a spiral and run 256 iterations of Lloyd's algorithm
- Particle positions even out
- Asymptotically in a large domain, the Voronoi cells become regular hexagons



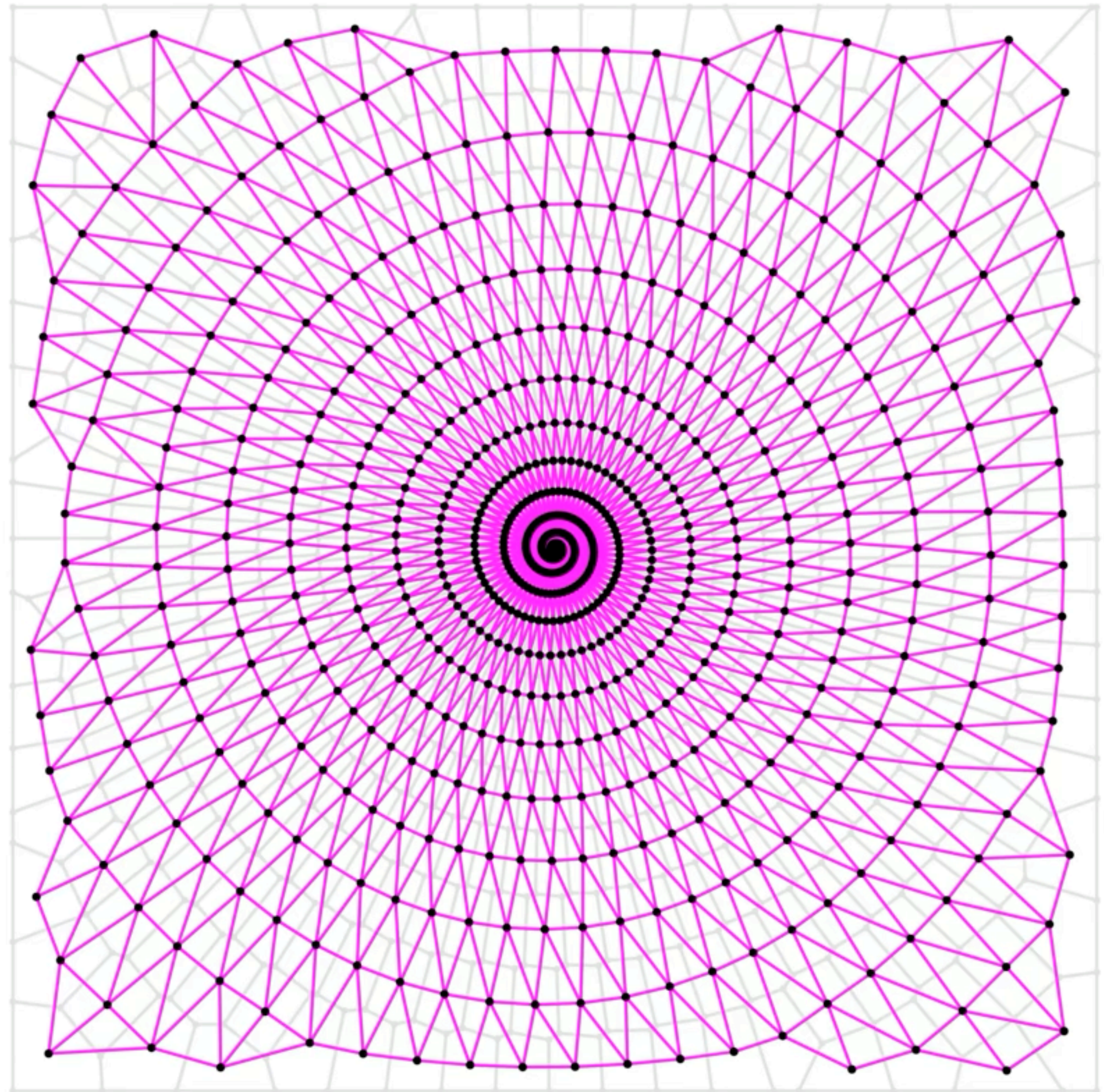
Lloyd's algorithm: example 2

- Start with 1000 particles in a domain covering a fifth of each side, and run 1024 Lloyd iterations
- Short-range density fluctuations are quickly damped out
- Long-range density fluctuations take longer to damp out

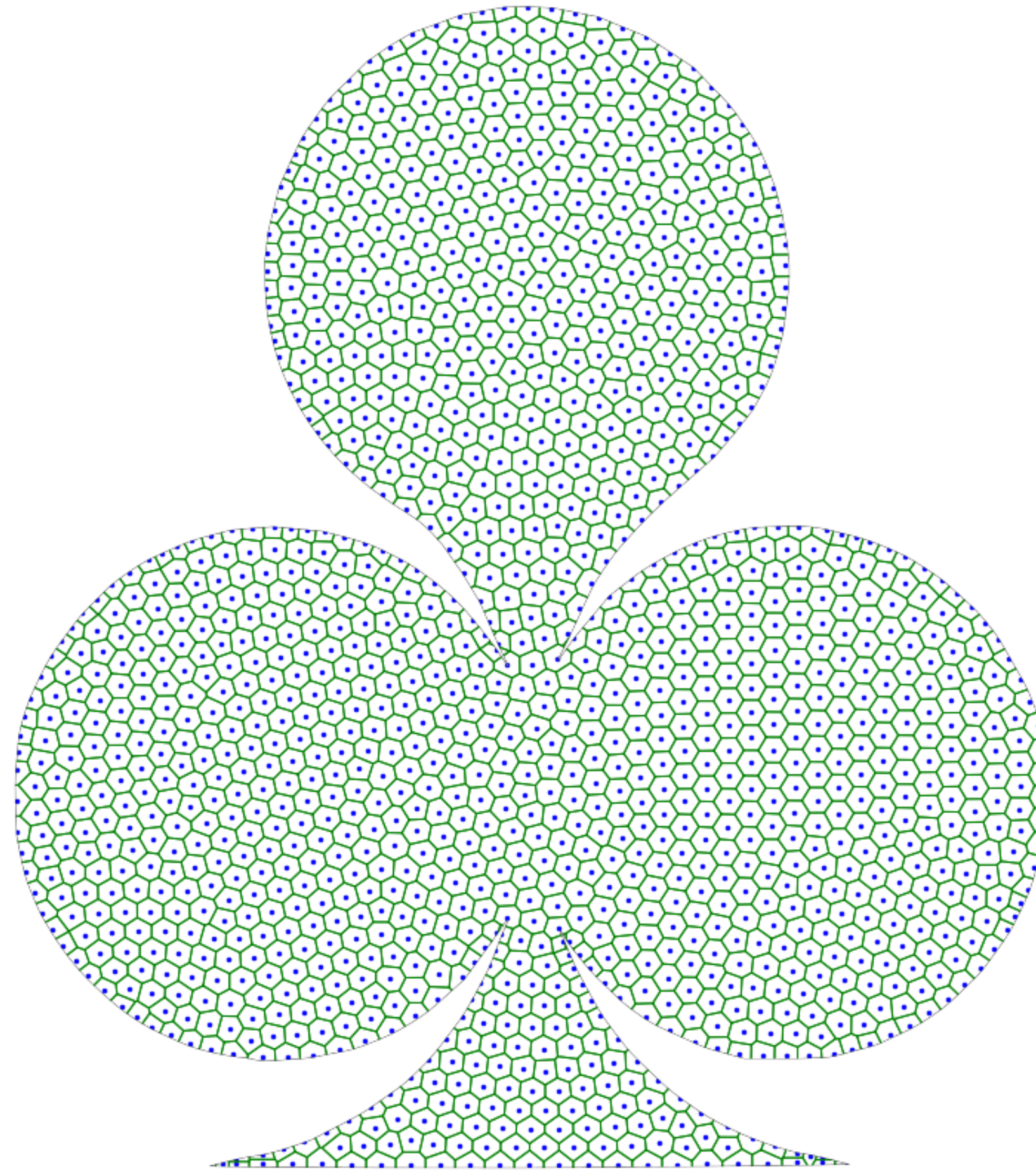


Mesh generation with Lloyd's algorithm

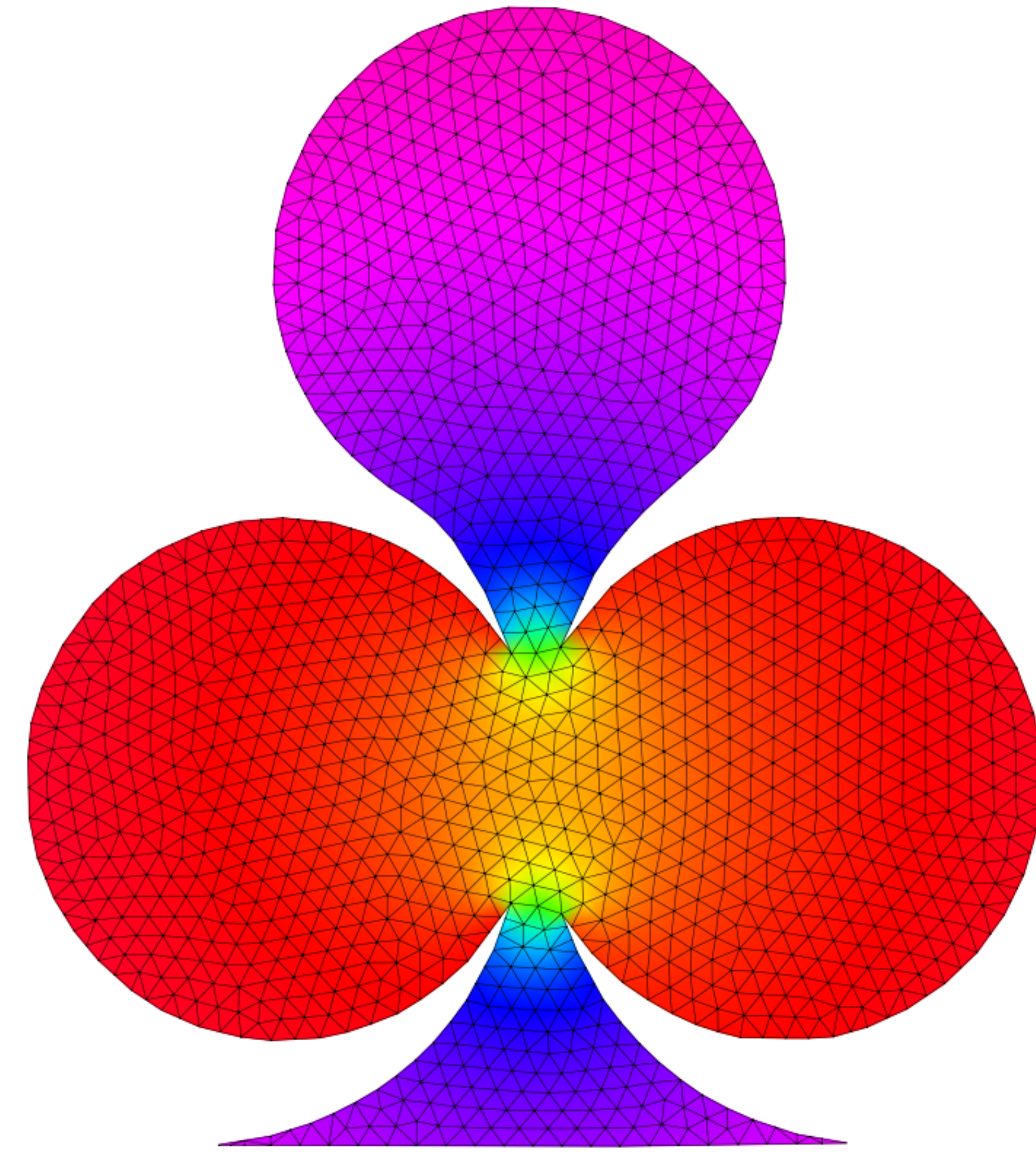
- The Delaunay triangulation of the particles after applying Lloyd iterations is a good computational mesh
- Tends to favor near-equilateral triangles, which are good for numerical methods like the finite-element method (FEM)



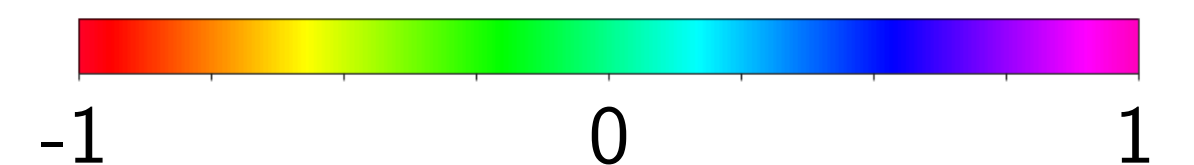
Meshing in complicated domains

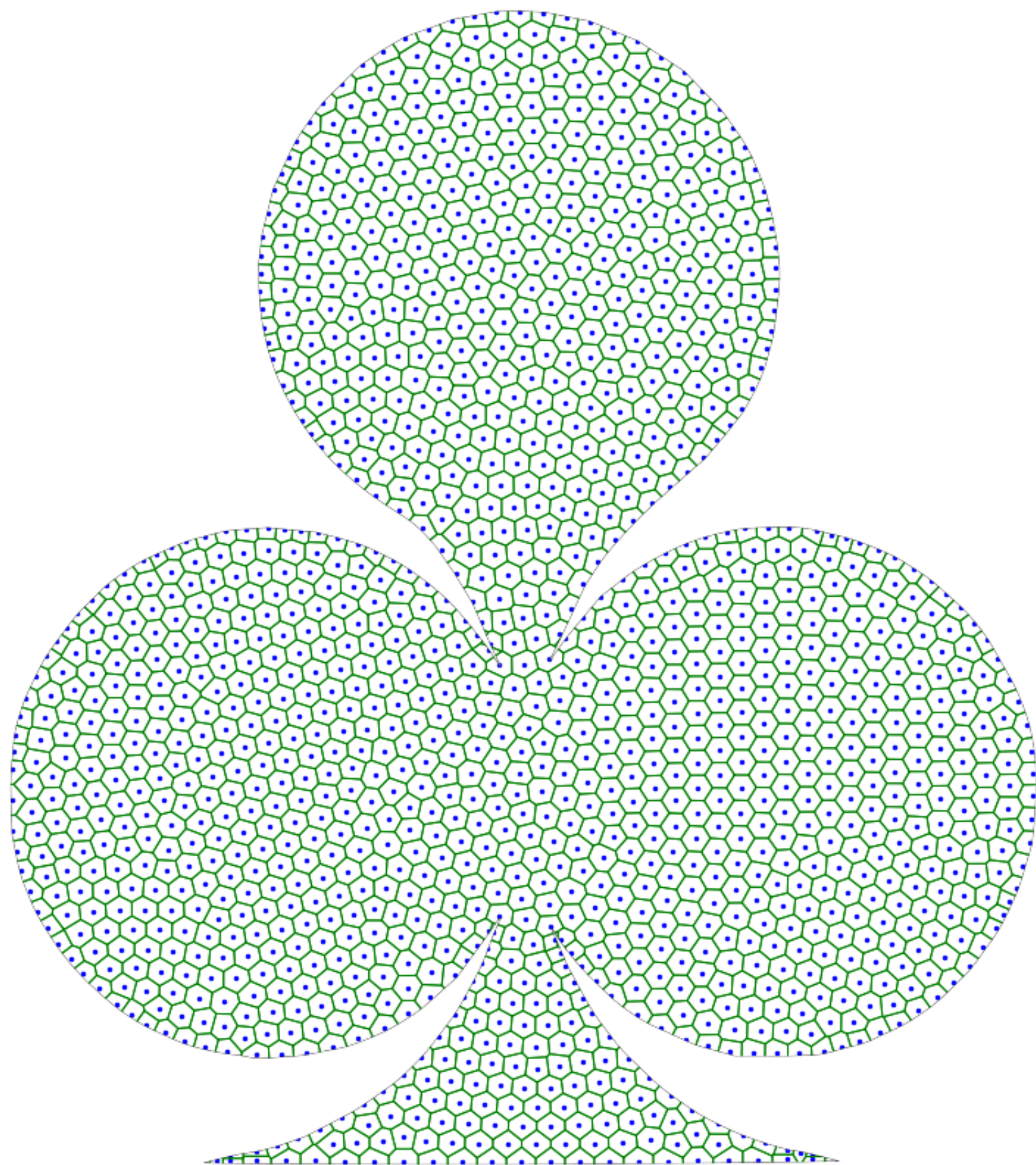


Voronoi cells in club shape after Lloyd's algorithm

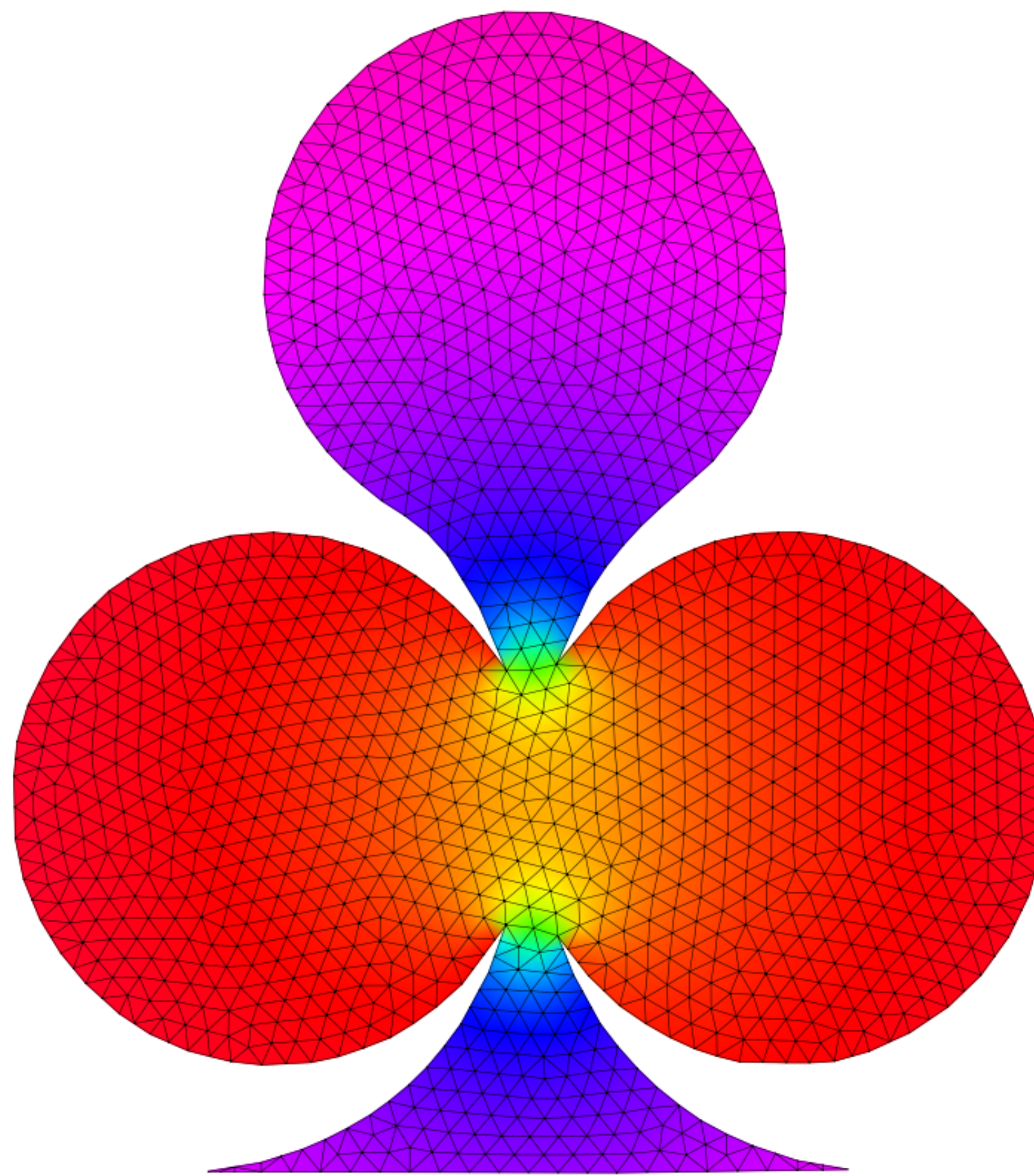


Delaunay mesh and reference FEM solution to Laplace equation

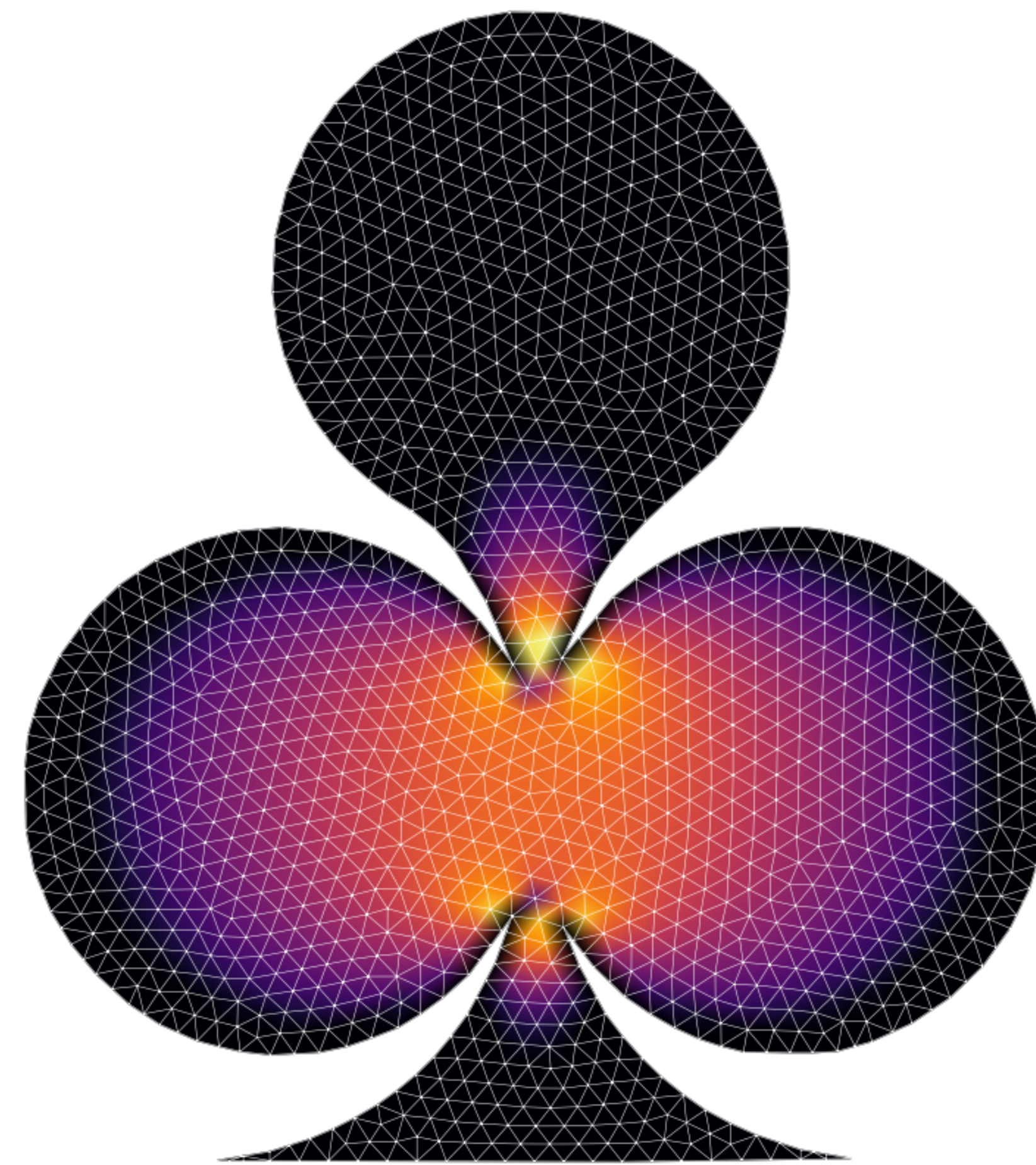
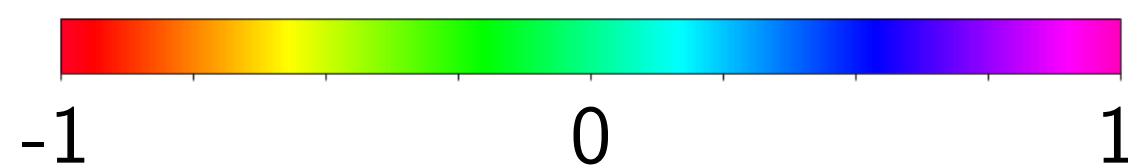




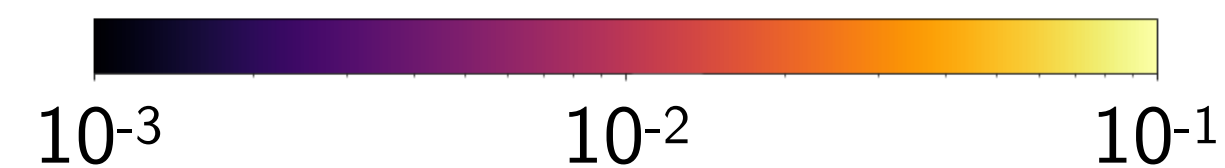
Voronoi cells in club
shape after Lloyd's
algorithm

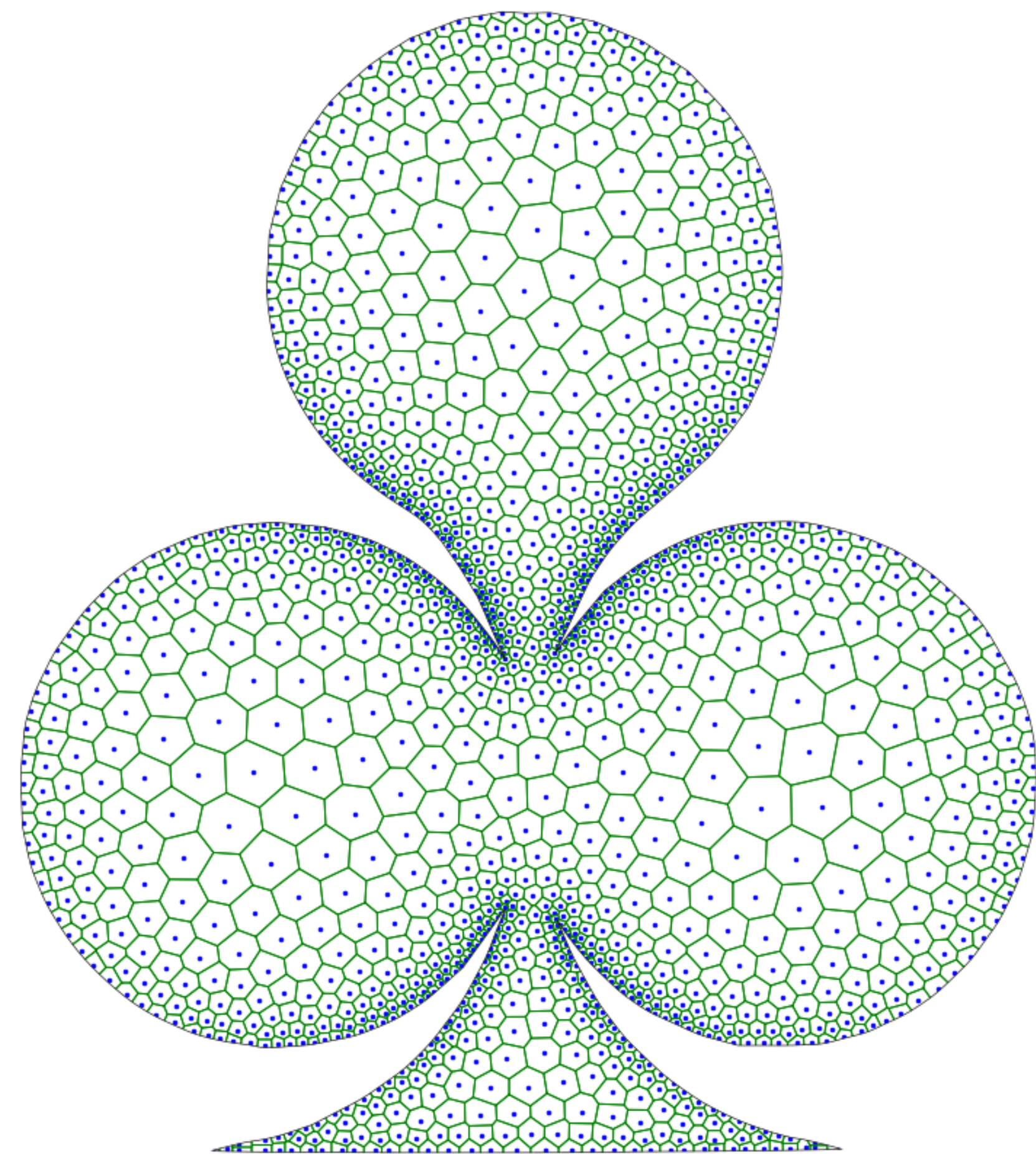


Delaunay mesh and
reference FEM solution
to Laplace equation

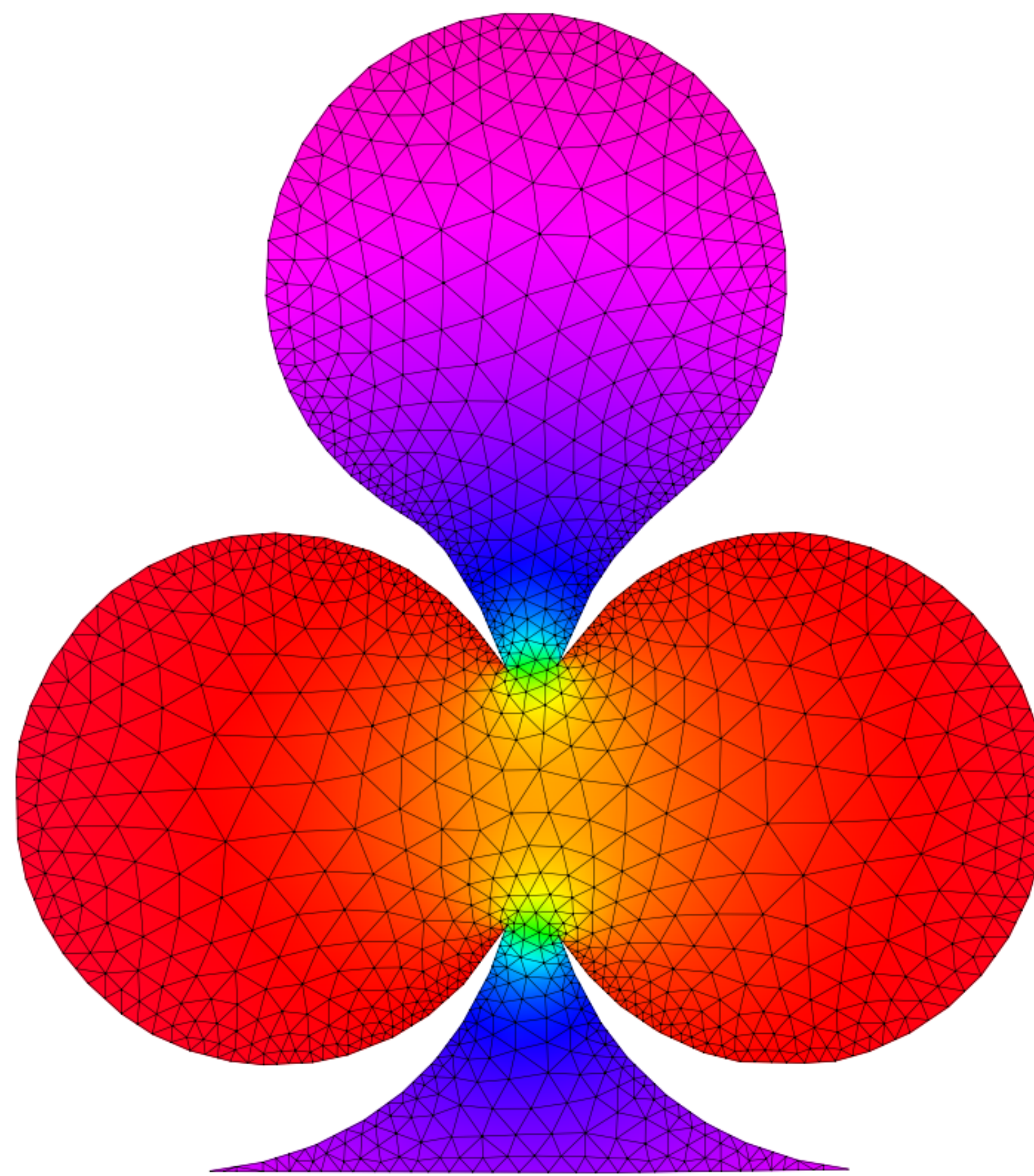


Error between FEM
solution and exact
solution

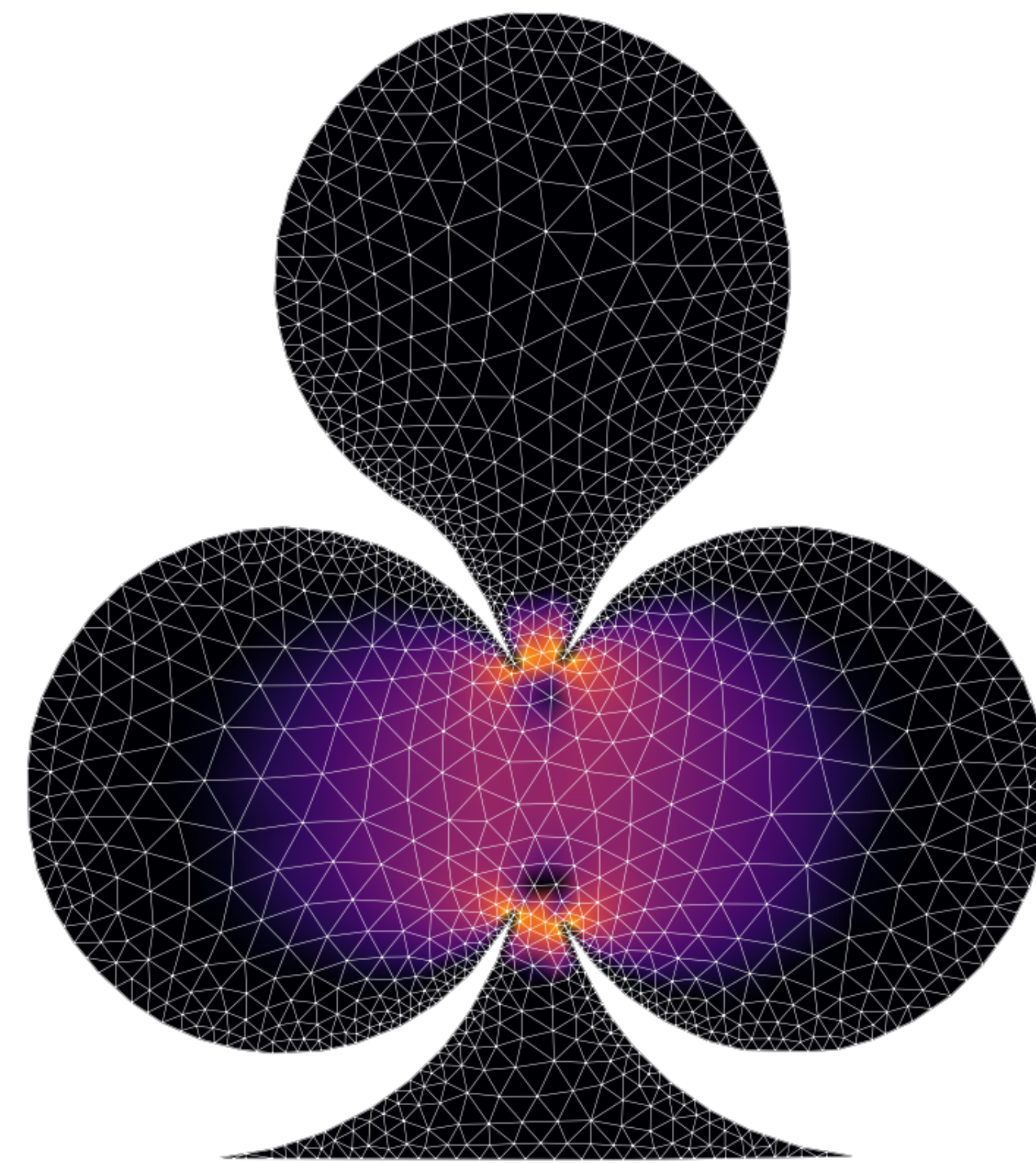
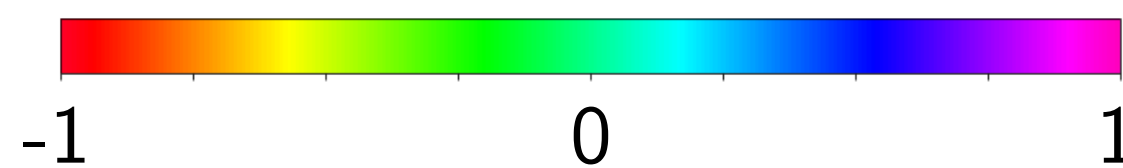




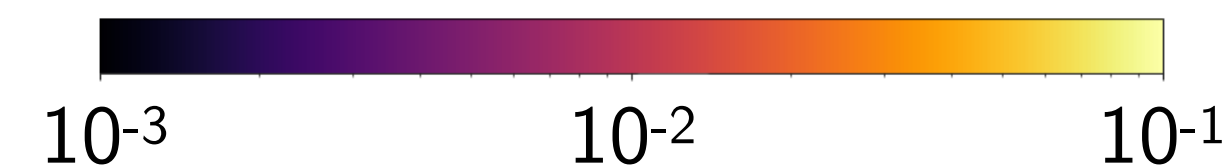
Voronoi cells in club
shape after Lloyd's
algorithm
(with weighting)



Delaunay mesh and
reference FEM solution
to Laplace equation

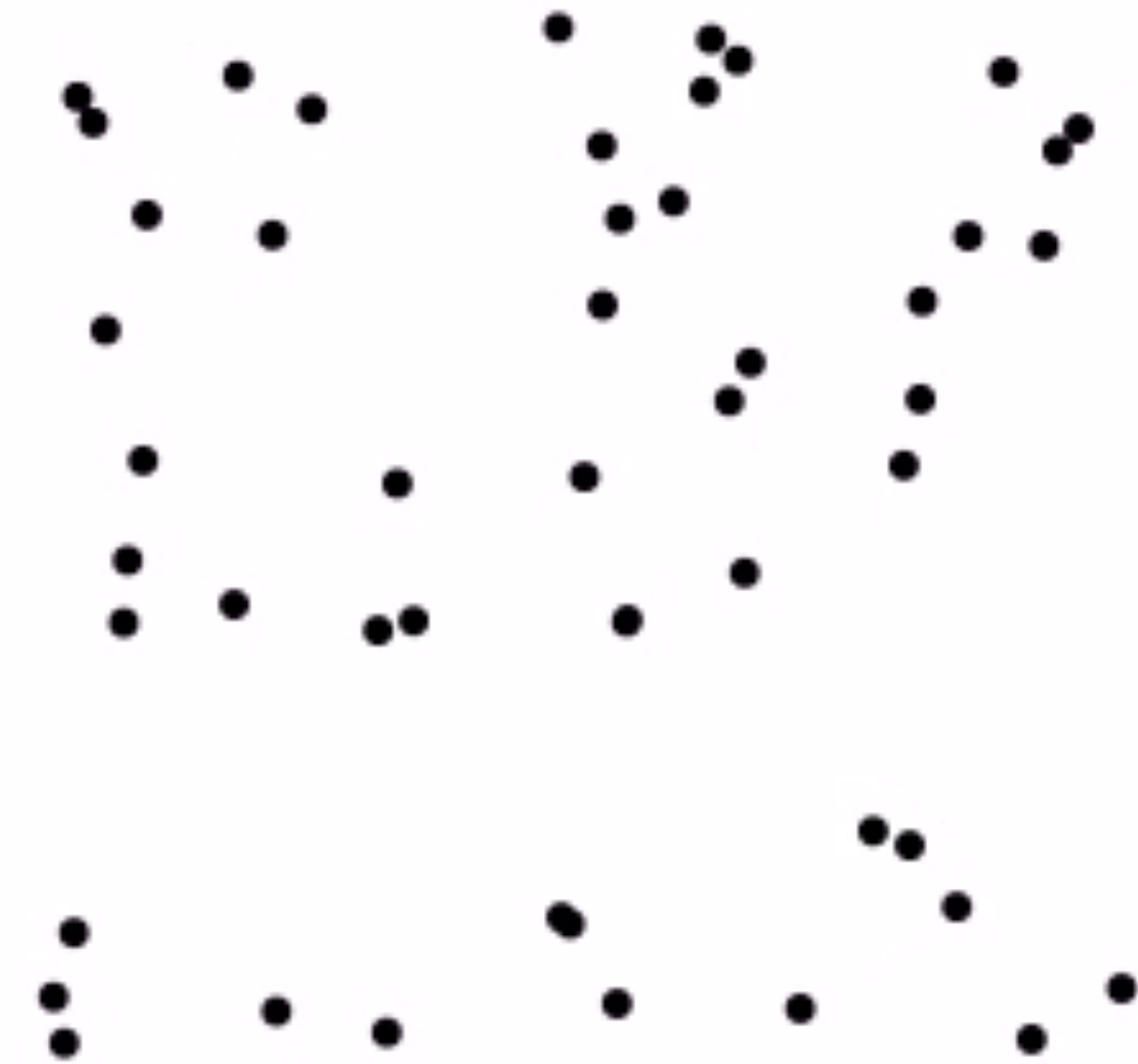


Error between FEM
solution and exact
solution



Alternative meshing approach: DistMesh

- An alternative mesh generation method to incorporate repulsive springs along each Delaunay edge
- Implemented in the DistMesh software package
- In our tests, DistMesh has better performance but has slightly lower mesh quality*



DistMesh examples
(P.-O. Persson)

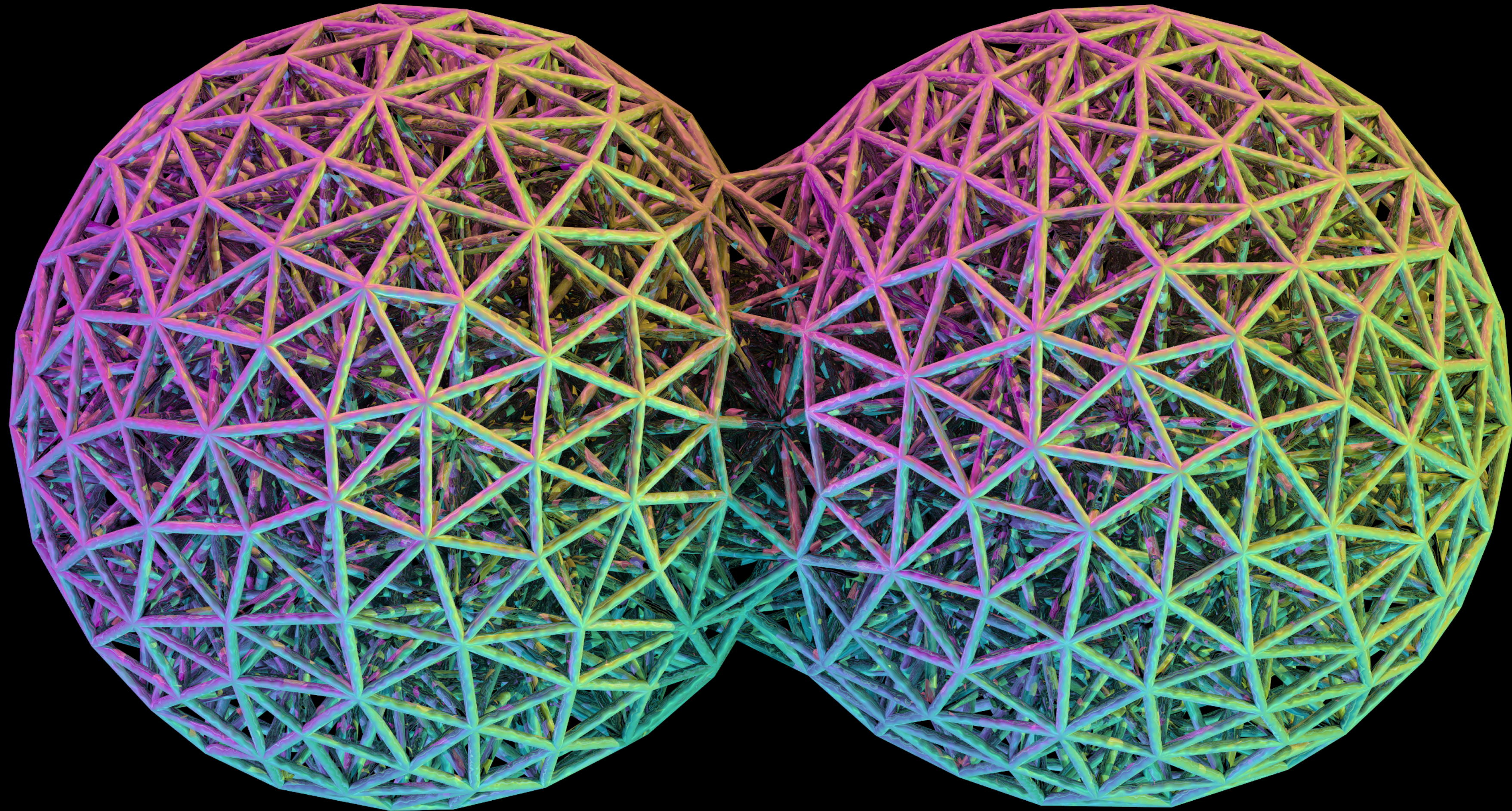
<https://persson.berkeley.edu/distmesh/>

P.-O. Persson and G. Strang, *A simple mesh generator in MATLAB*, SIAM Review **46**, 329–345 (2005).

* *Mesh quality* is a technical definition—see references. High mesh quality implies more triangles close to being equilateral.

Three-dimensional domains

Using TriMe++ (TRlangular MEshing)



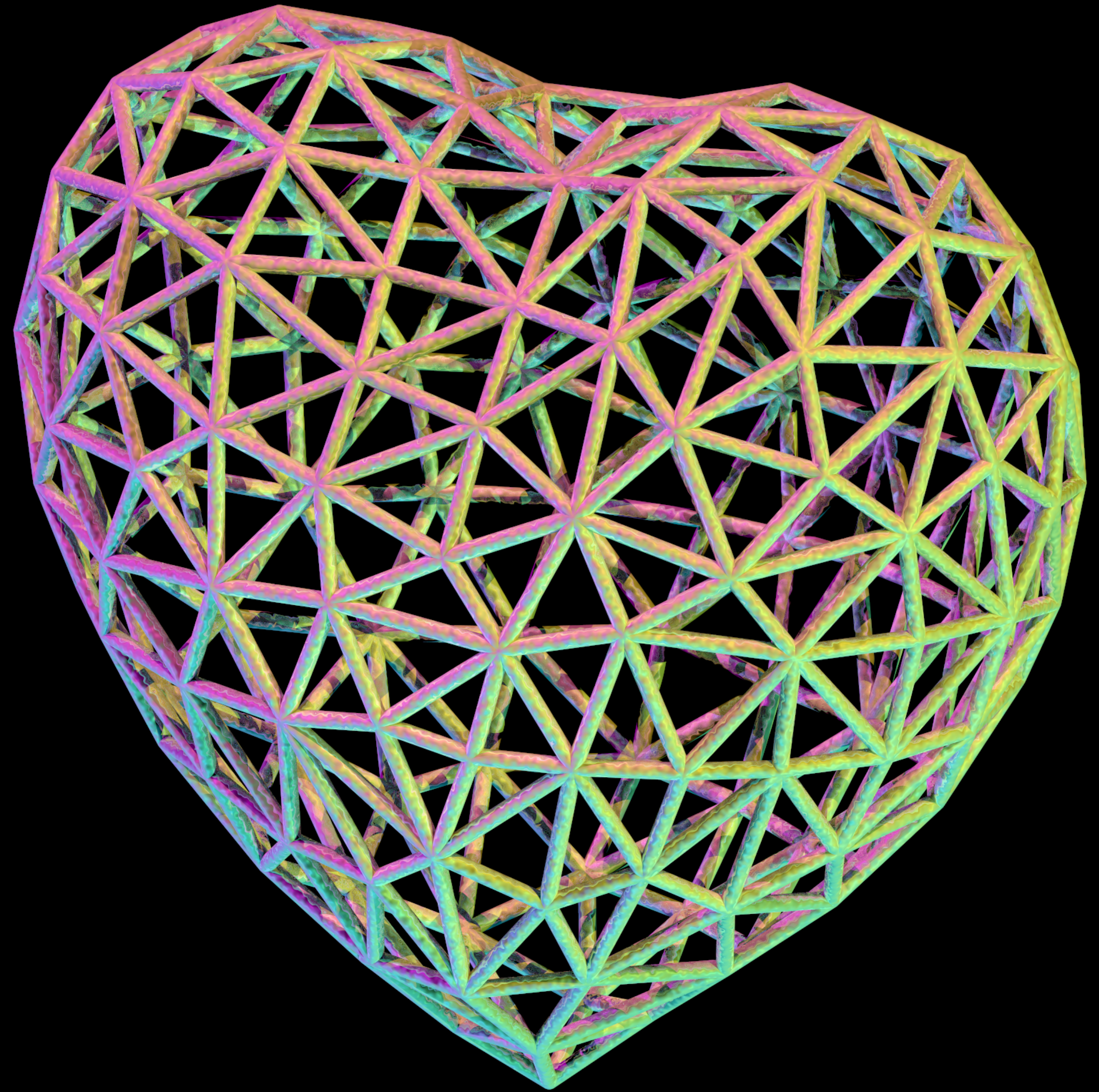
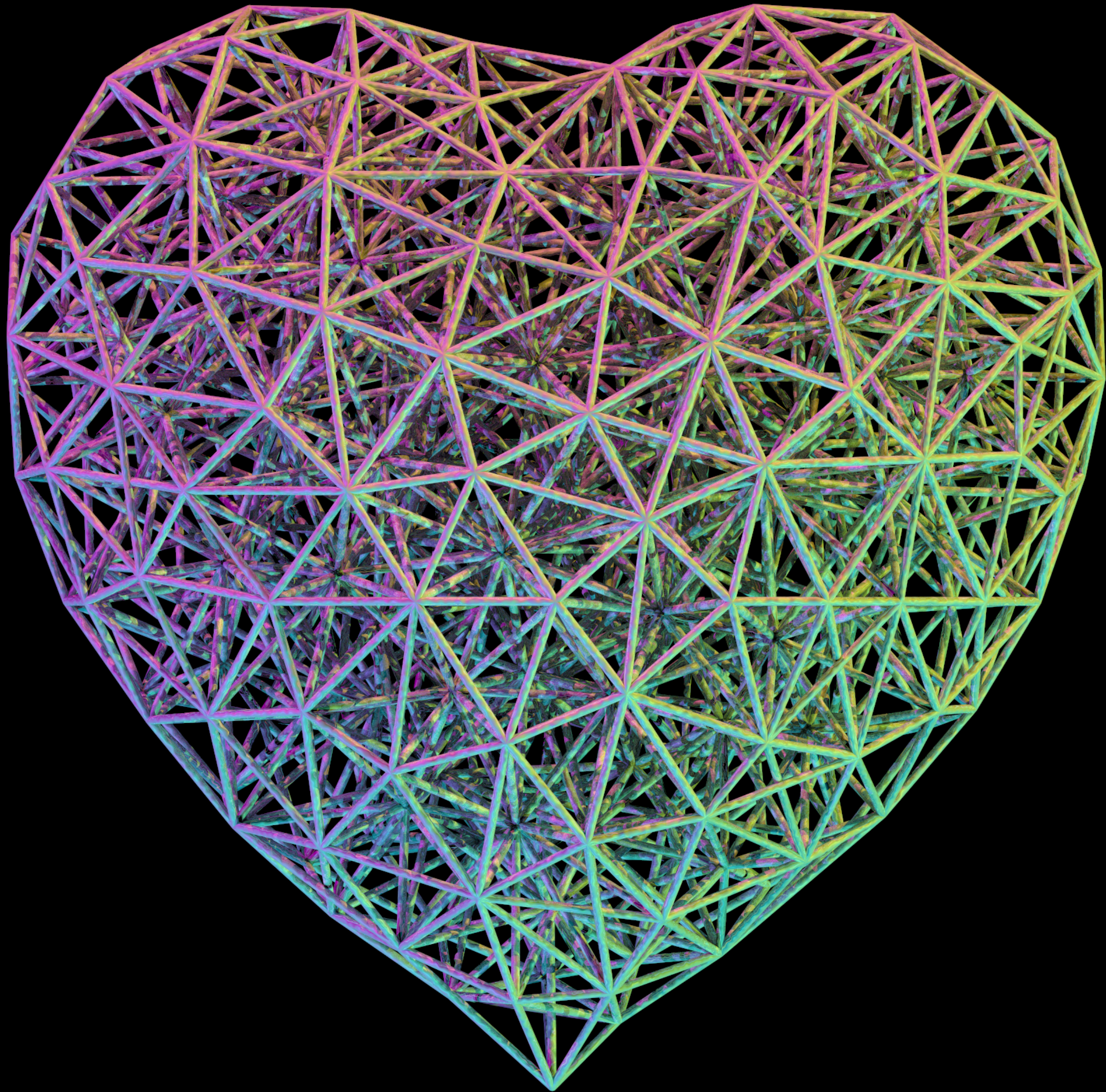
Jiayin Lu

花生米

(Using a combination of Lloyd's algorithm and the DistMesh algorithm)

J. Lu et al., *An extension to Voro++ for multithreaded computation of Voronoi cells*, Comput. Phys. Commun. **291**, 108832 (2023).

J. Lu and C. H. Rycroft, *TriMe++: Multithreaded triangular meshing in two dimensions*, Comput. Phys. Commun. **308**, 109442 (2025).



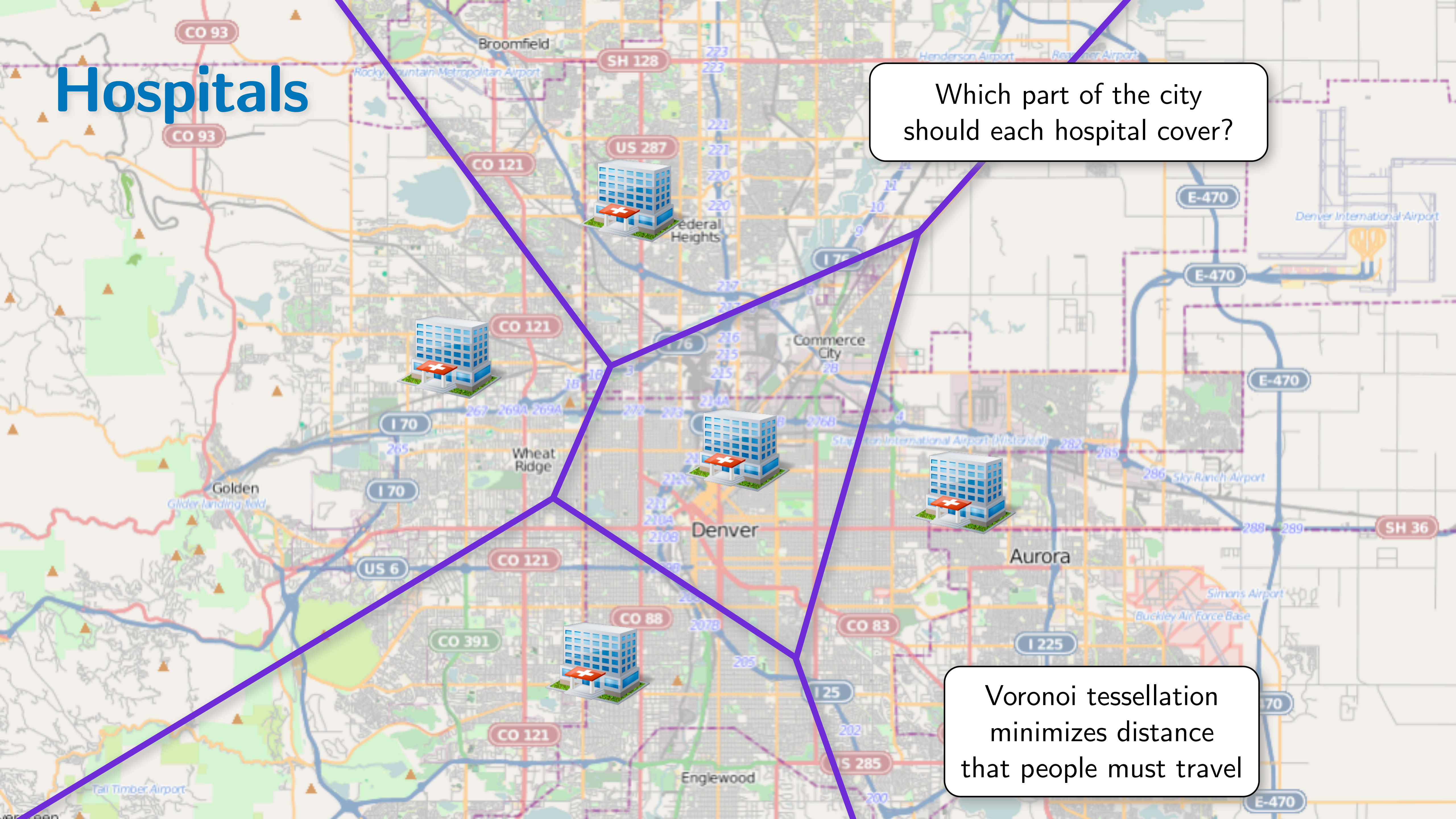
Applications of Voro++

(Extra material not covered in lecture)

Hospitals

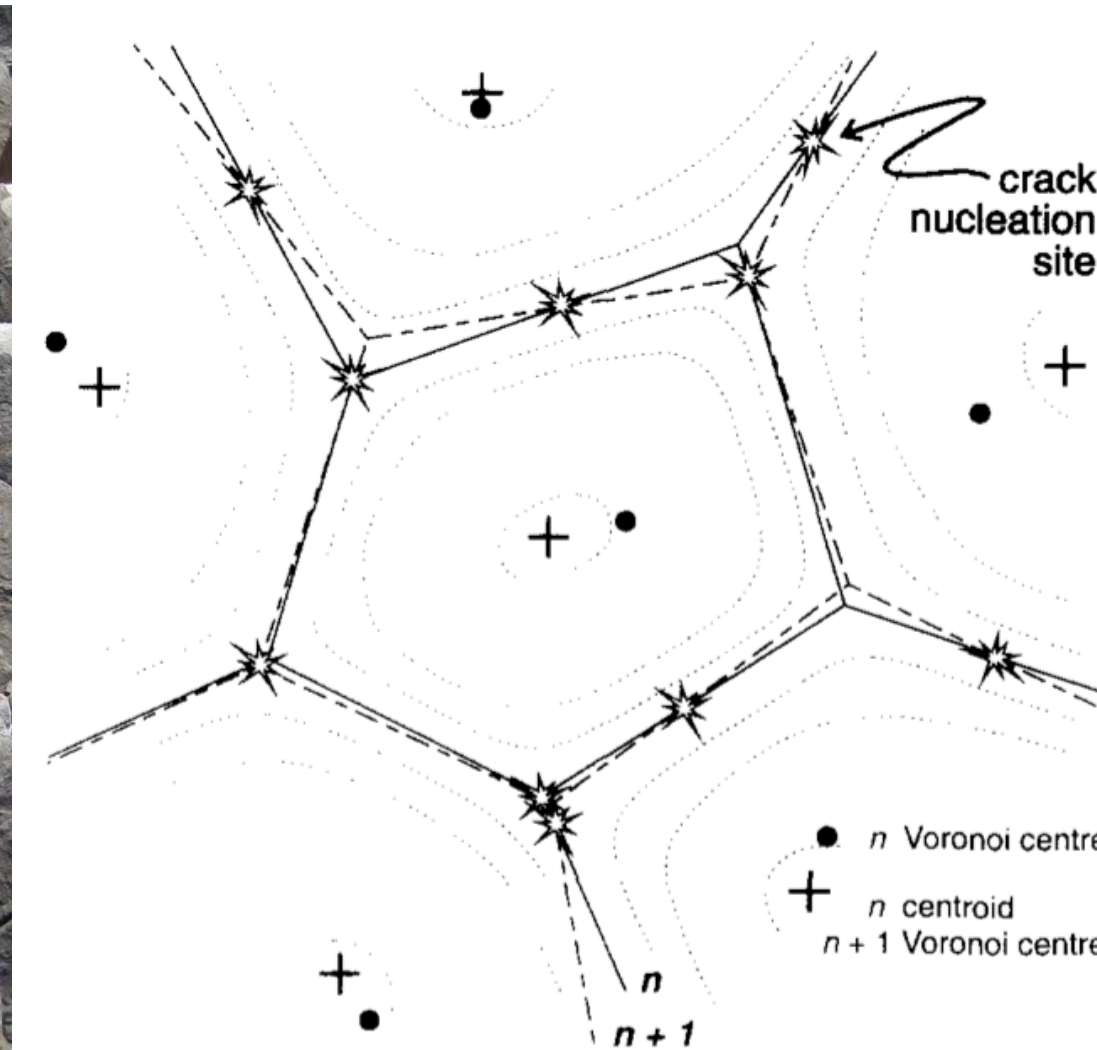
Which part of the city
should each hospital cover?

Voronoi tessellation
minimizes distance
that people must travel



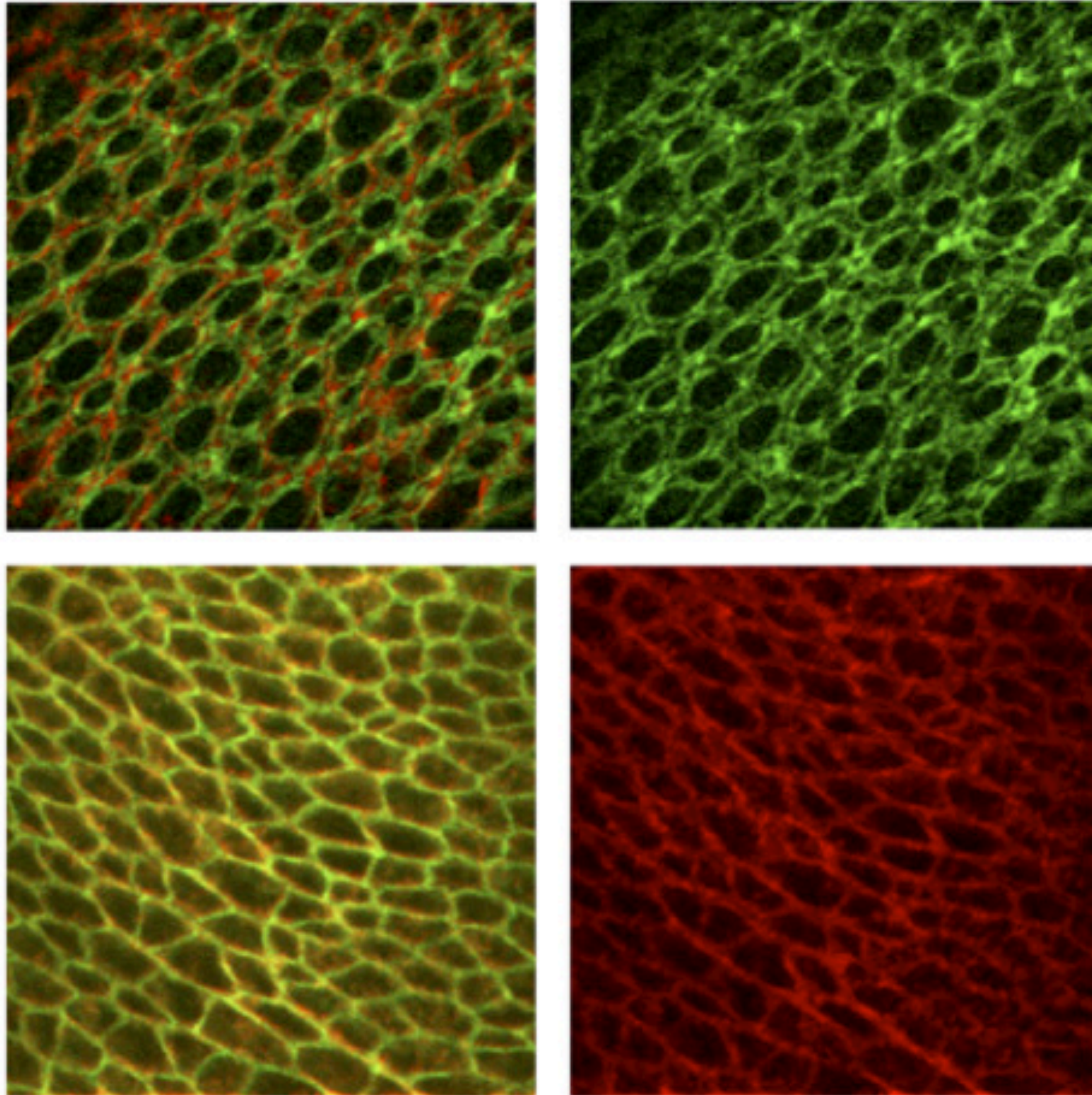
Basalt column formation

(at Giant's Causeway, Northern Ireland)



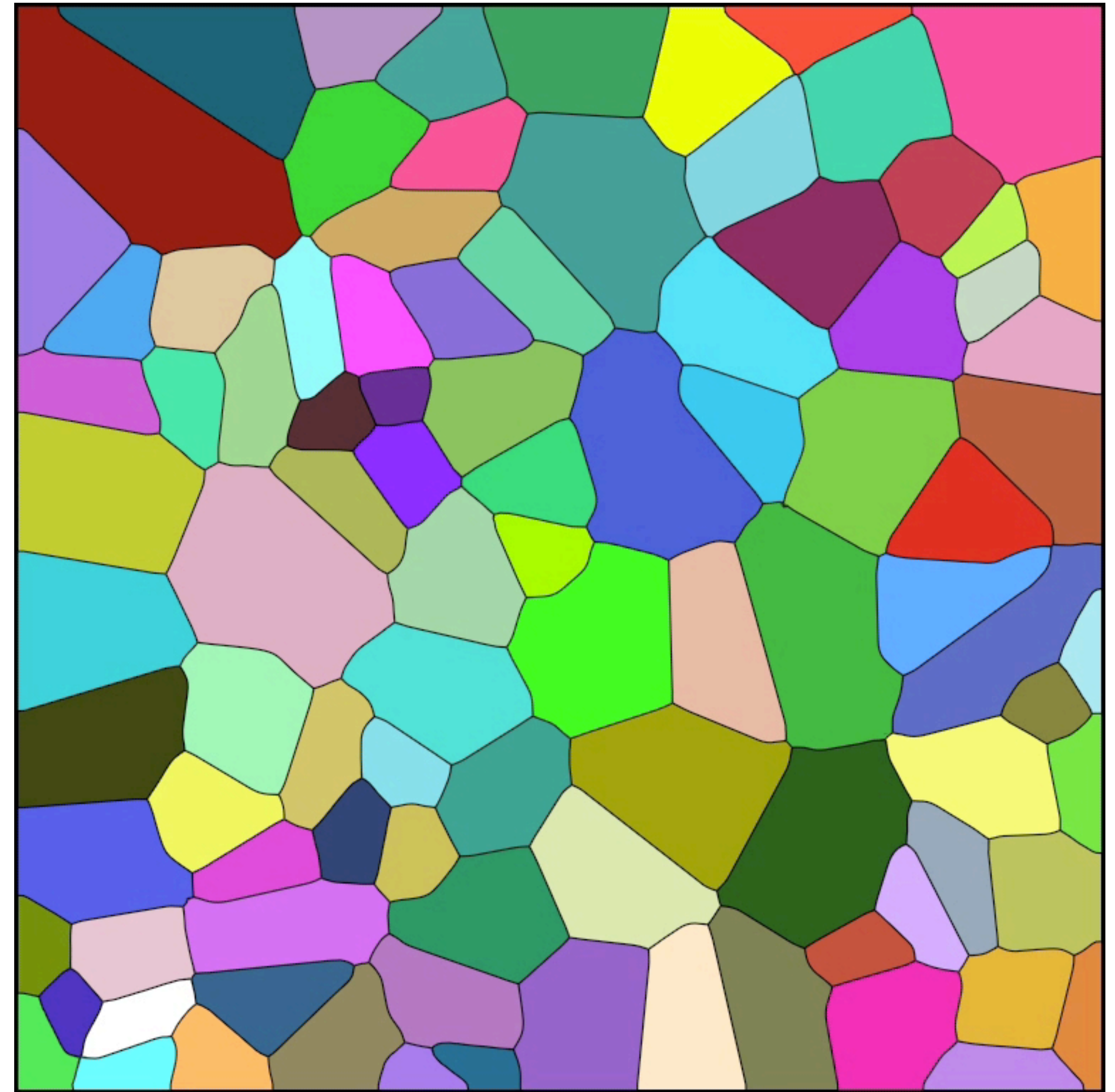
- Vertical cracks form during rapid cooling of lava
- As they extend upwards, the cracked regions become shaped more like Voronoi cells

Patterns in cell boundaries



Cell membranes in a fruit fly (*Drosophila*) wing

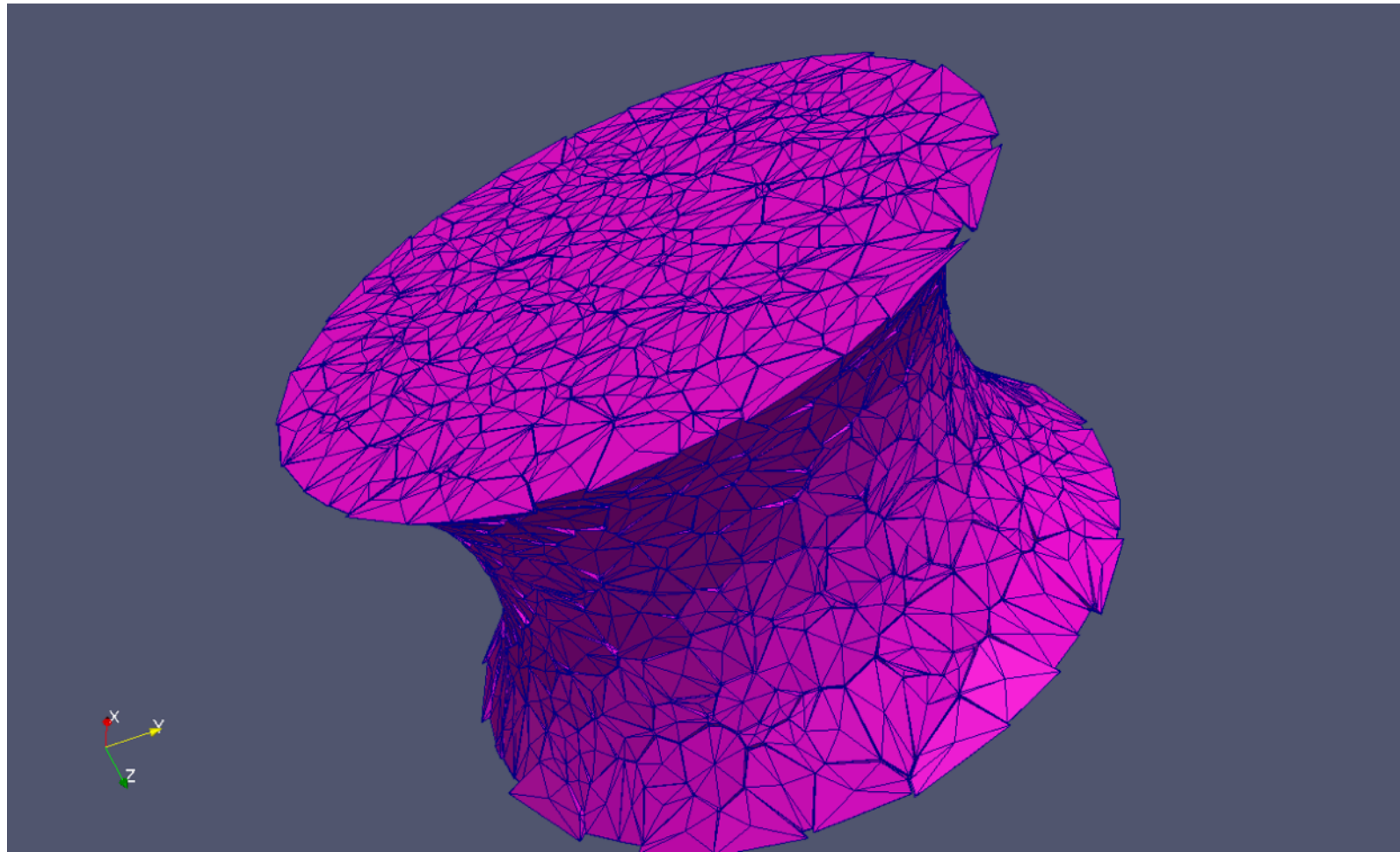
R. Farhadifar, Current Biology **17**, 2095-2104 (2007).



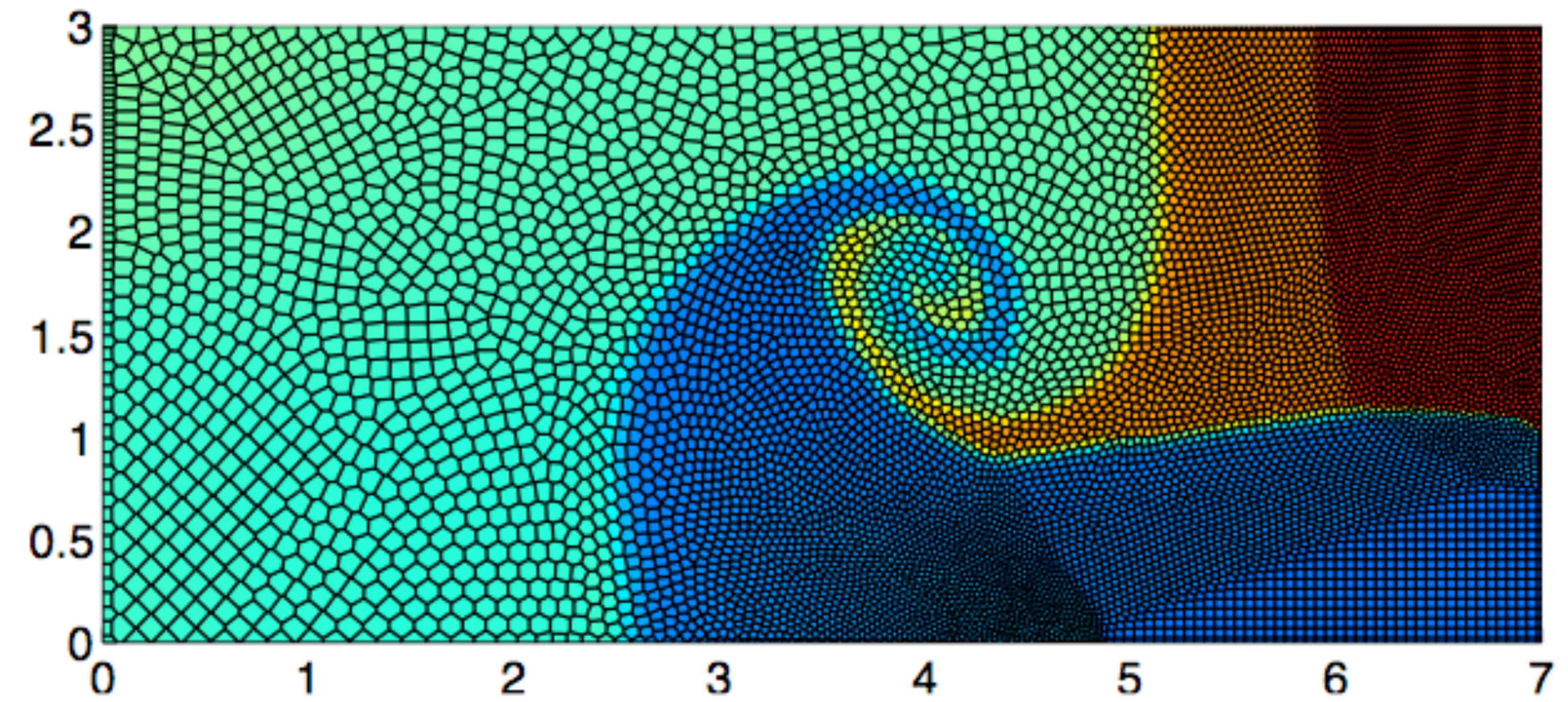
Simulation of a 2D fluid evolving under surface tension

R. Saye and J. Sethian, Proc. Natl. Acad. Sci. **105**, 907-911 (2011).

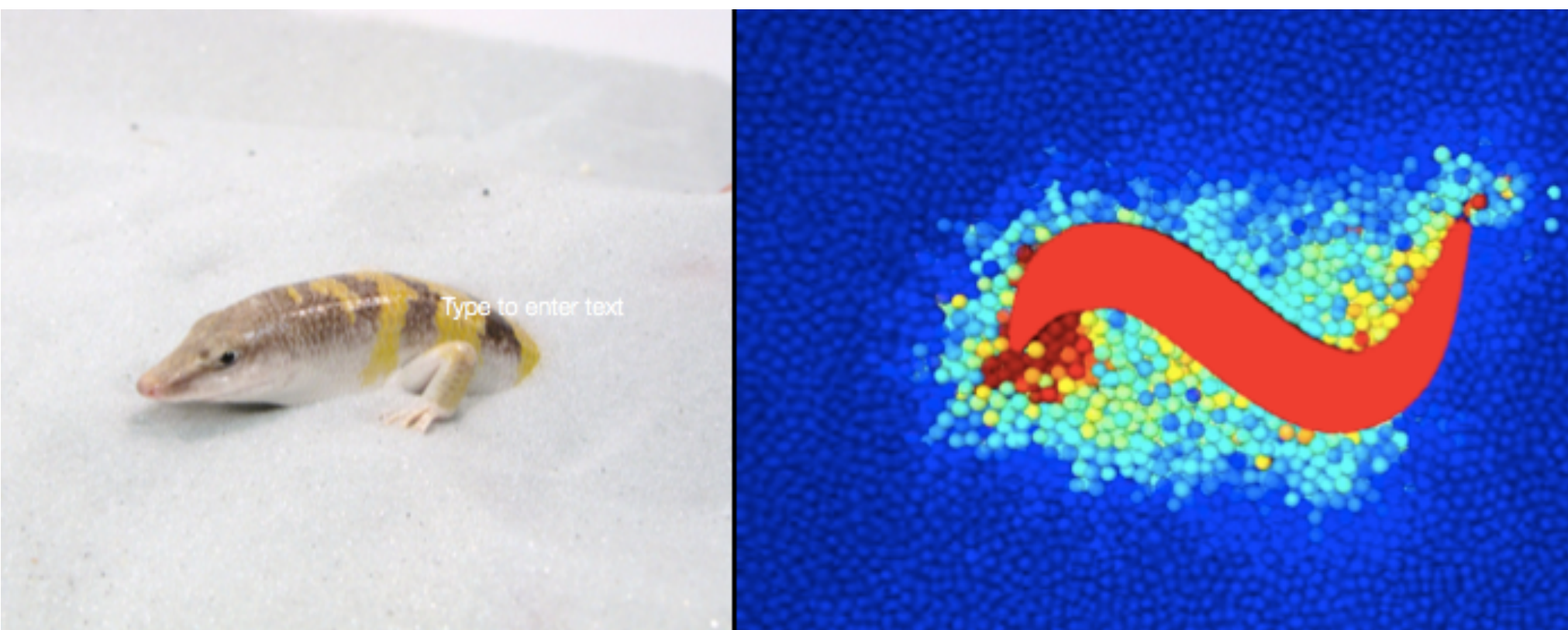
Applications of Voro++



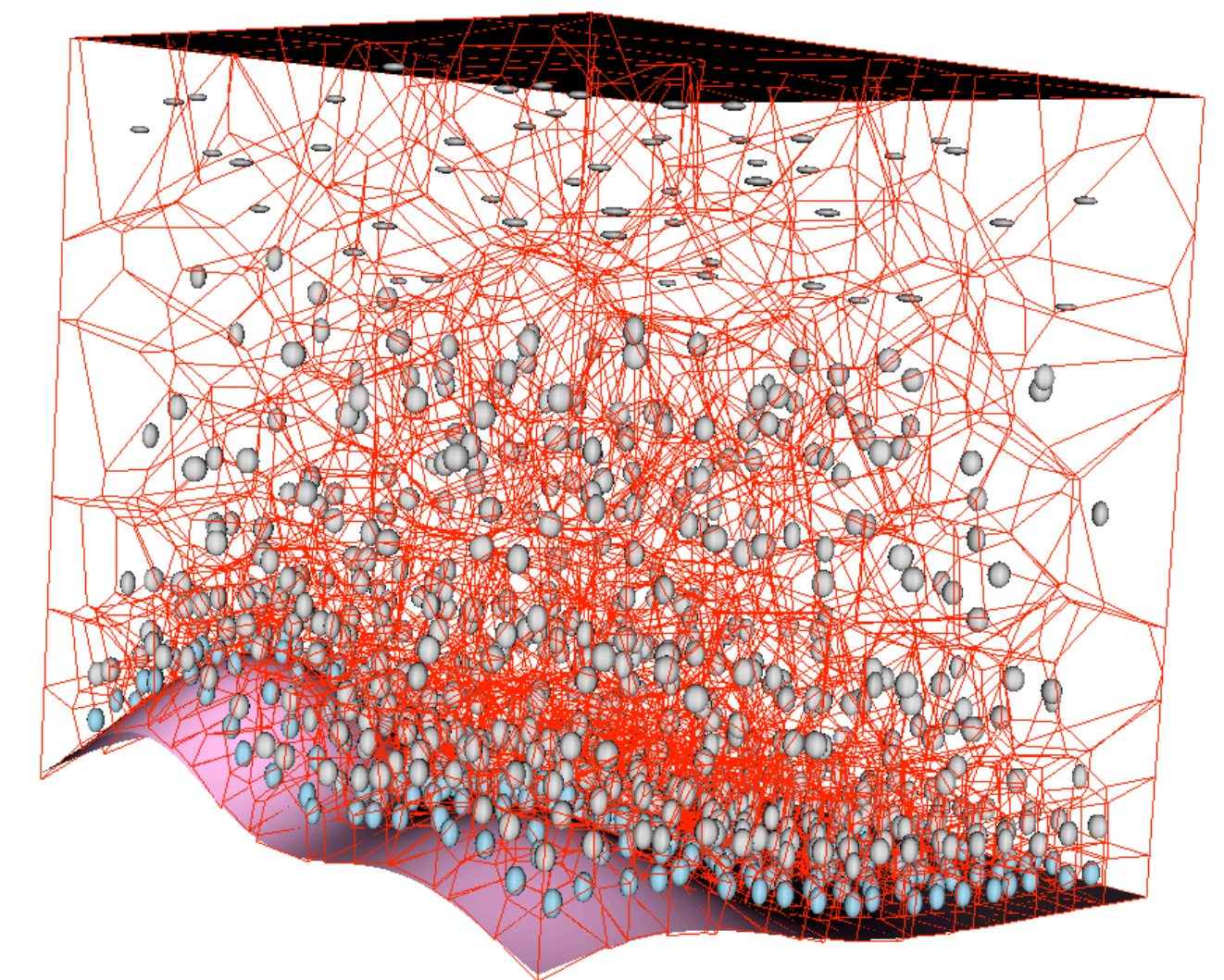
Solidification processes
(*André Phillion, UBC*)



Fluid simulation based upon Voronoi cells
(*M. Shashkov, Los Alamos National Lab*)



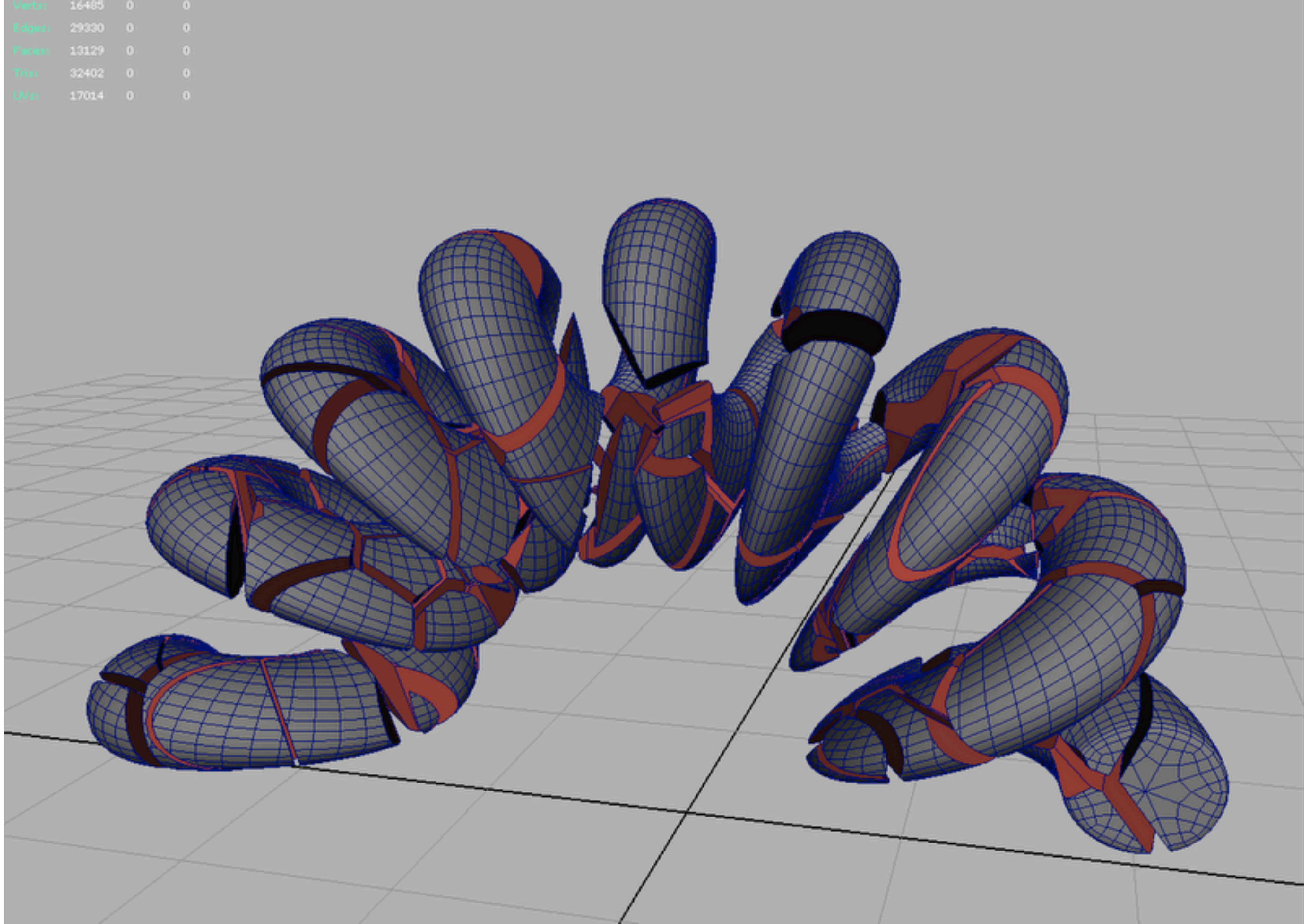
Mechanics of sand swimmers
(*Goldman group, Georgia Tech*)



3D biological cell boundary modeling
(*M. Guillaud & L. Fenelon, BC Cancer Research Center*)

Computer graphics of object shattering

(by Dave Greenwood)



Computer graphics of object shattering

(Using Xplode plugin for Cinema4D, by Manuel Magalhaes)



Voronoi-based sculptures

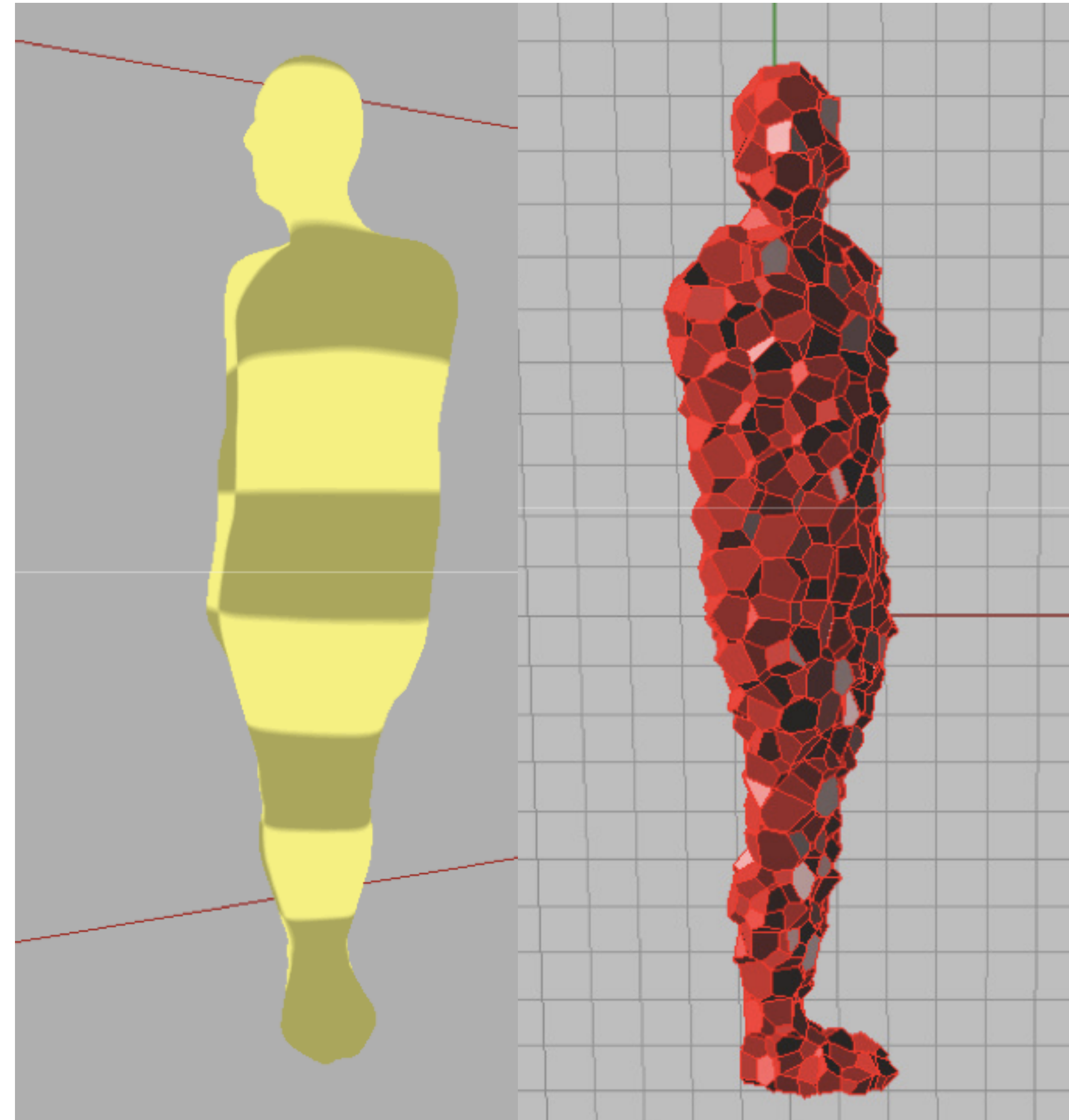
*The Angel of the North,
by Antony Gormley (1998)
(Near Newcastle, England)*



Another Place (2005)



Event Horizon (2010)



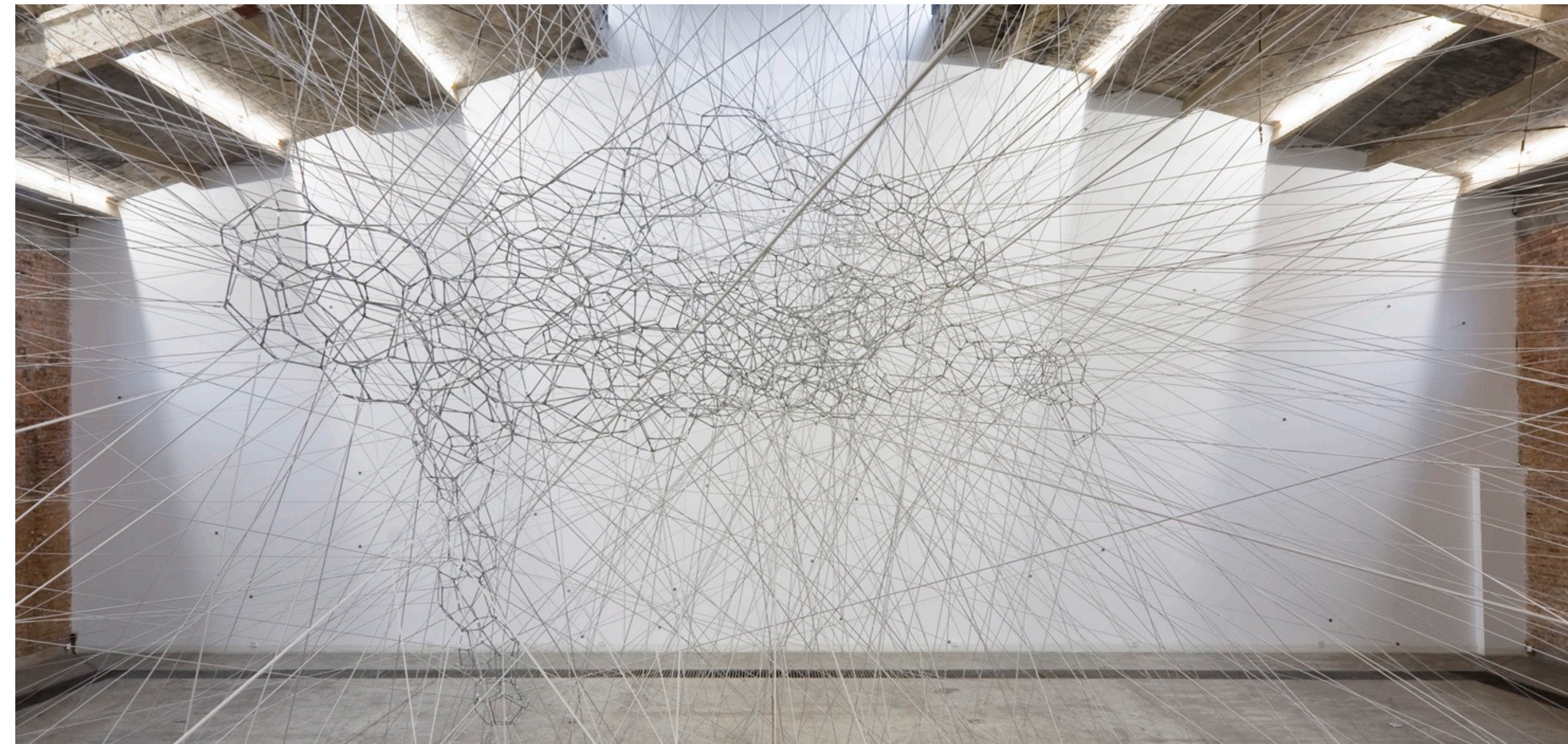
Voronoi-based sculpture
(T. Simmonds, Antony Gormley studio, London)

Sculptures based on Voronoi cells

(from www.antonygormley.com)



Another Singularity (China), 2009



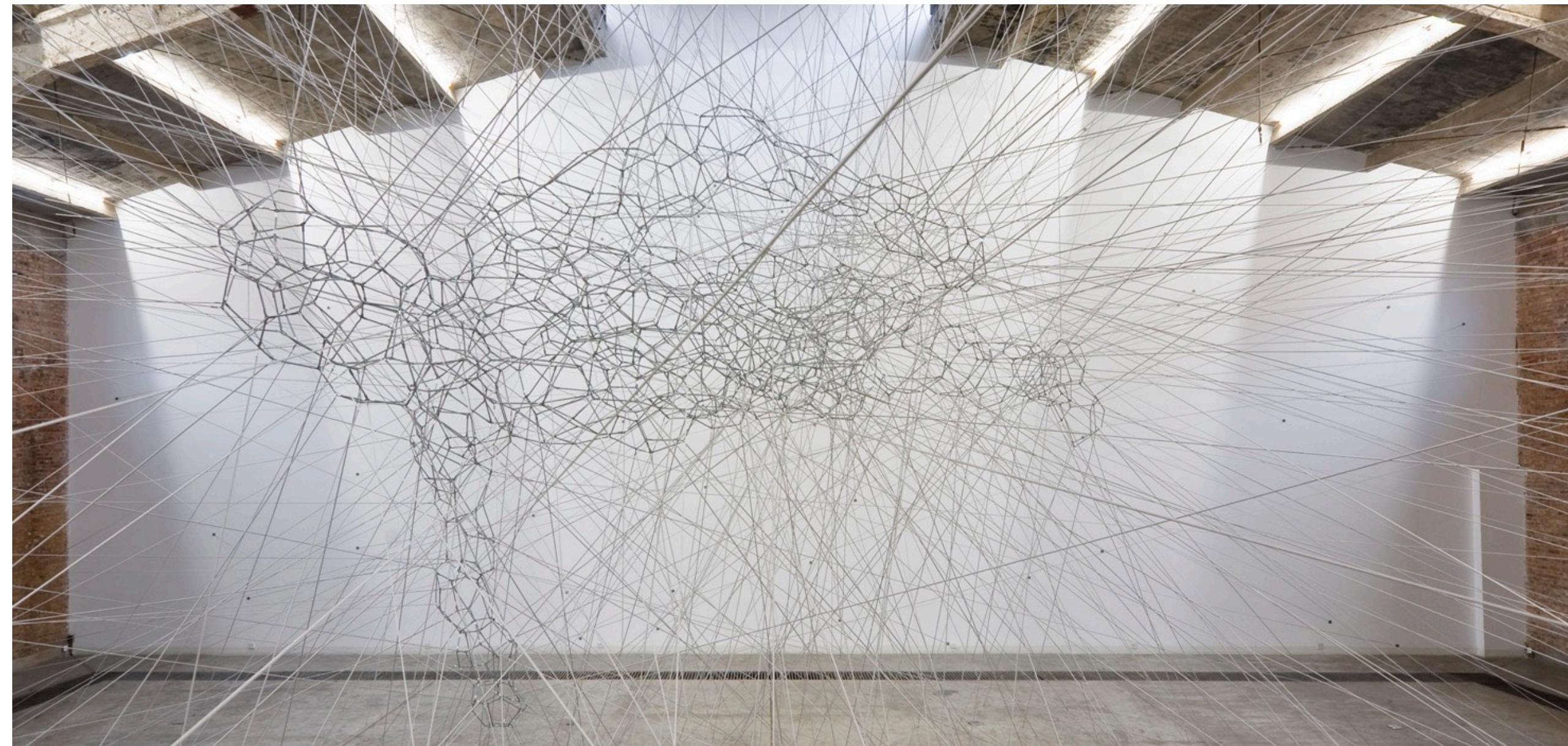
Fuse, 2011



Chord (MIT Mathematics), 2016

Sculptures based on Voronoi cells

(from www.antonygormley.com)



Fuse, 2011



Chord (MIT Mathematics), 2016



Chord (MIT Mathematics), 2016



Chord (MIT Mathematics), 2016