Efficient nearest neighbor searching on the sphere for computing Voronoi diagrams

Philip Caplan*

Abstract

Power diagrams (a generalization of Voronoi diagrams) can be used to perform particle-based fluid simulations of the atmosphere by solving a semi-discrete optimal transport problem at each time step of the simulation. Such an approach requires calculating several spherical Voronoi diagrams at each time step, so this calculation must be as efficient as possible. Voronoi diagrams can be efficiently computed by iteratively intersecting each cell with the halfspaces defined between a site and each of its nearest neighbors until the radius of security is reached. Computing these nearest neighbors for points on the sphere with a kd-tree is slower compared to using a kd-tree for points distributed in a cube. Alternatively, once an initial Voronoi diagram is obtained, the dual Delaunay edges incident to each site can be traversed in a breadth-first manner to compute nearest neighbors for subsequent Voronoi diagram calculations. Another idea employs a Spherical Quadtree to find the triangles containing each site at successive subdivisions of an octahedron, which are then used to find nearest neighbors. The performance of all three nearest neighbor approaches is first analyzed, and then the resulting Voronoi diagrams are used to perform particle-based fluid simulations on the sphere.

References

[1] T. O. Gallouët and Q. Mérigot, *A Lagrangian scheme à la Brenier for the Incompressible Euler Equations*, Foundations of Computational Mathematics, 18 (2017), pp. 835 – 865.

[2] P. Z. Kunszt, A. S. Szalay, and A. R. Thakar, *The Hierarchical Triangular Mesh*, in Mining the Sky, 2001, pp. 631–637.

[3] B. Lévy and N. Bonneel, *Variational Anisotropic Surface Meshing with Voronoi Parallel Linear Enumeration*, in Proceedings of the 21st International Meshing Roundtable, 2013, pp. 349–366.

Supplementary Material

This section elaborates upon the methods and results described in the abstract. Experiments were performed using a 10-core 2021 MacBook M1 Pro.

*Middlebury College Department of Computer Science

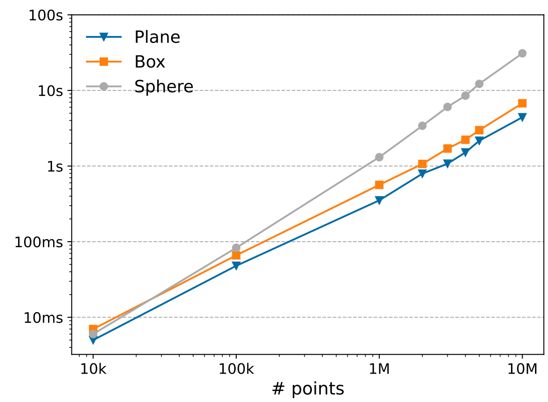

Figure 1: Time to compute the 50 nearest neighbors to each site using a kd-tree when the points are randomly distributed in a plane, a box, or on a sphere.

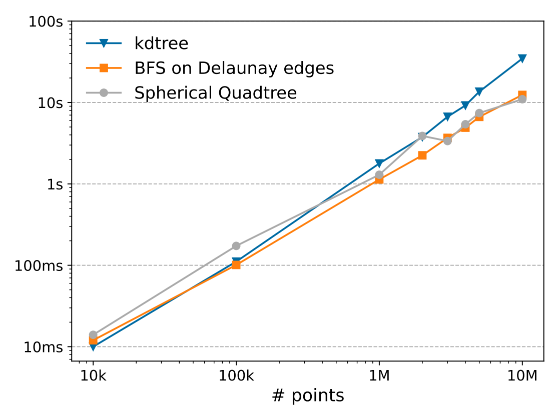

Figure 2: Total time to compute nearest neighbors and calculate a Voronoi diagram using the three nearest neighbor approaches investigated in this work. The time reported for the Delaunay Breadth-First Search (BFS)-based method is the time to compute nearest neighbors *after* the initial Voronoi diagram is calculated.

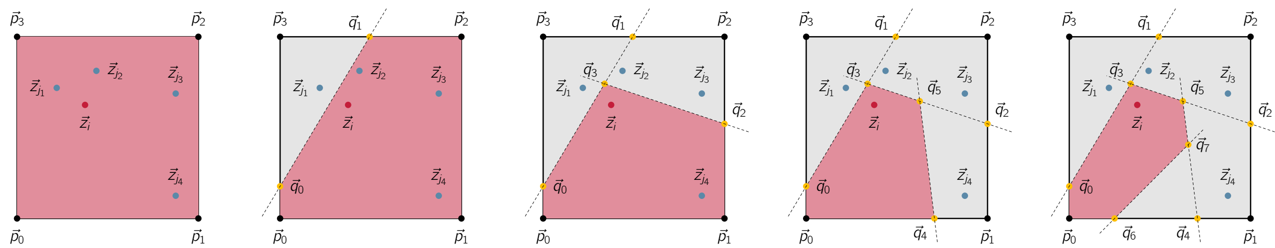

Figure 3: Each Voronoi cell is iteratively calculated by intersecting the current polygon with a plane defined between the site $\vec{z}_i$ and the next nearest neighbor $\vec{z}_j$. Clipping stops when the radius of security is reached [3].

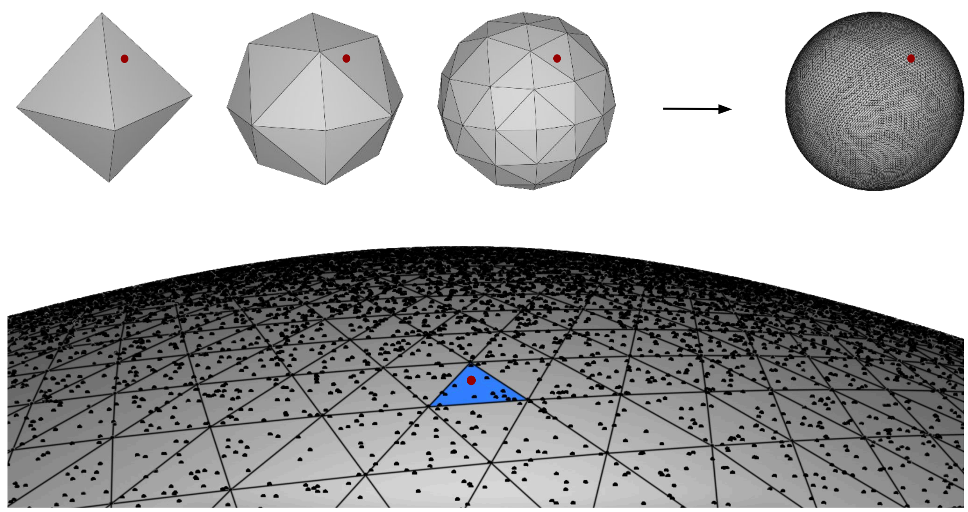

Figure 4: (*top*) The containing triangle of each site (in red) is found within each subdivision of an octahedron, effectively searching through a Spherical Quadtree [2] until the finest subdivision is reached. (*bottom*) To obtain the nearest neighbors of a query point (in red), points in the blue triangle and any triangles in the one-ring of its vertices are accumulated and then sorted by distance to the query point.

Figure 5: Particle-based simulation of an incompressible fluid on a rotating sphere using the Gallouët-Mérigot scheme [1]. The complete simulation can be viewed here.