Voronoi Diagrams on Spheres

Sam Wood

Voronoi diagrams, like maps, can be used to visually represent data. My longtime love of maps combined with my recent interest in Voronoi diagrams encouraged me to find a project integrating both of these interests. Initially, I planned to focus my project on data visualization using Voronoi diagrams. However, the difficulty I had in coding caused me to change the focus of my project to creating Voronoi diagrams on the surface of a sphere.

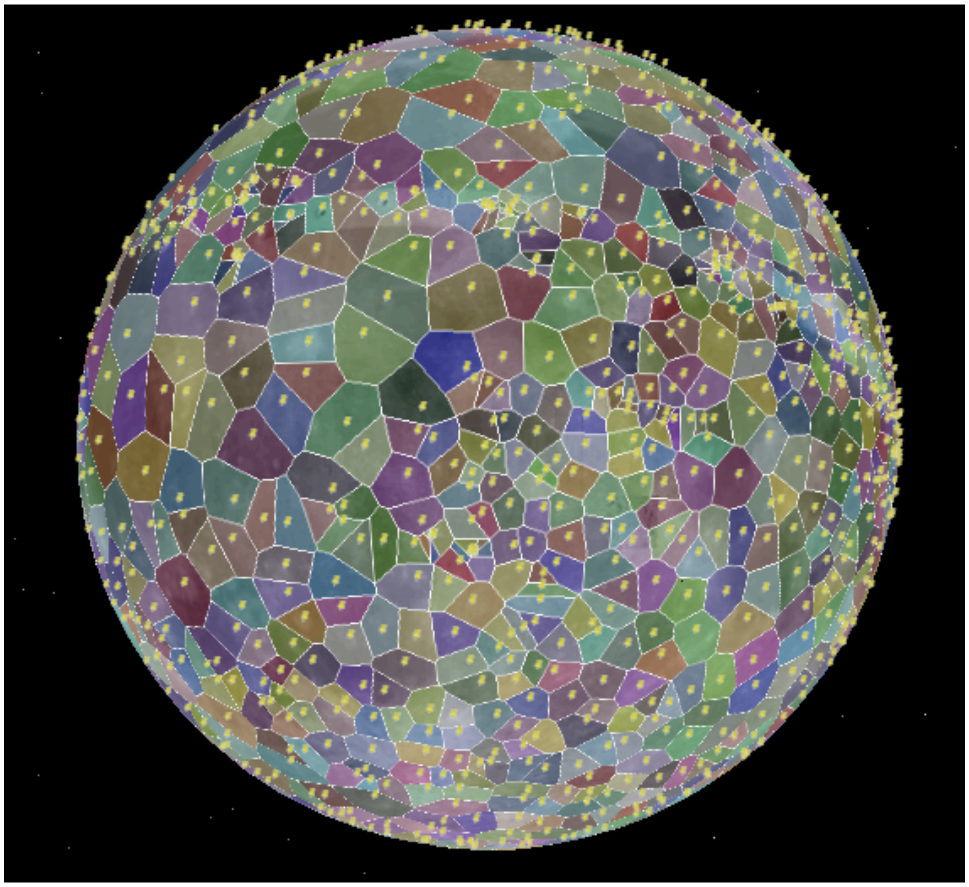

Figure 1: Voronoi diagram of the Moon.

1 Creating a Voronoi diagram on the surface of a sphere

1.1 Difficulties

Initially, I thought it would be trivial to create Voronoi diagrams on the surface of the earth, by creating a 2D Voronoi diagram using latitude and longitude and then fitting it to the earth. However, I realized that this does not work because distance is measured differently on the surface of a sphere and it would be very complicated to ensure that the Voronoi diagram wraps around the edges of the map properly. Thus, in order to create a global Voronoi diagram, I needed to create it on the surface of a sphere. I had little luck in searching for an algorithm or implementation for this and decided to design and implement my own algorithm.

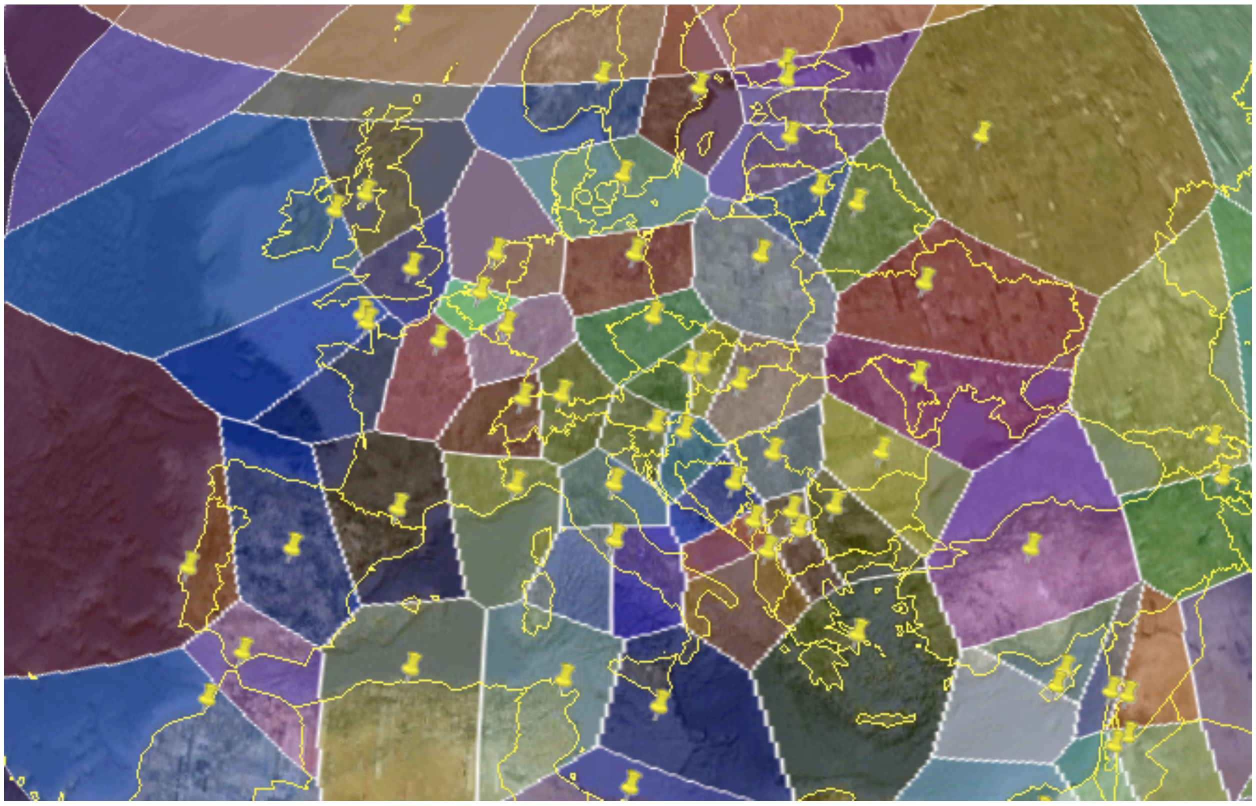

Figure 2: Voronoi diagram of Europe

1.2 Algorithm

The initial insight for my algorithm came from [2], which suggested that the convex hull of a set of points on a sphere forms the Delaunay triangulation on the sphere. (If more than three points are co-circular, it is technically not a Delaunay triangulation but can still be used to compute the Voronoi diagram.) From this I was able to develop my Voronoi diagram algorithm. The algorithm uses CGAL's 3D incremental convex hull algorithm to compute the convex hull of a set of points [4]. The edges of the convex hull form the Delaunay edges.

Next, the algorithm iterates through the Delaunay edges and computes the dual of each edge. For each Delaunay edge, the dual edge is the segment whose endpoints are the circumcenters of the two faces that share the Delaunay edge. I developed this algorithm myself, using CGAL's function to find the circumcenter of points in 3D [4]. This part of my code generates the Voronoi edges, but does not associate them with any sites or other edges. The output is a series of edges which are listed in no particular order. For all the difficulty I had in understanding CGAL and developing the algorithm, the code is quite simple and concise.

1.3 Python and KML additions

To view the Voronoi diagrams I created using CGAL, I translated my data to KML [10] using Python. In the process, I associated Voronoi edges with sites, creating actual polygonal Voronoi regions. (The polygons had not been created in my C++ code.) This allowed me to create polygons in KML and associate names with them.

2 Diagrams

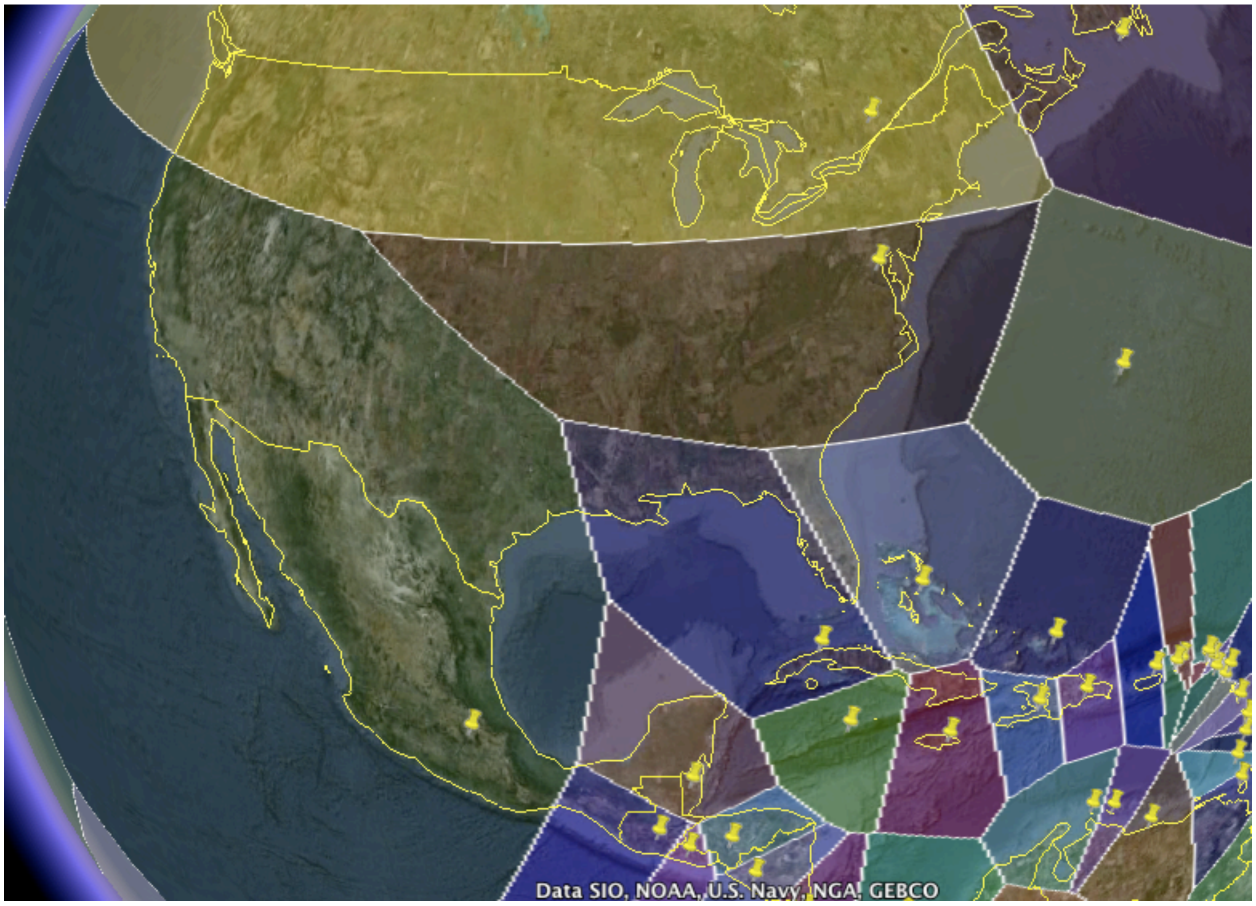

Figure 3: Voronoi diagram of the US.

I generated Voronoi diagrams based on three data sets: world capitals [3], lunar craters [7], and craters on mars [7]. The first Voronoi diagram (capital-world.kml) redraws political boundaries by creating a Voronoi diagram with national capitals as the sites. Thus, every point within a capital's Voronoi region is closer to its capital than to any other capital. In addition, the oceans and all bodies of water are included in Voronoi regions, eliminating international waters and allocating waters to different countries. The Delaunay triangulation is also included (capital-delaunay.kml).

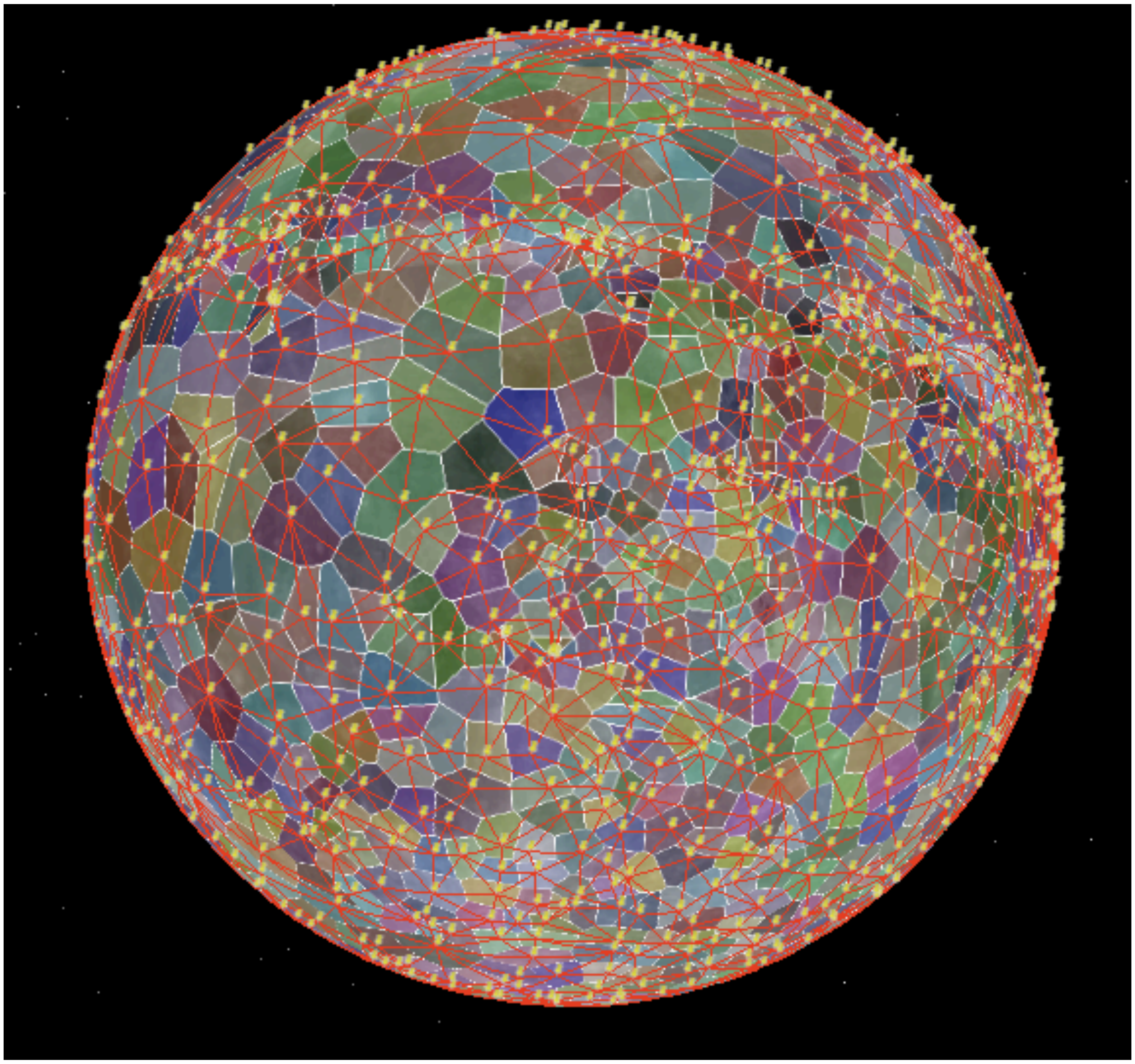

Figure 4: Voronoi diagram and Delaunay triangulation of the Moon.

The second diagram draws the Voronoi regions of craters on the Moon (moon-craters.kml). If we are to colonize the Moon in the future, my Voronoi diagram creates a logical and practical political division of the Moon. Naturally, civilizations would develop in lunar craters, much like they develop in valleys on earth. Therefore, having these craters as the Voronoi sites is the most sensible political division. In addition, I have included the Delaunay triangulation of the Moon (moon-delaunay.kml). With population centers located at lunar craters, the Delaunay triangulation forms an ideal roadway system, connecting each crater to its nearest craters.

The case of Mars is very similar to the Moon. Mars has a different coordinate system than the Earth and Moon. Google Earth's coordinate system for Mars doesn't line up with the edges I computed.

3 Tools

The main tools I used to create and represent the Voronoi diagrams were CGAL [4] and Google Earth [8]. There is a known bug in Google Earth that prevents polygons from crossing the International Dateline. Consequently, the Voronoi regions that cross the International Dateline wrap all the way around the earth. To fix this, I attempted to find the intersection of the regions with the International Dateline and cut the polygons in two. This is not a particularly pretty solution, but it is mostly successful. There are still some issues with it, particularly around the poles, but the major part of the world is fine.

4 Special Thanks

Special thanks to my brother, Ben Wood, for moral support, helpful unix commands, and data and language details.

References

[1] *Algorithmic Foundations of Geographic Information Systems.* Editors: Marc van Kreveld, Jurg Nievergelt, Thomas Roos, and Peter Widmayer. New York: Springer, 2000.

[2] an online forum that I stumbled upon and have been unable to relocate

[3] "CIA - The World Factbook", `https://www.cia.gov/library/publications/the-world-factbook/fields/2057.html`, 2009.

[4] "Computational Geometry Algorithms Library", `http://www.cgal.org/`.

[5] Dale, Peter. *Introduction to Mathematical Techniques used in GIS.* New York: CRC Press, 2005.

[6] de Berg, Mark, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications.* Third Edition. The Netherlands: Springer, 2008.

[7] "Gazetteer of Planetary Nomenclature", USGS Astrogeology Research Program, `http://planetarynames.wr.usgs.gov/`, 2009.

[8] Google Earth, `http://earth.google.com/`, 2009.

[9] *The Handbook of Geographic Information Science.* Editors: John P. Wilson and A. Stewart Fotheringham. Malden, MA: Blackwell, 2008.

[10] "KML Reference", `http://code.google.com/apis/kml/documentation/kmlreference.html`, 2009.

[11] Okabe, Atsuyuki, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams.* Second Edition. New York: Wiley, 2000.

[12] O'Rourke, Joseph. *Computational Geometry in C.* Second Edition. New York: Cambridge, 2008.

[13] Various articles. `http://en.wikipedia.org/`.