Arcs on Spheres and Snakes on Planes

David Swart

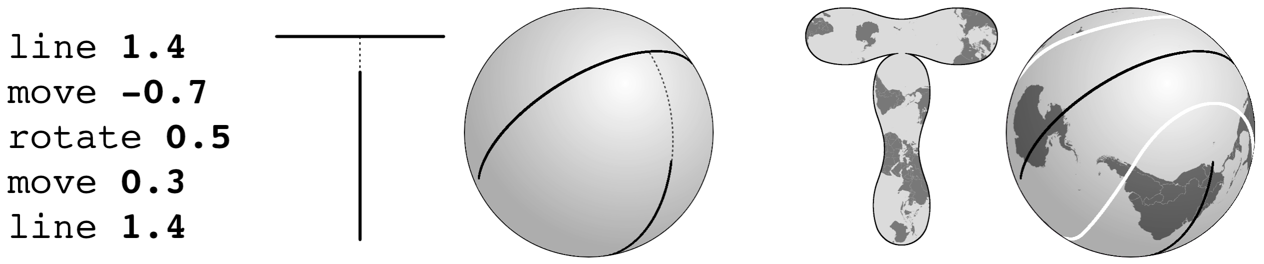
Waterloo, Ontario, Canada; dmswart1@gmail.com

Abstract

An existing method to map points from the sphere to the plane uses a turtle language to specify a ‘skeleton’ which is then used to create artistic map projections. We augment this method by adding a new turtle command to draw non-great-circle arcs. We discuss some technical details and show some early artistic explorations.

Introduction

In 2009, I wrote about some software that enables a user to explore ways to mathematically unwrap spherical imagery to the plane as a projection [4]. The input to the software is a set of turtle commands to move, rotate and draw lines on both surfaces. Two turtles, one on the sphere and one on the plane, draw a set of corresponding segments. The Voronoi regions on the sphere (the set of points that are closest to each drawn segment) are then mapped to their corresponding line segments on the plane. See Figure 1. For the user’s convenience, angles are specified such that one unit is 180° (π radians). For example, `[r 0.5]` is a 90° turn to the left.



```
line 1.4
move -0.7
rotate 0.5
move 0.3
line 1.4
```

Figure 1: *Turtle commands define lines on both a sphere and on a plane which is then used to map content from the sphere to the plane.*

Since then, I have continued to explore the artistic potential of this tool, mapping spherical imagery such as panoramas and the globe into projections that I find interesting and pleasing. In 2021, I added the ability to automatically tweak a turtle program to find a way to unwrap the sphere into a target shape [5]. This paper describes a further addition to this software, namely the ability to instruct the turtles to draw arcs. To be specific, any arcs not just ones which lie along great circles.

The next section is a quick review of conic projections; next, we discuss some technical details about the new arc instruction; finally, we follow with some artistic explorations and possible future projects.

Conic Projections

The new capability of our software is the ability to map portions of the sphere by using a conic projection. There are many good resources that cover the topic including one by Snyder [3], however, we can outline some basic points here.

Conic projections can be thought of as mapping points on a sphere to a cone first, and then unwrapping this cone onto a plane. As a result, the lines of latitude (parallels) will get mapped to concentric circles and the lines of longitude (meridians) are mapped to equally spaced rays, radially oriented line segments. There are three notable types of conic projections distinguished by the spacing between the parallels.

The *Albers conic* projection has parallels spaced out so that the resulting map preserves *area*. The *Lambert conic conformal* preserves *angles* resulting in a conformal map where local shapes look nice. In fact, Statistics Canada sets this projection as the official projection for maps and does a good job keeping the shape of Canada from looking overly distorted. For our purposes, we are most interested in the *equidistant conic* projection (described by Ptolemy in 150 AD!) which preserves *distances* along the meridians and along one or two standard parallels.

Arcs on Spheres

A natural way to specify an arc with a turtle language is with the command `[a` θ r`]` where θ is the sweep angle and r is the arc's radius. However, having two turtles on two different surfaces, we wish to keep the *geodesic curvature* of each arc the same. That is, each turtle will impart the same amount of local curvature as it draws. To keep geodesic curvature the same, we need different θ and r values for the plane than for the sphere. So for our program, we will ask the sphere turtle to use the given values (θ_S and r_S) and the planar turtle will use values $\theta_P = \theta_S \times \cos(r_S)$ and $r_P = \tan(r_S)$ according to properties of an equidistant conic projection with a standard parallel equal to the sphere arc.

Once our corresponding arcs are specified, our software can now map points by copying the specific distance and angle to the closest point of an arc on one surface and then using these values to calculate the corresponding point on the other. See Figure 2.

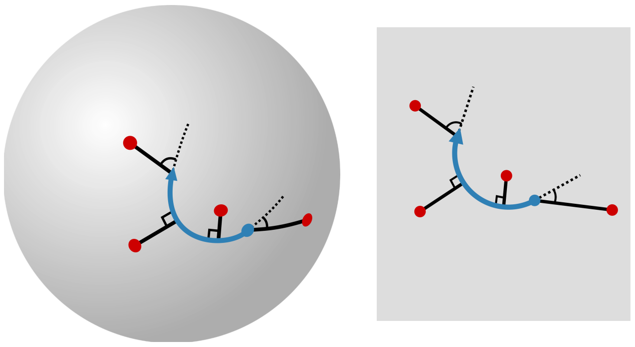


Figure 2: *Points on the sphere (red) are mapped according to their relative position to the nearest arc (blue).*

It is worth taking a moment for a technical description of the resulting projections of our software. Our turtle method maps Voronoi regions piece by piece. If points on the sphere are in the region closest to a line segment then the region is mapped with an equirectangular projection. If the region is closest to a line segments' end point then it is mapped with an azimuthal equidistant projection. And now, regions closest to an arc are mapped with an equidistant conic projection. Thus, taken altogether, the projections we produce can be thought of as a piecewise mix of these three.

Our implementation is written in JavaScript and can be run in a web browser. Readers are encouraged to try this code out for themselves. The source code and instructions for downloading and running the software are on GitHub [2].

Explorations

Jellybeans. The first attempts I tried were turtle instructions consisting of just one arc `[a 1 x]`. A sweep of 1 (180°) on the sphere and a radius of x ranging from 0 to 0.5. Figure 3 shows the results using a map of the world as the source spherical imagery. Right away I found their shapes simple and appealing. Also notice how a radius of 0 is equivalent to just a point `[l 0]` and a radius of 0.5 results in a "line" equivalent to `[l 1]`.

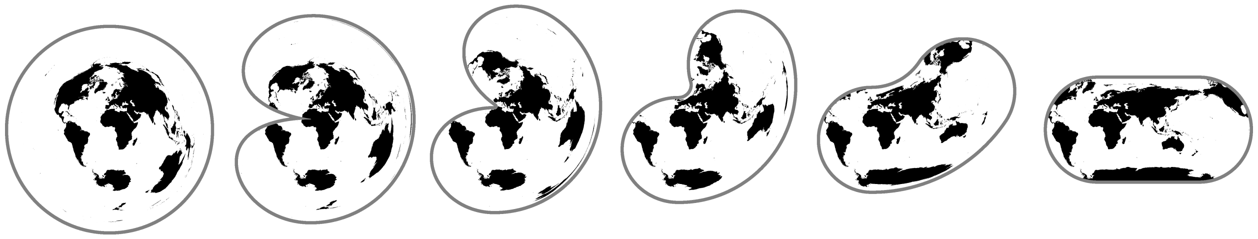


Figure 3: *Jelly beans*

Wiggles. Next, I thought about a series of 180° arcs going above and below the equator returning to the start. We can define each of n arcs that circle the globe as `[a 1 ±1/n]`. The radius here alternates positive and negative indicating the center of the arc alternating on the left and right side. Figure 4 shows four of these projections using a 2017 panorama of my backyard hand drawn by me and inked by Michael Swart.

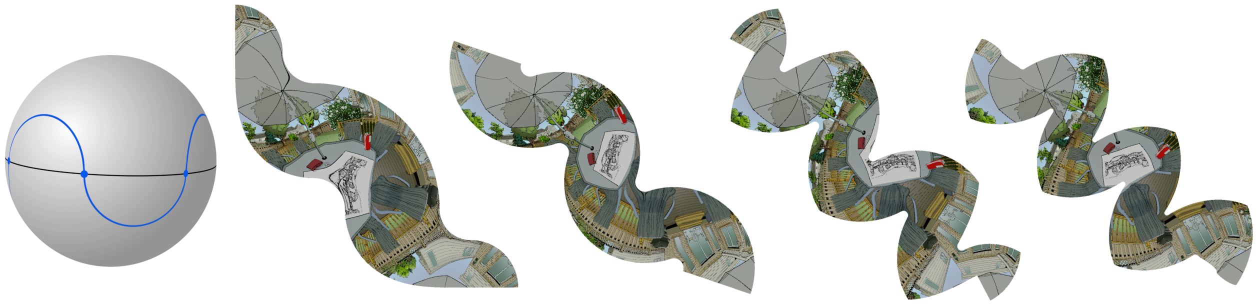


Figure 4: *A sphere with six 180° arcs oscillating above and below the equator followed by the projections resulting from 3, 4, 5, and 6 arcs.*

Sphericons. I have long been inspired by the variety and imagination of the output projections available in Lloyd Burchill's Flexify 2 Photoshop plugin [1] which is where I first learned of sphericons. Imagine the solid of revolution of a regular polygon. Then slice this shape into two equal halves with a plane containing the axis of revolution, and put them back together, rotating the cross section onto itself. The resulting shape has a surface which is well suited for construction using our new software. As an example, consider the solid of revolution of an octagon cut in half and rotated by 45°. The path along this surface is visited by eight spherical arcs which we can enter into our software as: `[a 1 0.125 a 1 0.375 a 1 -0.375 a 1 -0.125 a 1 0.125 a 1 0.375 a 1 -0.375 a 1 -0.125]`. We see the result shown in Figure 5.

Snakes. The unrolled sphericon projection looks like a snake. Pushing this example further, I wanted to create a more intricate, meandering, space filling path on the sphere and look at the resulting projections. To get this curve, we use the *Repulsive Curves* technique from Yu et al. [6] and approximate these meandering paths with our arcs. The implementation details would not fit in this paper, but readers can find the code in GitHub [2] and try it out for themselves. I have created several of these snakes. See Figure 6. I am personally very pleased with them. I especially like their graffiti-like quality.

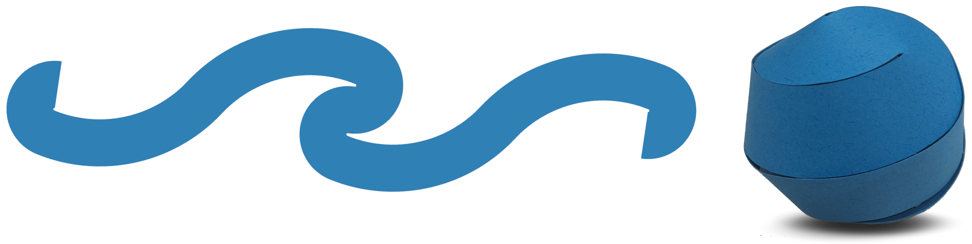


Figure 5: *A sphericon constructed with eight arcs on the sphere.*

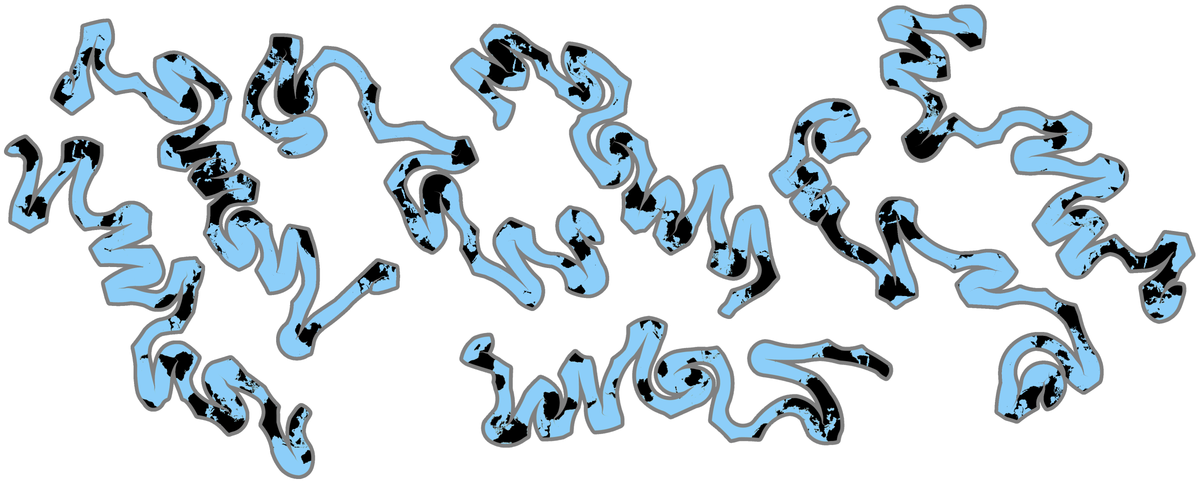


Figure 6: *Snakes on a plane.*

Future work

This will not be the end of my explorations of arcs on spheres. I hope to further explore the idea of having a space filling repulsive-curve-like path on the plane corresponding to a space filling curve on the sphere that intersects itself. My intent is to construct a woven sphere from a strip of paper efficiently cut out of a finite rectangular piece of paper. This new ability to trace curved paths with arcs will be a valuable tool to help me achieve this. From an artistic point of view I have also begun work on a woodcut print of a snake-like projection drawn in the shape of a dragon. The results are pending and not yet available at the time of writing.

References

[1] L. Burchill. *Flexify output modes - Flaming Pear Software*. http://www.flamingpear.com/flexify-output-modes.html.

[2] GitHub. *dmswart/Personal-Projects*. http://github.com/dmswart/Personal-Projects.

[3] J. P. Snyder. *Flattening the Earth: Two Thousand Years of Map Projections*, 4th ed.Chicago Press, 1997.

[4] D. Swart. "Using Turtles and Skeletons to Display the Viewable Sphere." *Proceedings of Bridges 2009: Mathematics, Music, Art, Architecture, Culture*. C. S. Kaplan and R. Sarhangi, Eds. London: Tarquin Publications, 2009. pp. 39–46. http://archive.bridgesmathart.org/2009/bridges2009-39.html.

[5] D. Swart. "Orange Peel Optimization." *Proceedings of Bridges 2021: Mathematics, Art, Music, Architecture, Culture*. D. Swart, F. Farris, and E. Torrence, Eds. Phoenix, Arizona: Tessellations Publishing, 2021. pp. 241–248. http://archive.bridgesmathart.org/2021/bridges2021-241.html.

[6] C. Yu, H. Schumacher, and K. Crane. "Repulsive Curves." *ACM Trans. Graph.*, vol. 40, no. 2, 2021.