Final Report

Bethany Witemeyer

Geodesics on Tensor Product Surfaces

Problem Summary

The calculation of geodesics, or shortest paths, on a surface is an important problem due to its wide range of applications. Geodesics are used in geometric modeling, engineering, fabrication, and many other fields. However, the calculation of geodesics, especially on arbitrary surfaces, can be incredibly complicated. For continuous surfaces, geodesics are usually found by numerically solving a differential equation.

For this project, we look specifically at calculating geodesics on tensor product surfaces. Rather than solving a differential equation, we discretize the path between two points and perform a minimization on the length of the path. We use the Euclidean distance to approximate the path length.

Previous Work

Much of the previous work on geodesic calculation consider the problem of calculating geodesics on a discritized surface, such as a mesh. Most of these works compute geodesics as the shortest path between points [1,10,11]. Additionally, several works use an iterative approach to compute the shortest geodesics [3,6]. Crane et al. use the relationship between heat and distance to compute geodesics [2]. Finally, Polthier and Schmies consider the problem of the straightest geodesic rather than the shortest [9].

When dealing with continuous surfaces, the most common approach is to set up a differential equation [4, 5, 12]. These differential equations do not typically have a closed form solution, especially for arbitrary surfaces, so numerical methods are usually employed to solve them. Pham-Trong et al. instead discretize the path between two points, use Bèzier curves to connect them, and utilize the process of subdivision to find the geodesic [8]. Finally, Ng et al. use a radial basis function method on tensor product Bèzier patches to compute geodesics.

For this project, I used MATLAB 2021a [7] to write all my code.

Description of Work

The first part of this project was spent getting a user interface put together. I wanted the code to be interactive, and have included a link to it on my project webpage. The user interface can be seen in Fig. 1. The user can choose from a variety of surfaces using a drop down menu in the bottom left corner. Additionally, the user can control the placement of the start and

end point using the text fields in the bottom right corner. These values are parameter values, and thus are expected to be between 0 and 1. Pressing the button reading "Calculate Geodesic" will run the minimization routine that computes the approximate geodesic on the surface between the start and end points.

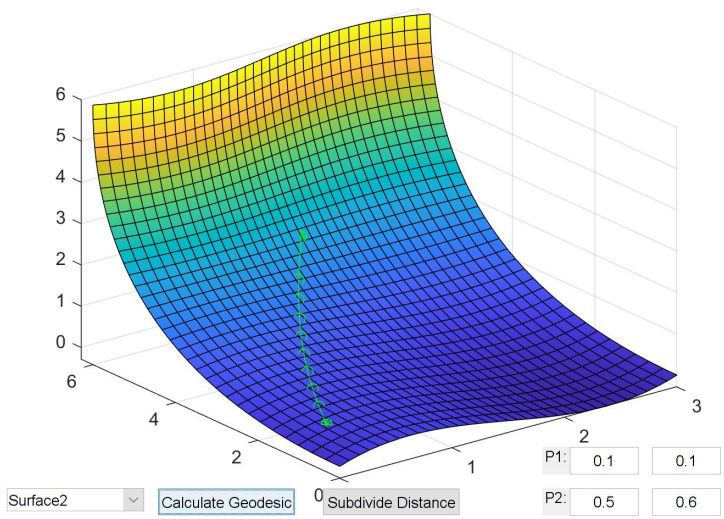

Figure 1: A geodesic on a relative smooth surface

The minimization function minimizes the length of the geodesic path. The initial guess is formed by taking steps of equal size in the parameter space to form a discrete path with 10 segments. The length of the path is calculated by computing the sum of the Euclidean distances of each of the segments. This means that the geodesics calculated in this program are approximate. For surfaces that are relatively smooth, such as the one in Fig. 1, this length calculation works well. However, for surfaces that are not relatively smooth, the fact that all points on the path except the endpoints are free to move around in space can lead to optimal paths like the one shown in Fig. 2.

To address this issue, I implemented a second distance function that further subdivides the path without creating any additional optimization variables. Specifically, rather than calculate the Euclidian distance of each segment, each segment was divided into 10 sub-segments. The sum of the Euclidean distances of all the sub-segments was then used for the length of

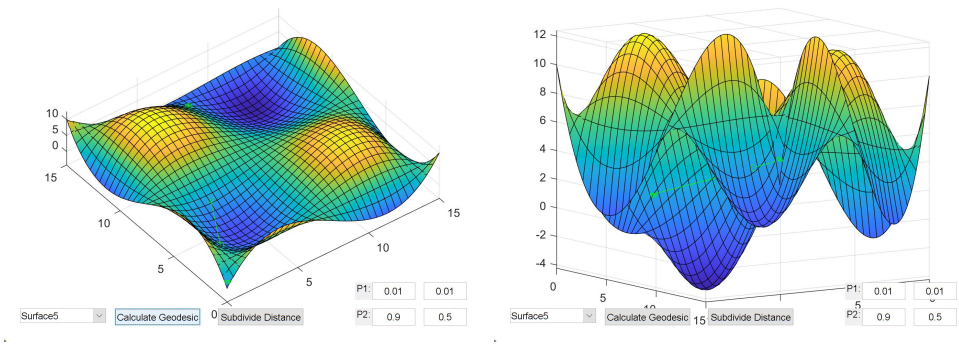

Figure 2: A optimal path that is not a geodesic

the path. This creates a distance measure that is closer to the length of the actual path on the surface, as seen in Fig. 3. This distance measure can be toggled in the user interface using the "Subdivide Distance" button.

Results

The result of this project is an interactive program that calculates approximate geodesics on tensor product surfaces, as seen in Fig. 1 and Fig. 3. The first surface I tested on was a cubic Bèzier surface where all the control points were in the same plane (Fig. 4). With this surface, any geodesic would be the Euclidean distance between the start and end points. This enabled me to verify that my minimization function was working correctly.

The program contains six surfaces. The first four are Bèzier tensor product surfaces, and the last two are LaGrange tensor product surfaces (with the parameter values evenly spaced along the interval [0,1]). Surface 2 (Fig. 1) is the Bèzier tensor product surface that was given to the class on Homework 4. The surface used in Figs. 2 and 3 is Surface 5. Surfaces 4 and 6 are generated from the same set of control points, and both surfaces had a very interesting result from running the minimization function. I expected that the minimization function would result in something similar to Fig. 2. However, both distance measures resulted in a geodesic curve that looked like the one shown in Fig. 5. According to the MATLAB output, the minimization function found a local minimum. So, it is possible that some geodesics will not suffer (as much) from the distance approximation.

Beyond the flat surface, all the geodesics calculated were evaluated on whether or not they "looked right." This is because I did not have exact distances to compare the results to. However, most of the geodesics calculated

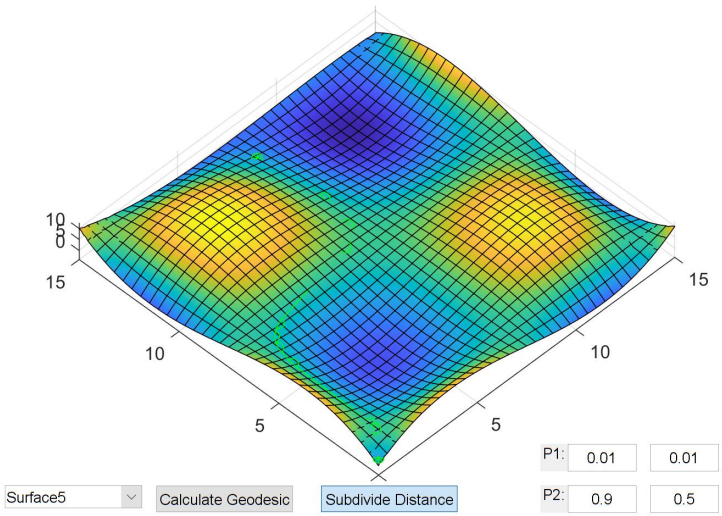

Figure 3: The geodesic found using the updated distance calculation

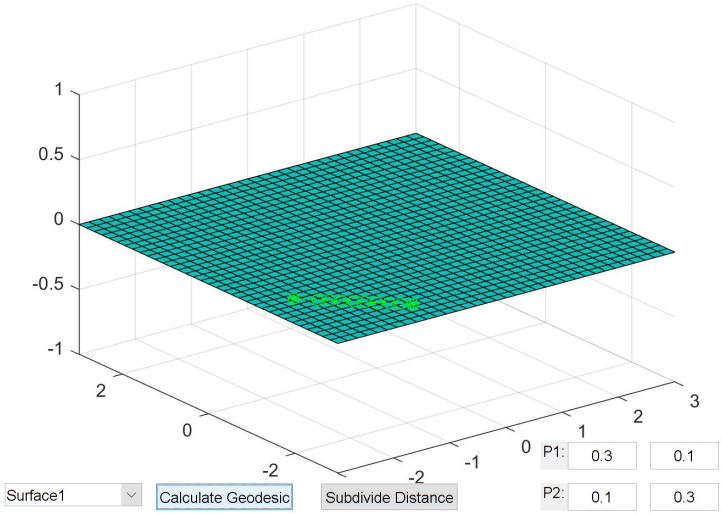

Figure 4: A geodesic on a flat plane

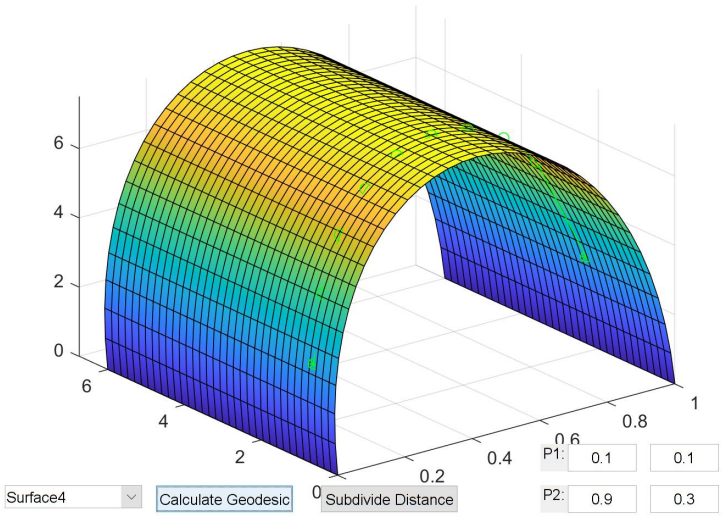

Figure 5: A geodesic corresponding to a local minimum

did look reasonably close to what I would expect the actual geodesic to be.

Analysis of Work

I do not think this project has accomplished anything new. While I did not find many source that approched this problem in a similar way, this approch I used is a fairly naive, and I would be shocked if it had not been tried before.

I did a decent job of meeting the goals I had for this project, at least for the first two updates. Time management was hard for me with this project, though, and several of those goals were met after the deadline I had originally set for myself. For this final report, I actually only accomplished one of my original goals. One goal was to reparameterize the geodesic paths using arc length as the parameter. The resulting curves had points that were somewhat evenly spaced, though, so I opted to not include the reparameterization. Additionally, I originally had the goal of visualizing the intermediate steps of the minimization process. This goal was originally made when I thought I would be controling all the minimization code. Instead, I ended up using MATLAB's built in fminunc function to do the minimization. This function runs fairly quickly, so visualizing the intermediate steps wasn't as important.

The goal I did accomplish was finding the minimization settings that give

the best results. That was accomplished by modifying the calculation of the length of the geodesic so that the distance was closer to the actual path length (see above for details). I also did not accomplish any of my stretch goals for this project. I simply did not make enough time for them.

If someone were to continue working on this project, some of my stretch goals would be good areas of future work, and I think they would yield some interesting results. For example, using machine learning to speed up the calculation of geodesics, looking at applying methods used for discretized surfaces to continuous surfaces, and comparing this method to some of the other methods for calculating geodesics on continuous surfaces (like differential equations). Additionally, I think it would be interesting to see if there was a certain aspect of surfaces that lead to local minima as discussed about Surfaces 4 and 6 earlier in this report.

References

[1] Jindong Chen and Yijie Han. Shortest paths on a polyhedron. In *Proceedings of the sixth annual symposium on Computational geometry*, pages 360–369, 1990.

[2] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Transactions on Graphics (TOG)*, 32(5):1–11, 2013.

[3] Takashi Kanai and Hiromasa Suzuki. Approximate shortest path on a polyhedral surface based on selective refinement of the discrete graph and its applications. In *Proceedings Geometric Modeling and Processing 2000. Theory and Applications*, pages 241–250. IEEE, 2000.

[4] Ron Kimmel, Arnon Amir, and Alfred M. Bruckstein. Finding shortest paths on surfaces using level sets propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):635–640, 1995.

[5] Takashi Maekawa. Computation of shortest paths on free-form parametric surfaces. 1996.

[6] Dimas Martinez, Luiz Velho, and Paulo C Carvalho. Computing geodesics on triangular meshes. *Computers & Graphics*, 29(5):667–675, 2005.

[7] MATLAB. *version 9.10.0.1649659 (R2021a) Update 1*. The MathWorks Inc., Natick, Massachusetts, 2021.

[8] Valérie Pham-Trong, Nicolas Szafran, and Luc Biard. Pseudo-geodesics on three-dimensional surfaces and pseudo-geodesic meshes. *Numerical Algorithms*, 26(4):305–315, 2001.

[9] Konrad Polthier and Markus Schmies. Straightest geodesics on polyhedral surfaces. In *ACM SIGGRAPH 2006 Courses*, pages 30–38. 2006.

[10] Micha Sharir and Amir Schorr. On shortest paths in polyhedral spaces. *SIAM Journal on Computing*, 15(1):193–215, 1986.

[11] Vitaly Surazhsky, Tatiana Surazhsky, Danil Kirsanov, Steven J Gortler, and Hugues Hoppe. Fast exact and approximate geodesics on meshes. *ACM transactions on graphics (TOG)*, 24(3):553–560, 2005.

[12] Dan-Eugen Ulmet, AC Albu, and M Craioveanu. Geodesic offsets of spline curves on spline surfaces-an industrial perspective. In *Proceedings of the 24th Conference of Geometry and Topology*, pages 263–274, 1996.