CSE306 Geometry Processing

Vrushank Agrawal

June 2023

1 Introduction

This Geometry Processing project is implemented in C++ with the following elements: Sutherland-Hodgman for polygon clipping, power diagram using parallel linear enumeration, and Tutte's Mapping. The rendered images in the report are 512×512 pixels. My laptop uses the 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz with 8 logical processors.

2 Code

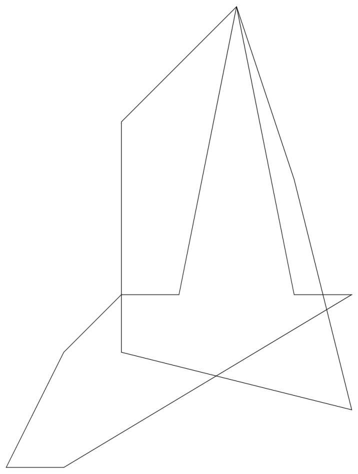
The entire code is available in one the *Geometry Processing* folder. The folder contains the files *vector.h* which contains the `Vector` class, *svg.h* which contains the `Polygon` class and the `svg` functions to save the images, *lbfgs.h* which would have contained the lbgfs implementation, and the *main.cpp* file that contains the main algorithms mentioned in the introduction. which have been ordered as `Sutherland-Hodgman` in *clipPolygon*, `Voronoi Parallel Linear Enumeration` in *voronoiPLE*, and finally `Tutte's Mapping` in *tutte*. I attempted to implement the *L-BFGS* and *Fluid Dynamics* lab but they took too much time and I could not finish it. All the algorithms were implemented with the instructions of and inspiration from the lecture notes.

3 Renders

This section will display the different renders in the project.

3.1 Polygon Clipping

The figure 1 is a render for Polygon Clipping where 1a shows the two Polygons before clipping while 1b shows the final clipped polygon.


(a) Polygons before clipping

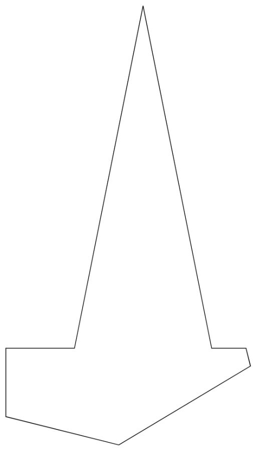
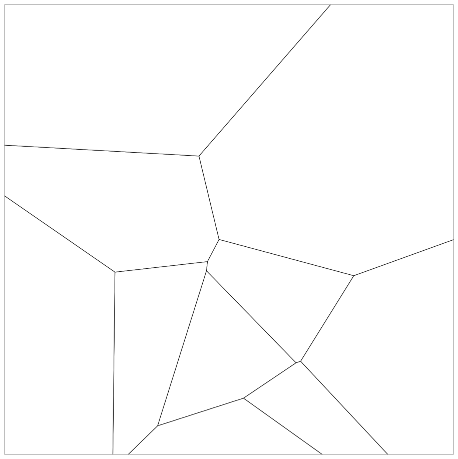

(b) Polygons after clipping

Figure 1: Polygon Clipping

3.2 Voronoi

The figure 2 is a render for Voronoi diagrams where 2a shows the Voronoi diagram with 10 polygons while 2b shows the Voronoi diagram with 1000 polygons.


(a) Voronoi diagram with n=10

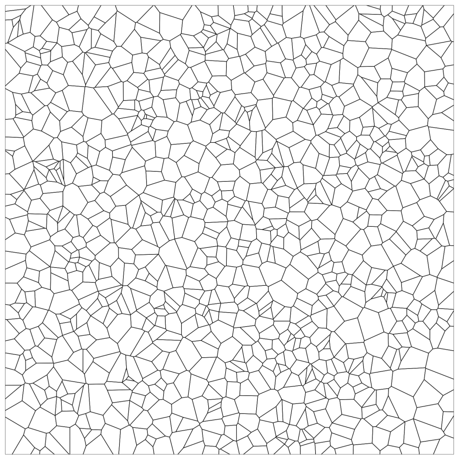
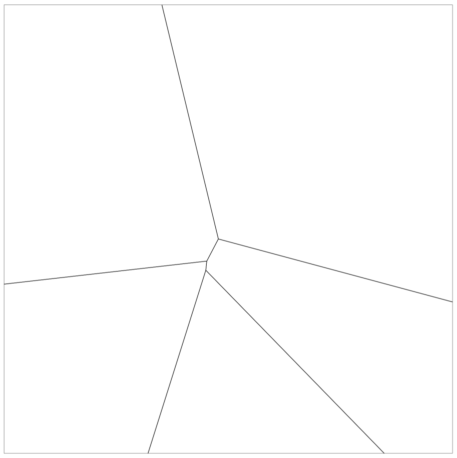

(b) Voronoi diagram with n=1000

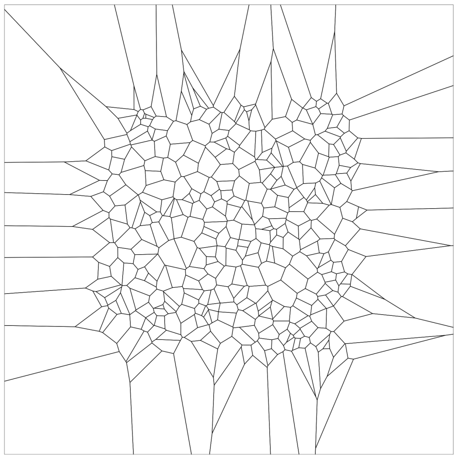
Figure 2: Voronoi diagrams

3.3 Power Diagram

The figure 3 is a render for Power Voronoi diagrams where 3a shows the Power Voronoi diagram with 10 polygons while 3b shows the Power Voronoi diagram with 1000 polygons. In the diagrams, the weights of the polygons were initialized to zero if any of their points remained outside the bounds of the square polygon (0.2, 0.8) x (0.2, 0.8).



(a) Power Voronoi diagram with n=10



(b) Power Voronoi diagram with n=1000

Figure 3: Power Voronoi diagrams

3.4 L-BFGS

I tried to implement L-BFGS but I could not succeed in making it work. There seems to be an error in my `integrateSqDist` function in the `Polygon` class that I implemented but if I try to fix it I keep on getting a segmentation fault. You may check my code in the repository.