PROJECT REPORT

Computer Graphics (2024-2025)

Assignment 2

Submitted By: Andreea Patarlageanu

Bachelor of Science

École Polytechnique

June 4, 2025

This project contains 4 files, *main.cpp*, *main2.cpp*, *main3.cpp* and *main4.cpp*, each corresponding to each part of this second project, meaning:

- *main.cpp* corresponds to the ”Voronoï Parallel Linear Enumeration algorithm in 2D”
- *main2.cpp* and *main3.cpp* correspond to ”Optimal Transport with Equal-Weight Voronoi Diagrams”
- *main4.cpp* corresponds to the Fluid Simulation

1 Voronoï Parallel Linear Enumeration algorithm in 2D

In this first part of the project, I implemented the Voronoï Parallel Linear Enumeration algorithm in 2D. You can see the diagrams generated for 2 different values of N and their respective computational time:

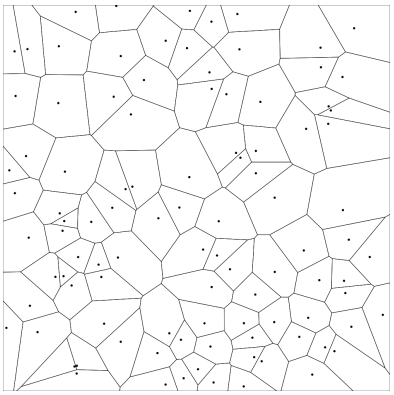

Figure 1: Voronoi compute time: 0.0090844 seconds for N = 100.

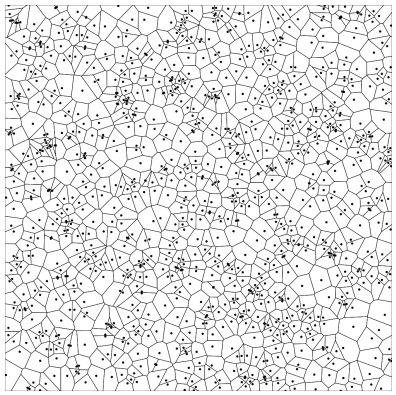

Figure 2: Voronoi compute time: 0.179938 seconds for N = 1000.

2 Optimal Transport with Equal-Weight Voronoi Diagrams

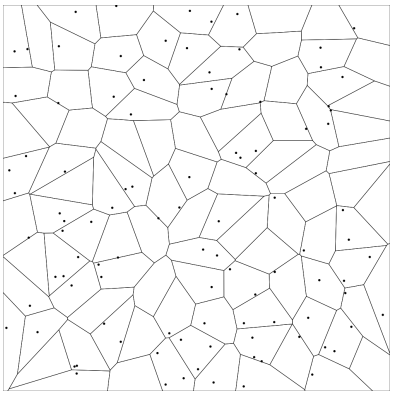

Figure 3: Voronoi compute time: 1.2336 seconds for N = 100.

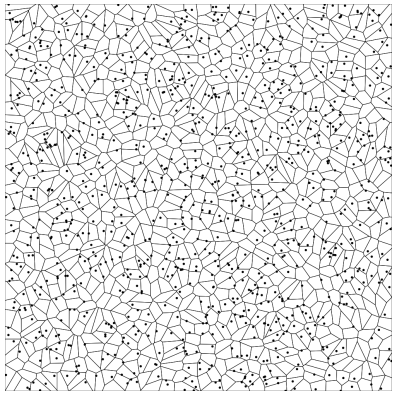

Figure 4: Voronoi compute time: 101.109 seconds for N = 1000.

In order to check if the result is correct, meaning that the cells have the same area, I added the following piece of code (deleted before adding it on Github, its role was just for debugging):

```
int optimize() {
    ...
    int ret = lbfgs(...);
    ...
    return ret;
```

```
}

//check maximum absolute area error
double max_diff = 0.0;
double desired_area = 1.0 / N;

for (int i = 0; i < N; ++i) {
    double area = ot.vor.diagram[ i ].area();
    double diff = std::abs( area - desired_area );
    if ( diff > max_diff )
        max_diff = diff;
}

std::cout<<"Max absolute area difference from 1/N: "<<max_diff<<std::endl;

//check the optimization
if ( ret == LBFGS_SUCCESS )
    std::cout<<"LBFGS optimization SUCCESS."<<std::endl;
else
    std::cout<<"LBFGS optimization FAILED."<<std::endl;
```

The output contained the following lines, which suggest the correctness of the result:

Code Output

```
Max absolute area difference from 1/N: 2.85928e-06
LBFGS optimization SUCCESS.
```

This output suggests that the **largest deviation** is very small, meaning that the areas of the cells are very close to $\frac{1}{N}$. Moreover, the second line shows that the optimization converged successfully. Thus, the result is **correct**.

3 Computational Fluid Dynamics

Thanks to the tutoring session, I realized that the barycenters were wrong. The values should be in $(0, 1)$, but mine were all negative, resulting in the wrong output. I added this code for outputting the values of the barycenters:

```
for ( int i = 0; i < vor.diagram.size(); i++ ) {
        Vector barycenter = vor.diagram[i].centroid();
        std::cout <<" barycenter: ("<<barycenter[0]<<", "<<barycenter[1]<<")"<<std::endl;
}
```

and my output looked like this, which was wrong, because it is negative:

Code Output

```
barycenter:  (-0.683008, -0.51299)
barycenter:  (-0.892057, -0.317515)
barycenter:  (-0.122975, -0.310944)
barycenter:  (-0.49286, -0.168261)
```

The **PROBLEM:** the sign of the barycenters is influenced by the orientation, which can be clockwise or counterclockwise, of the polygon's vertices. We need to ensure the correctness of the sign, otherwise, the centroid is computed incorrectly. Thus, I had to modify the sign of the returned result in the *centroid()* function:

```
double a = area();
c_x = c_x / ( 6.0 * a );
c_y = c_y / ( 6.0 * a );
Vector cen( -c_x, -c_y, 0.0 );  //minus sign
return cen;
```

In the end, I got a nice-looking fluid simulation, which can be found on GitHub as a *GIF*, called *fluid_simulation*.

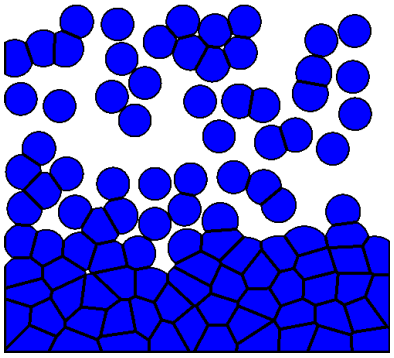

Figure 5: Fluid simulation time: 463.046 seconds for 100 particles and 270 frames.

4 Acknowledgments

I acknowledge that most of my code was written after the instructions given in class. I managed to follow the TD, almost every time, having the part of the project done before the end of the TD. In case of small bugs in my code, Joanne Jegou was the one to help me. I did **not** copy a function or part of a code, I just made small modifications in formulas if it was the case. Only for debugging purposes, AI was used only once, but not helpful anyway. The tutoring session was the one which actually helped me a lot, resulting in the code I have now.