Implementing Three Voronoi Diagram Computation Algorithms and Comparing Their Performance

We are going to implement three different approaches for calculating and visualizing the Voronoi Diagram of a set of points in 2D and then compare their performances with each other on how efficient and accurate they are. The approaches are listed below:

- Randomized Incremental Algorithm
- Fortune’s Algorithm or Plane Sweep Algorithm
- The Flipping Algorithm starts with an arbitrary triangulation and converts it by flipping the diagonals

Randomized Incremental Algorithm (RIA) will add new sites one by one onto the existing Delaunay triangulation while updating it after a new addition. The number of face updates will determine RIA’s running time.

Fortune’s Algorithm is used to discover the Delaunay triangle to pass through while Delaunay triangulation uses a horizontal line that sweeps upward across the plane. The front of this sweep is called the sweep line. It moves in discrete steps only when a site or Voroni triangle is discovered. Fortune’s Algorithm’s running time is computed by the sweep line moves and data structure -to store the list of sites- update.

The Flipping Algorithm requires an arbitrary triangulation at the beginning therefore Lexicographic Triangulation will be used as a technique for arbitrary triangulation. Flip algorithms help to prove that arbitrary point sets have Delaunay triangulation and its running time is bound to the number of points in that arbitrary set.

For the implementation, C++ programming language is decided to be used as there is a software project, API, called CGAL [1] and it yields easy access to the use of triangulations and Voronoi diagrams. The non-existing algorithms will be written by us or more specific libraries in C++ which will be discussed later with TA if it becomes very necessary to use.

As is said in the first paragraph, these algorithms will be compared to each other and ordered between them in different ways such as efficiency, accuracy, and so on.