Assignment Two: Delaunay Triangulation and Voronoi Diagram

David Gu

Yau Mathematics Science Center
Tsinghua University
Computer Science Department
Stony Brook University

gu@cs.stonybrook.edu

October 27, 2020

Delaunay Triangulation

The input to the Delaunay Triangulation algorithm is a set of 2D points

$$P = \{p_1, p_2, \ldots, p_n\}$$

The output is the Delaunay Triangulation of the point set P.

Input

The input points are randomly generated within the unit disk.

Output

The Delaunay triangulation is represented as a triangle mesh, using Dart data structure to store.

Delaunay Triangulation

Algorithm Pipeline

- For each point $p_i(x_i, y_i) \in P$, construct a point q_i,

$$q_i = \left(x_i, y_i, \frac{1}{2}(x_i^2 + y_i^2) \right), i = 1, 2, \cdots, n.$$

- Compute the convex hull of $\{q_1, q_2, \cdots, q_n\}$;
- Remove all the faces of the convex hull, whose normals are upward;
- The projection of the left faces induce the Delaunay triangulation;

Delaunay Triangulation

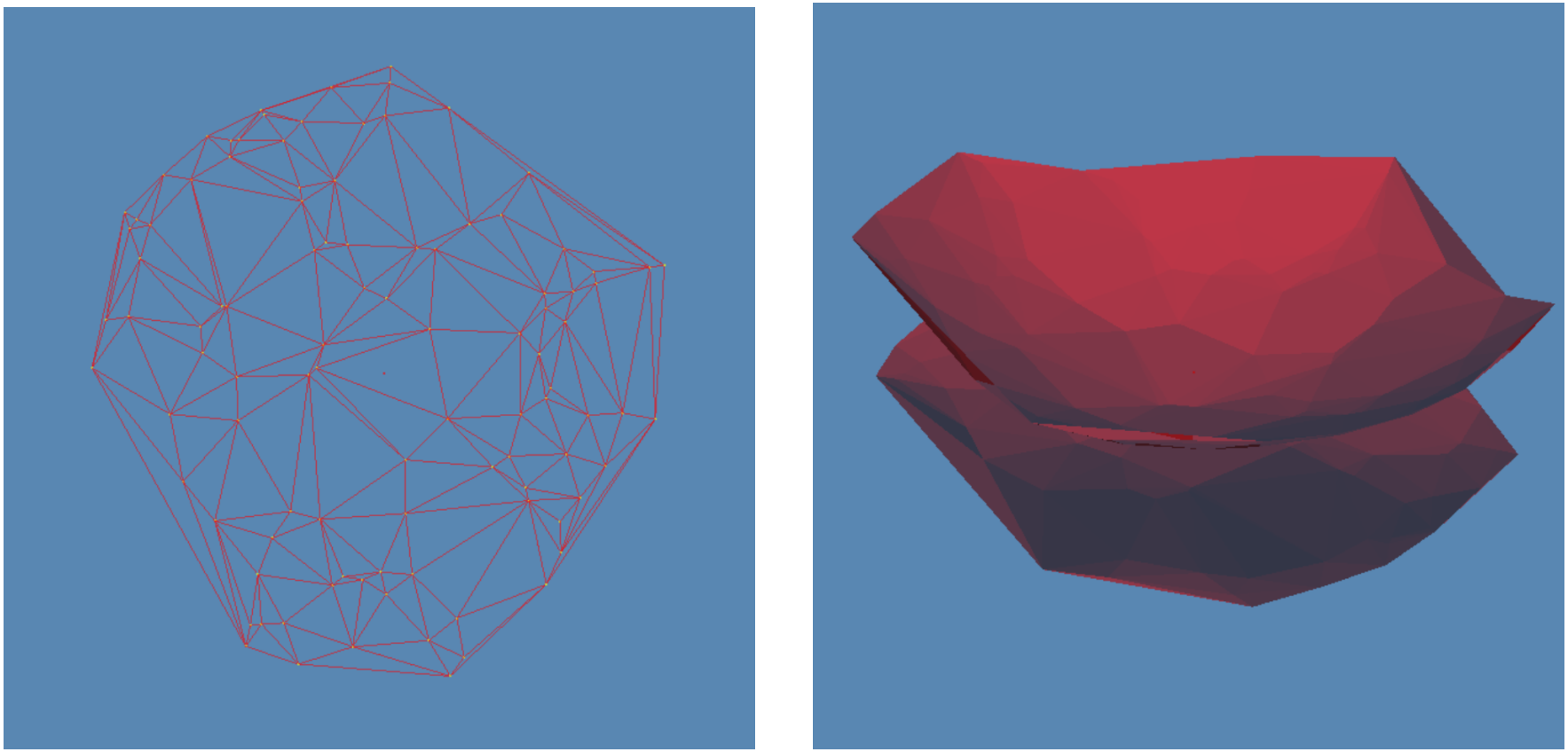

Figure: Delaunay Triangulation.

Voronoi Diagram

Upper Envelope

- For each point $p_i(x_i, y_i) \in P$, construct a point q_i,

$$q_i = \left(x_i, y_i, \frac{1}{2}(x_i^2 + y_i^2) \right), i = 1, 2, \cdots, n.$$

- Compute the convex hull of $\{q_1, q_2, \cdots, q_n\}$;
- Remove all the faces of the convex hull, whose normals are upward;
- For each face on the convex hull, compute the dual point;
- For each interior edge on the convex hull, compute the dual edge;
- For each interior vertex on the convex hull, compute the dual face;
- For each boundary edge, compute the dual ray;

Voronoi Diagram

Voronoi Diagram

- Compute the upper envelope
- Project the upper envelope to obtain the Voronoi diagram.

Example

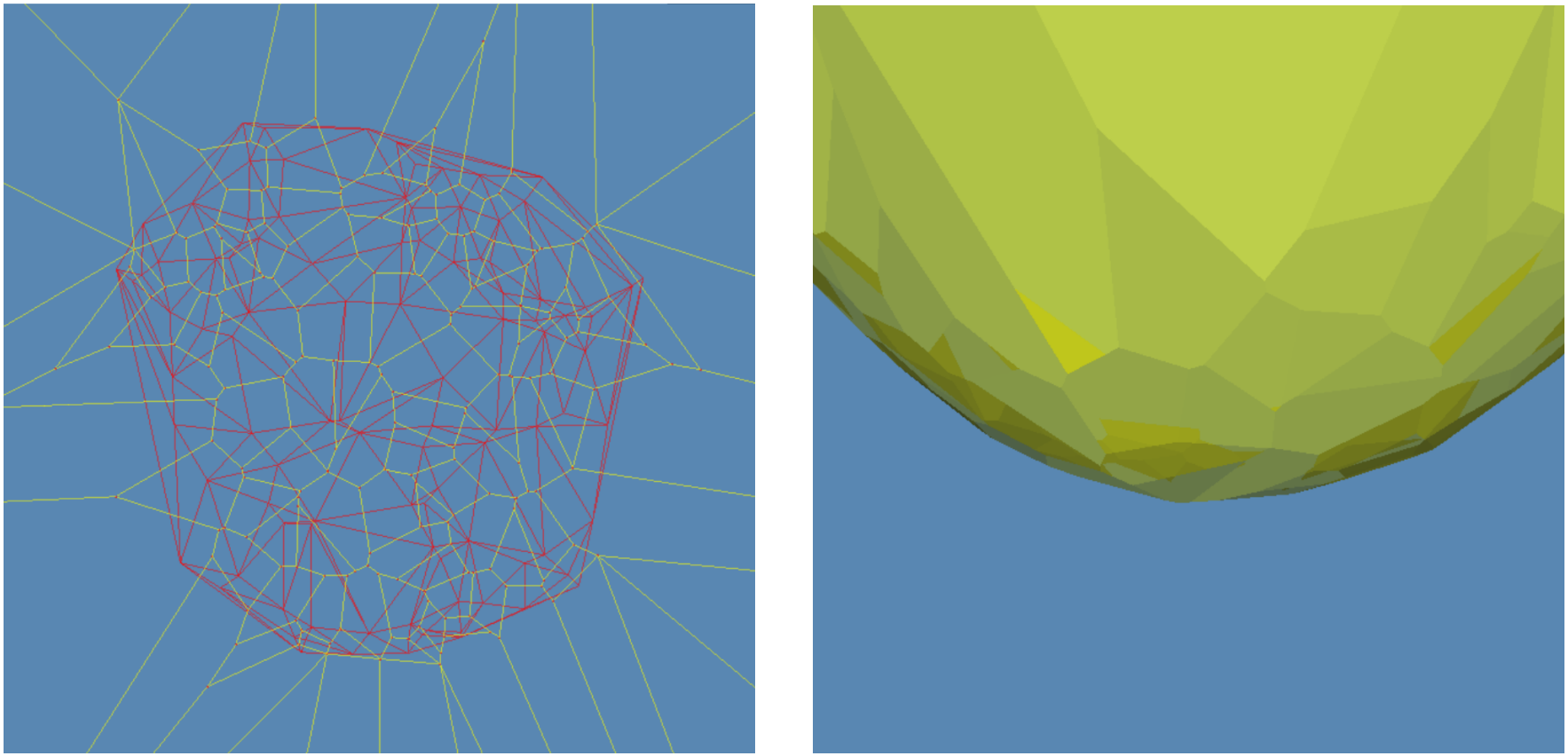

Figure: Voronoi diagram (left) and the upper envelope (right).

Instruction

Dependencies

1. 'DartLib', a general purpose mesh library based on Dart data structure.
2. 'freeglut', a free-software/open-source alternative to the OpenGL Utility Toolkit (GLUT) library.

Directory Structure

- 3rdparty/DartLib, header files for mesh;
- convex_hull/include, the header files for convex_hull;
- convex_hull/src, the source files for convex_hull;
- power_diagram/include, the header files for convex_hull;
- power_diagram/src, the source files for convex_hull;
- CMakeLists.txt, CMake configuration file;

Configuration

Before you start, read README.md carefully, then go three the following procedures, step by step.

1. Install [CMake](https://cmake.org/download/).
2. Download the source code of the C++ framework.
3. Configure and generate the project for Visual Studio.
4. Open the .sln using Visual Studio, and complie the solution.
5. Finish your code in your IDE.
6. Run the executable program.

Configure and generate the project

1. open a command window
2. cd ot-homework2_skeleton
3. mkdir build
4. cd build
5. cmake ..
6. open OTHomework.sln inside the build directory.

Finish your code in your IDE

- You need to modify the file: ConvexHull.cpp, PowerDiagram.cpp and viewer.cpp;
- search for comments "insert your code"
- Modify functions:
  1. $ConvexHull :: \_volume\_sign(CConvexHullMesh :: CFace*, constCPoint)$
  2. $ConvexHull :: \_inside(constCPoint)$
  3. $ConvexHull :: \_remove\_visible(constCPoint)$
  4. $ConvexHull :: \_close\_cap(constCPoint)$
  5. $CPowerDiagram :: init(intnum\_pts)$
  6. $CPowerDiagram :: calc\_delaunay()$
  7. $CPowerDiagram :: calc\_voronoi()$
  8. $void\ drawBoundaryDualEdge(boolisPlane = false)$

Finish your code in your IDE

Insert your solution to the assignment one to ConvexHull.cpp, to implement the algorithms for Delaunay triangulation and Voronoi diagram.