Goldy

HomeWork 3

1. DataStructure Used:
    a. UniDirectional Graph: To store the triangulation after it is being created by inserting the points one by one. When a point is inserted, the new triangles are made as the children of the one it is inserted in with proper parents.
    b. HalfEdge: To maintain the vertices, edges and triangles I am using halfedge data structure.
        i. Each Triangle/Face points to its halfedges in anti-clockwise order and also to its children – after subdividing or updating during insertion of a new point
        ii. Each Halfedge points to the vertex its pointing to, the face it belongs to, its pair and the next halfedge in the face
        iii. Vertex- Storing the x, y coordinates of the point

2. Algorithm:

    I have used edge-flipping algorithm to get the Delauney triangulation.

    i. Points are inserted one by one in a large triangle and the graph is traversed until the child node (the smallest triangle) containing point is found
    ii. Point may lie inside the triangle or on any of the edges. These two cases are handled separately and new triangles are created, and added as nodes in the triangulation graph - Dot product check is used
    iii. After that adjacent edges are checked if they are still legal. For that I am finding minimum angle in the present case and the case with the edge flipped. Goal is to maximize the minimum angle.
    iv. If flipping takes place, adjacent edges are checked again if they are still legal. This goes on till all the surrounding edges satisfy the Delauney criteria.

    I am using a very large triangle just to find the triangle in which the point to be inserted lies.

    But in case any of the large edge is involved while calculating the angles, I am doing that lexicographically instead of numerically so that there isn't any floating point error.

The time complexity of this algorithm is O(NlogN).

I checked my program for two degenerate cases: **Co-linear points** and **4 Co-circular points**. It seems to work fine

Computing Voronoi from Delauney:

So when we find Delauney we also have Voronoi, Since delauney and Voronoi are dual to each other in case no more than 3 points are co-circular.

1. Each triangle in Delauney corresponds to one vertex in Voronoi
2. Each edge in Delauney is corresponds to edge in Voronoi
3. Each vertex in Delauney corresponds to a region in Voronoi.

Create a new Datastructure VOR: having a site, array of edges, and vertices

For first root node(triangle) in delauney-

1. Get the circumcenter: this is a Voronoi Vertex.
2. The vertices of the triangle forms Voronoi Site
3. Look at the adjacent traingles (using halfedge)- if there is one for any edge- get the circumcircle and join to the present one- to get an edge. If there is not vornoi edge goes to infinity passing through the midpoint of that edge.
4. Mark the face visited and go on looking in to the adjacent faces one by one for each half edge recursively