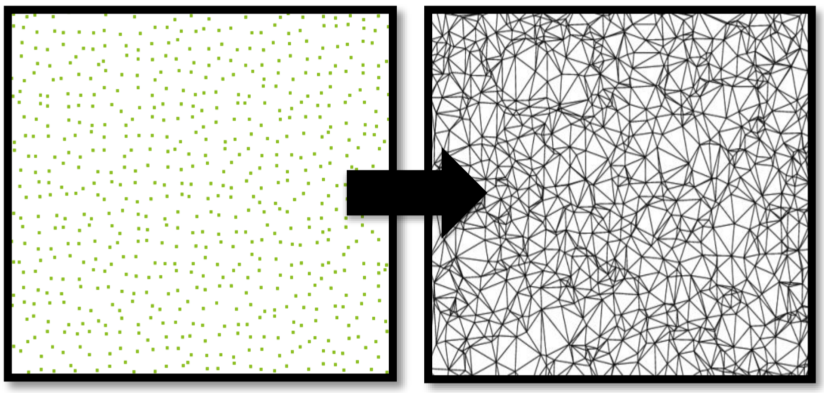
Images Synthesis practical 1

Delaunay Triangulation (vertex insertion method)

Delaunay triangulation (via vertex insertion) is a method of converting a point-cloud to well-formed triangular mesh.



Source code

The source code for the practical contains:

- a text file with a set of 3D points
- a parser to read this file and parse the contents into a std::vector (Parser.h)
- basic structs: Vertex, Face, Mesh (Mesh.h) and Edge (Delaunay.h)
- code to export Mesh to .ply format (Mesh.h and .cpp)
- a skeleton delaunay triangulation implementation (Delaunay.h and .cpp). The function triangulateMesh():
    - creates a super triangle (which contains all vertices)
    - has skeleton code to add vertices to the mesh
    - deletes super triangle from mesh at end of algorithm

What you have to do

1) Your job is to implement the function:

void addVertex(Mesh &mesh, const int vertexIndex);

Parameters

- Mesh &mesh

  Mesh object containing std::vectors of Vertex and Face structures

- const int vertexIndex

  The index of the current vertex

2) You will also have to implement all other structs and functions in order for addVertex to work correctly. E.g. you will need to store Edges somehow, and need a function which compares to see if two edges are the same edge

Pseudocode

```
Add container triangle to triangle_list
For vertex vi in vertex_list
{
    For triangle ti in triangle_list
    {
     If vi is inside circumcircle of ti
     {
          Save edges of ti to edges_list
          Delete triangle ti
     }
    }
    Remove duplicate edges in edges_list
    For edge ei in edges_list
    {
     Create triangle which links vertices of ei with vi
    }
    Clear edges_list
 }
 Eliminate triangles with vertices of container triangle from    triangle_list
```

Helper functions

bool isInsideCircumCircle(const Vertex A, const Vertex B, const Vertex C, const Vertex &point);

Receives three vertices of a triangle (A, B, C) and returns true if point is inside the circumcircle of that triangle, otherwiser returns false

How to view result

The result is written out in ASCII .ply format, a very simple mesh format.

The simplest way of visualising .ply is with meshlab: **http://www.meshlab.net/**

Basic Meshlab tutorial: **https://www.youtube.com/watch?v=Sl0vJfmj5LQ**

The final result should look a bit like this: