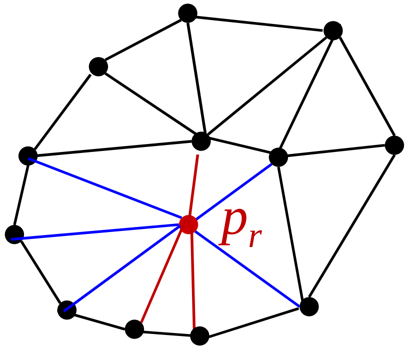
CMPS 6640/4040 Computational Geometry
Spring 2016



Delaunay Triangulations

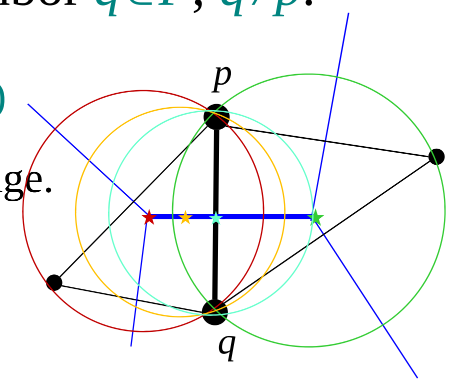
Carola Wenk

Based on:

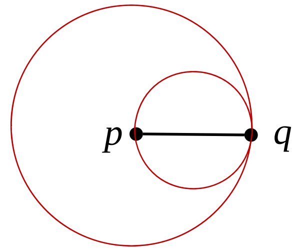
Computational Geometry: Algorithms and Applications

Applications of DT

- All nearest neighbors: Find for each $p \in P$ its nearest neighbor $q \in P$; $q \neq p$.
  - **Empty circle property:** $p,q \in P$ are connected by an edge in DT(P) $\Leftrightarrow$ there exists an empty circle passing through p and p.
    Proof: "$\Rightarrow$": For the Delaunay edge pq there must be a Voronoi edge. Center a circle through p and q at any point on the Voronoi edge, this circle must be empty.
    "$\Leftarrow$": If there is an empty circle through p and q, then its center c has to lie on the Voronoi edge because it is equidistant to p and q and there is no site closer to c.

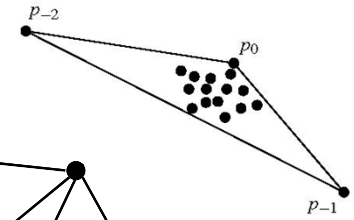


  - **Claim:** Every $p \in P$ is adjacent in DT(P) to its nearest neighbor $q \in P$.
    Proof: The circle centered at p with q on its boundary has to be empty, so the circle with diameter pq is empty and pq is a Delaunay edge.

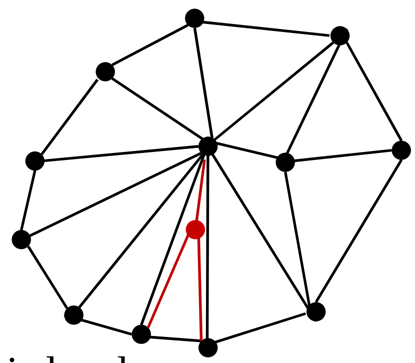


  - **Algorithm:** Find all nearest neighbors in $O(n)$ time: Check for each $p \in P$ all points connected to p with a Delaunay edge.
- Minimum spanning tree: The edges of every Euclidean minimum spanning tree of P are a subset of the edges of DT(P).

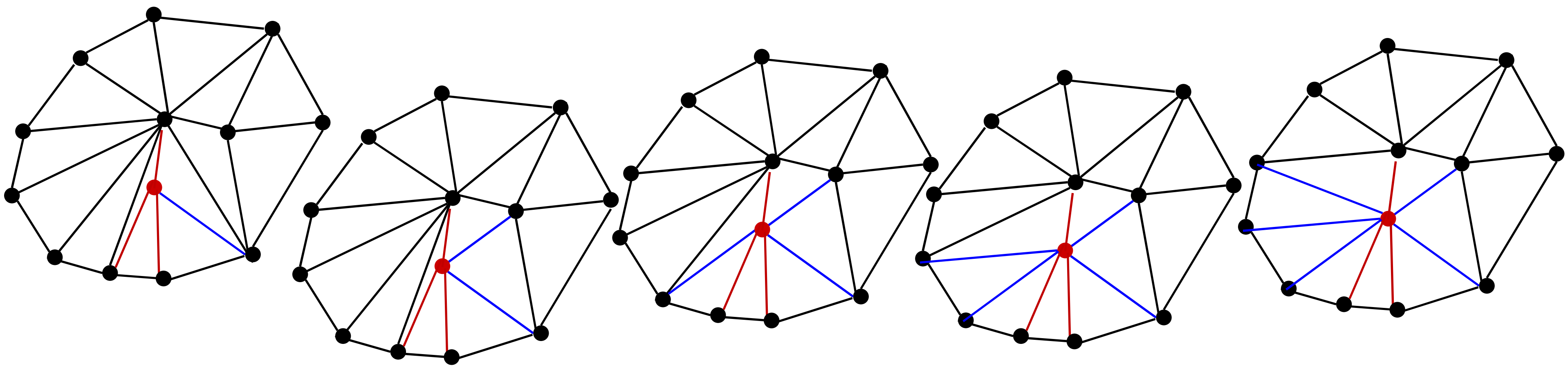
Randomized Incremental Construction of DT(P)



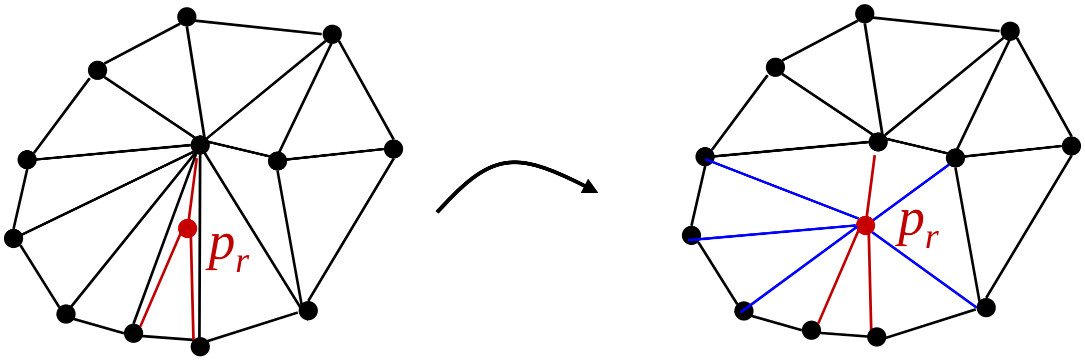
- Start with a large triangle containing P.
- Insert points of P incrementally:
  - Find the containing triangle
  - Add new edges



  - Flip all illegal edges until every edge is legal.



Randomized Incremental Construction of DT(P)



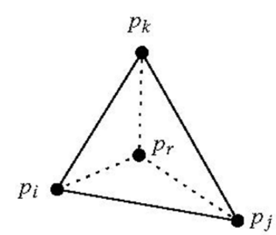
- An edge can become illegal only if one of its incident triangles changes.
- Check only edges of new triangles.
- Every new edge created is incident to p_r.
- Every old edge is legal (if p_r is on on one of the incident triangles, the edge would have been flipped if it were illegal).
- Every new edge is legal (since it has been created from flipping a legal edge).

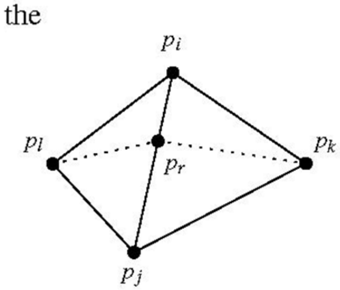
Pseudo Code

Algorithm DELAUNAYTRIANGULATION(P)
Input. A set P of $n+1$ points in the plane.
Output. A Delaunay triangulation of P.

Let p_0 be the lexicographically highest point of P, that is, the rightmost among the points with largest y-coordinate.
Let p_{-1} and p_{-2} be two points in $\mathbb{R}^2$ sufficiently far away and such that P is contained in the triangle $p_0 p_{-1} p_{-2}$.
Initialize $\mathcal{T}$ as the triangulation consisting of the single triangle $p_0 p_{-1} p_{-2}$.
Compute a random permutation $p_1, p_2, \ldots, p_n$ of $P \setminus \{p_0\}$.
for $r \leftarrow 1$ **to** n
    do (* Insert p_r into $\mathcal{T}$: *)
        Find a triangle $p_i p_j p_k \in \mathcal{T}$ containing p_r.
        if p_r lies in the interior of the triangle $p_i p_j p_k$
          then Add edges from p_r to the three vertices of $p_i p_j p_k$, thereby splitting $p_i p_j p_k$ into three triangles.
                LEGALIZEEDGE($p_r, \overline{p_i p_j}, \mathcal{T}$)
                LEGALIZEEDGE($p_r, \overline{p_j p_k}, \mathcal{T}$)
                LEGALIZEEDGE($p_r, \overline{p_k p_i}, \mathcal{T}$)
          else (* p_r lies on an edge of $p_i p_j p_k$, say the edge $\overline{p_i p_j}$ *)
                Add edges from p_r to p_k and to the third vertex p_l of the other triangle that is incident to $\overline{p_i p_j}$, thereby splitting the two triangles incident to $\overline{p_i p_j}$ into four triangles.
                LEGALIZEEDGE($p_r, \overline{p_i p_l}, \mathcal{T}$)
                LEGALIZEEDGE($p_r, \overline{p_l p_j}, \mathcal{T}$)
                LEGALIZEEDGE($p_r, \overline{p_j p_k}, \mathcal{T}$)
                LEGALIZEEDGE($p_r, \overline{p_k p_i}, \mathcal{T}$)
Discard p_{-1} and p_{-2} with all their incident edges from $\mathcal{T}$.
return $\mathcal{T}$

LEGALIZEEDGE($p_r, \overline{p_i p_j}, \mathcal{T}$)
(* The point being inserted is p_r, and $\overline{p_i p_j}$ is the edge of $\mathcal{T}$ that may need to be flipped. *)
if $\overline{p_i p_j}$ is illegal
  then Let $p_i p_j p_k$ be the triangle adjacent to $p_r p_i p_j$ along $\overline{p_i p_j}$.
        (* Flip $\overline{p_i p_j}$: *) Replace $\overline{p_i p_j}$ with $\overline{p_r p_k}$.
        LEGALIZEEDGE($p_r, \overline{p_i p_k}, \mathcal{T}$)
        LEGALIZEEDGE($p_r, \overline{p_k p_j}, \mathcal{T}$)

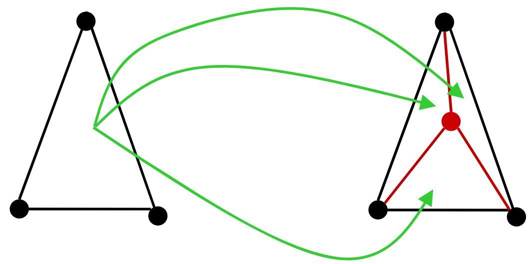




History

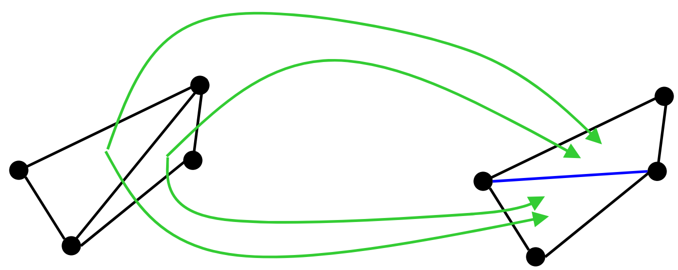
The algorithm stores the history of the constructed triangles. This allows to easily locate the triangle containing a new point by following pointers.

- Division of a triangle:



Store pointers from the old triangle to the three new triangles.

- Flip:



Store pointers from both old triangles to both new triangles.