Bowyer-Watson Algorithm

Delaunay Triangulation

Implemented by **Pooja Bhagat**

IIT Guwahati

Bowyer-Watson Algorithm

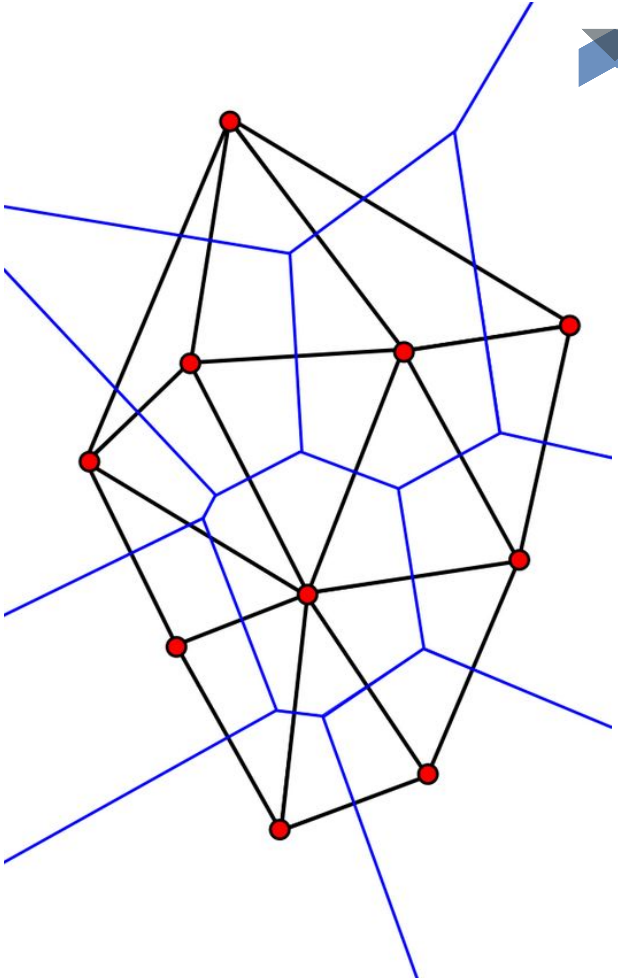
- The Bowyer–Watson algorithm is a method for computing the Delaunay triangulation of a finite set of points.
- The algorithm can be also used to obtain a Voronoi diagram of the points.
- The Bowyer–Watson algorithm is an **incremental algorithm**. It works by adding points, one at a time, to a valid Delaunay triangulation of a subset of the desired points.
- **Adrian Bowyer** and **David Watson** devised it independently of each other at the same time, and each published a paper on it in the same issue of **The Computer Journal**.

Bowyer-Watson Algorithm

- Bowyer-Watson algorithm can be used to compute Delaunay Triangulation in k dimensional Euclidean Space ($k>=2$).
- We will mainly focus on its implementation and analysis in 2 dimensional Euclidean Space.
- In k dimension the algorithm has $O(n^{1+1/k})$ running time.
- We will also look at how we can obtain the Convex Hull and Voronoi Diagram of a set of points from their Delaunay Triangulation.

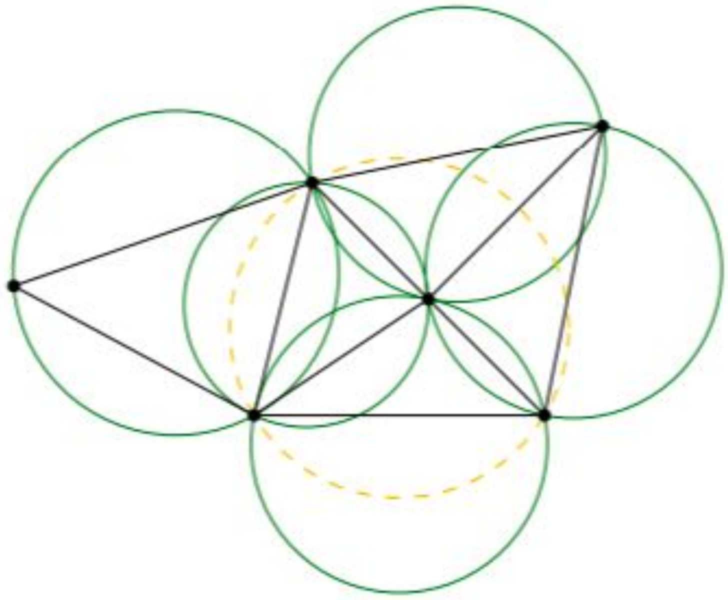
Delaunay Triangulation

- Delaunay Triangulation is the graph dual of Voronoi Diagram(VoD).
- For each face of the primal graph (VoD), we create a vertex, and then we add an edge between two such vertices if their faces are adjacent in VoD.
- **Uniqueness:** The Delaunay triangulation is unique given no four sites are co-circular.



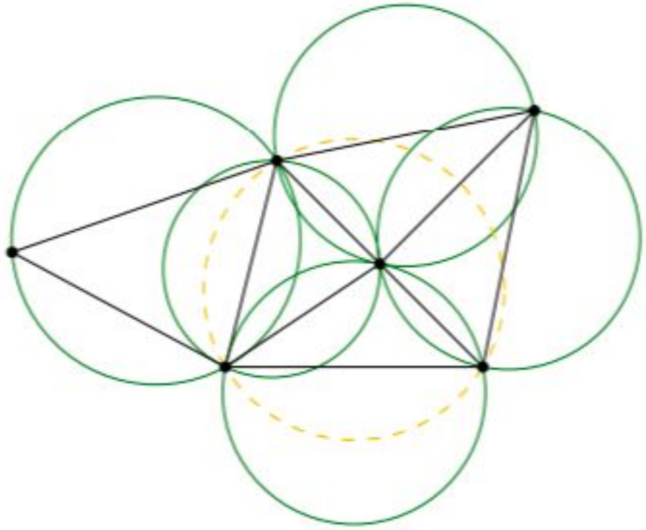
Circumcircle Criteria

- A triangulation of N>2 points is Delaunay if and only if the circumcircle of every interior triangle is point free.



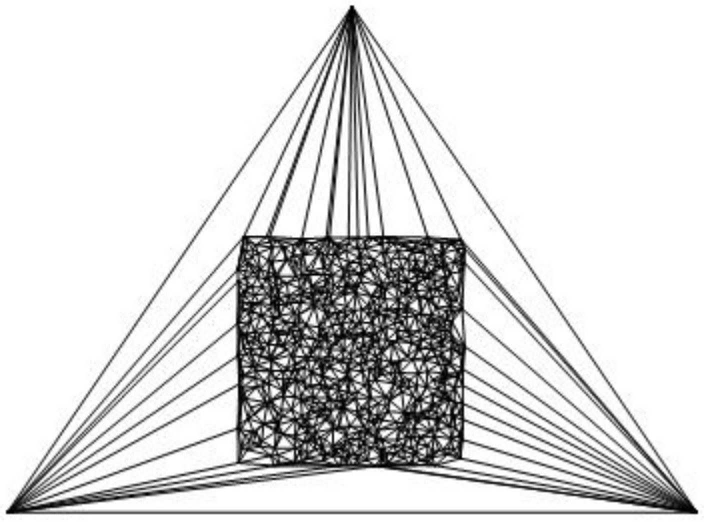
Edge Criteria

- The edge is in Delaunay Triangulation if and only if there exists an empty circle passing through its endpoints. An edge satisfying this property is said to be Delaunay.



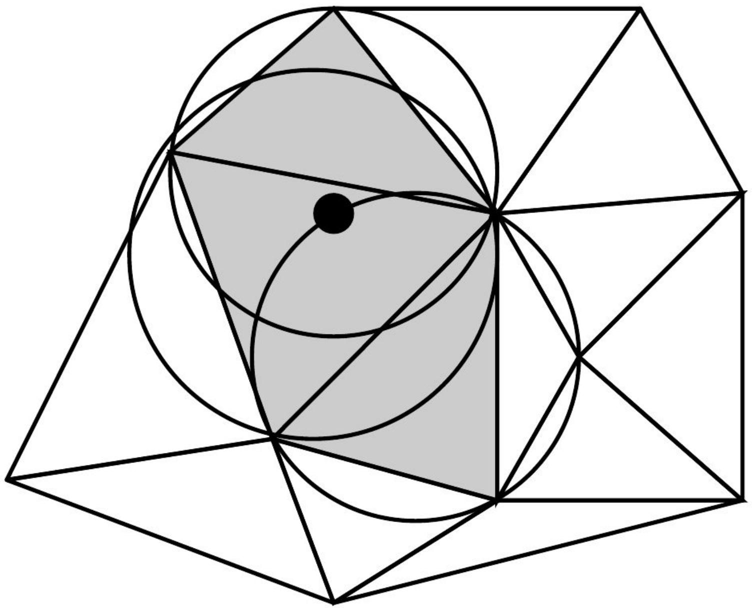
The Algorithm : Base Case

- Start with a Delaunay Triangulation whose convex Hull contains all the given set of points.
- Add a **super triangle** large enough to completely contain all the points.
- Super triangle should not affect the global delaunay triangulation of inner points.



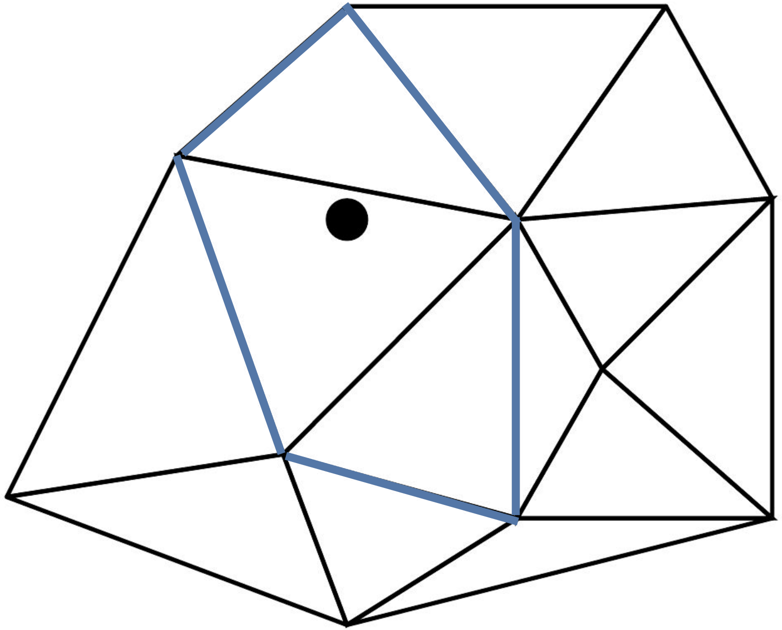
The Algorithm : Adding New Point

- Find set of all the triangles whose circumcircle encloses the new vertex (say T_{bad}).
- The triangles in T_{bad} no longer satisfy the **Circumcircle Criteria** hence are no longer Delaunay.
- All other triangles (*not in* T_{bad}) remain Delaunay triangles, and are left undisturbed.



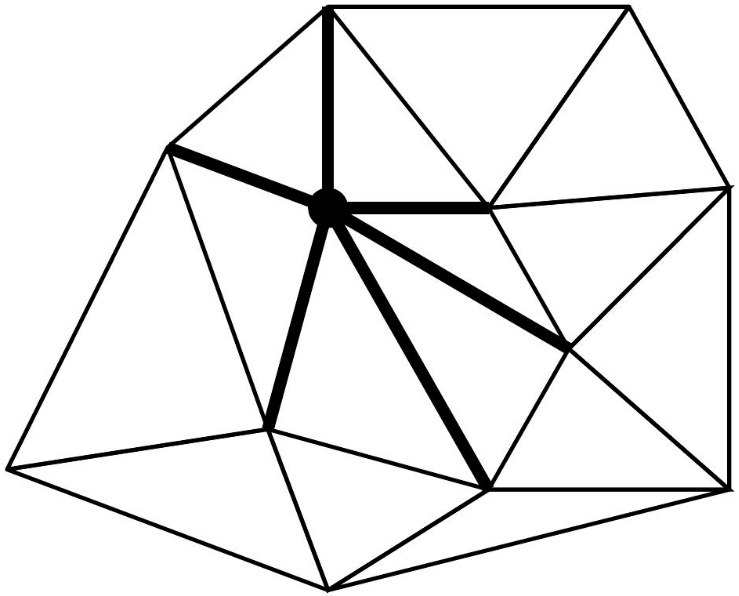
The Algorithm : Adding New Point

- The set of deleted triangles collectively form an insertion polygon, which is left vacant by the deletion of these triangles.
- Insertion polygon is shown in the figure.



The Algorithm : Adding New Point

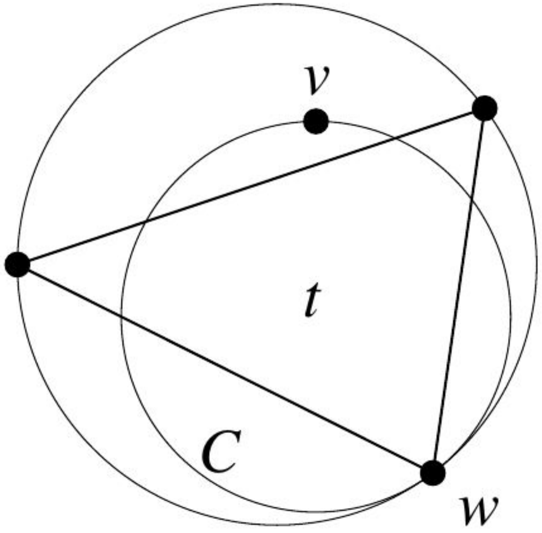
- The Bowyer-Watson algorithm connects each vertex of the insertion polyhedron to the new vertex with a new edge.



Correctness

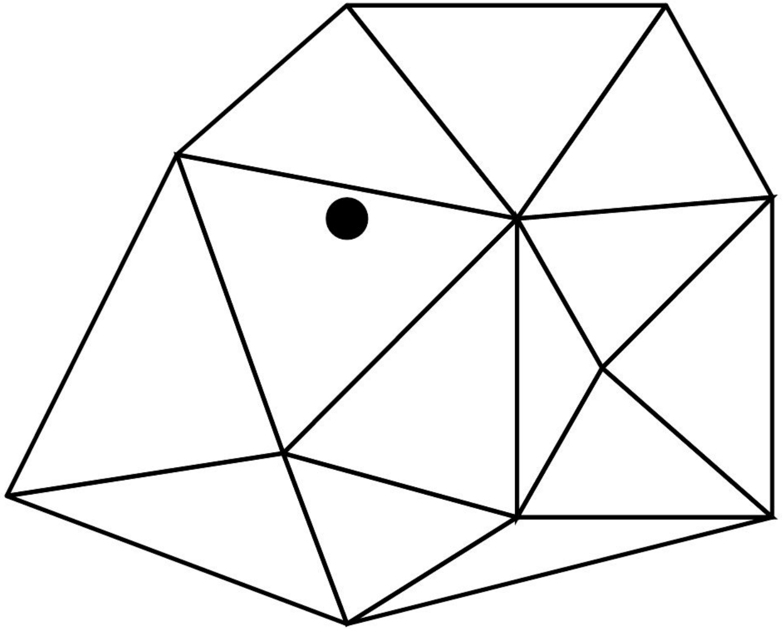
Theorem: Let **v** be a newly inserted vertex, let **t** be a triangle that is deleted because its circumcircle encloses **v**, and let **w** be a vertex of **t**. Then **vw** is Delaunay.

Proof: The circumcircle of **t** encloses no vertex but **v**. Let **C** be the circle that passes through **v** and **w**, and is tangent to the circumcircle of **t** at **w**. **C** is empty, so **vw** is Delaunay.



Data Structures

- List of Triangles
- Triangle
  - List of **vertices**
  - List of **edges**
  - List of **neighbours** - pair(triangle, edge)
  - **Circumcircle** (radius, center)
  - Boolean value **isDeleted**

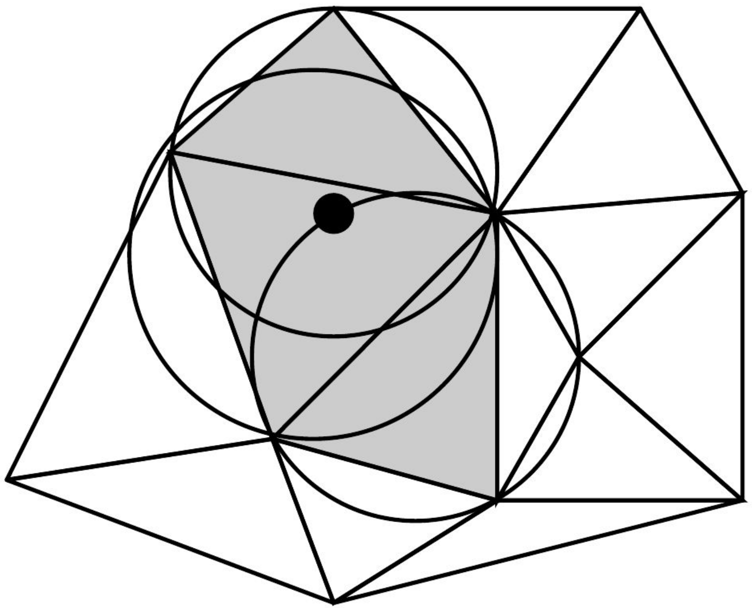


Finding T_{bad}

- **Brute Force:** Check all triangles O(n).

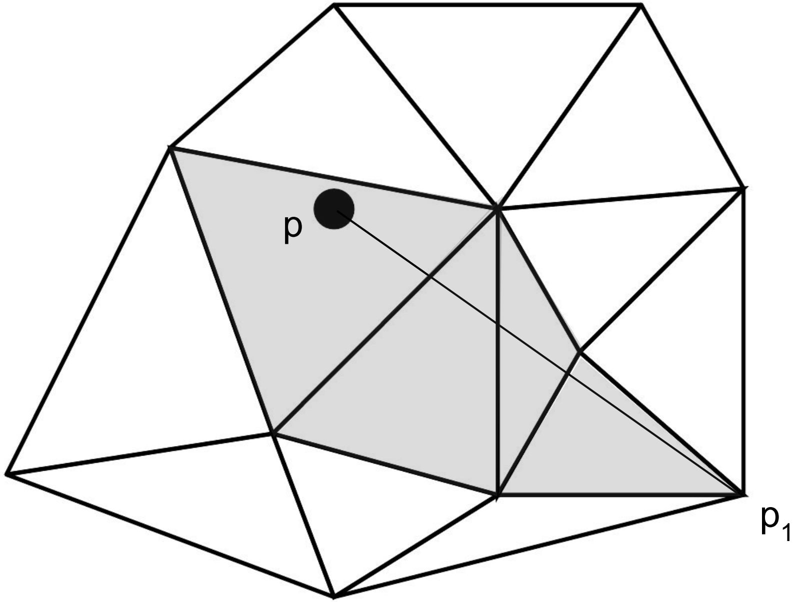
OR

- Find the triangle in which the new vertex lies using **Green-Sibson Method**.
- Perform **tree search** through the triangles looking for other triangles to be deleted.



Green-Sibson Method

- Green and Sibson propose taking a random point, say p_1 , and performing a walk to the query point p.

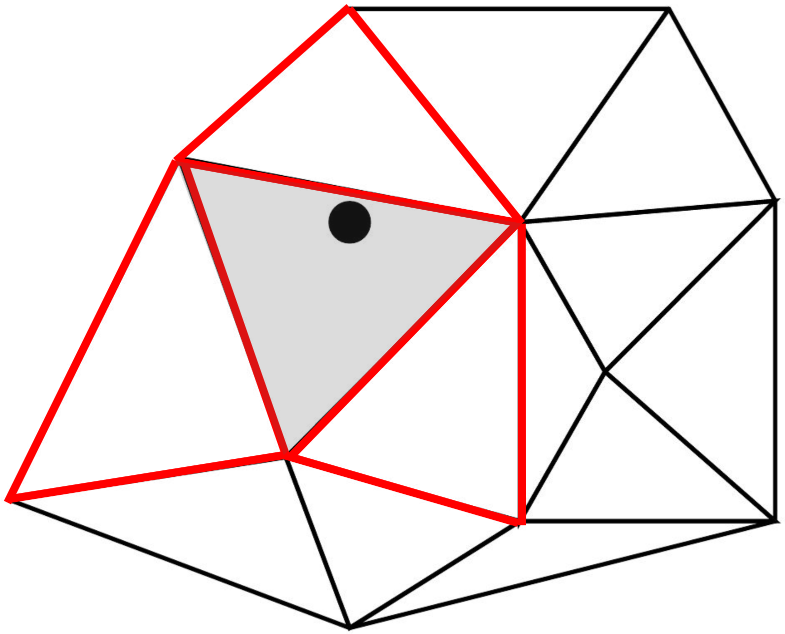


Green-Sibson Method

- The complexity of Green–Sibson method is bounded by degree(p_1, DT_n), the degree of p_1 in the Delaunay triangulation DT_n for $p_1, \ldots, p_n$ (to find the starting triangle in the Delaunay triangulation) plus N*, the number of triangles visited by (p, p_1) in DT_n.
- **Theorem:** The expected complexity of the Green–Sibson method for a random Delaunay triangulation and an independent and uniformly distributed query point on $[0, 1]^2$ is $\Theta(\sqrt{n})$.

Tree Search

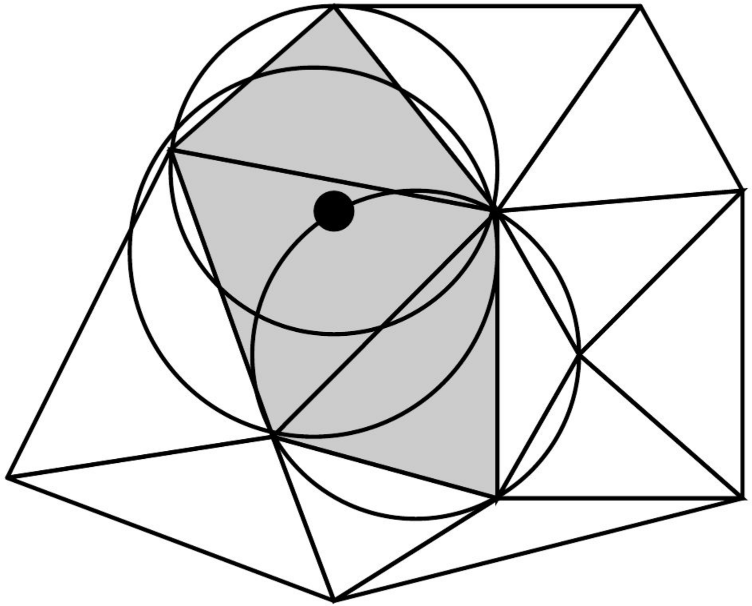
- Each triangle in Delaunay Triangulation except the ones on the boundary share their edges with other triangles (neighbours).
- A breadth first search like algorithm is used to find the triangles belonging to T_{bad}.
- Runtime is proportional to number of triangles in T_{bad} as the number of neighbours of a triangle is at most 3.



Size of T_{bad}

- $|T_{bad}|$ α degree(p_i, DT_i).
- Given k (dimension), expected degree of a vertex in Delaunay Triangulation is constant.
- In 2 dimensions the Euler-Poincare formula (V - E + F = 2) can be used to prove that,

  E[degree(p, DT)] = 6



Runtime of Algorithm

1. Add Super Triangle (p_{-1}, p_{-2}, p_{-3}) ------------- O(1)
2. For each i in 0 to n-1 ------------------------------ O(n) *
   a. Find the triangle in which p_i lies ------------------ $O(\sqrt{(i+3)})$
   b. Find T_{bad} -- O(1)
   c. Find boundary vertices of T_{bad} -------------------- O(1)
   d. Delete T_{bad} --- O(1)
   e. Add edges from p_i to boundary vertices. ----------------------------- O(1)
3. Delete all the edges connected to Super Triangle (p_{-1}, p_{-2}, p_{-3}). ------ O(n)

Analysis

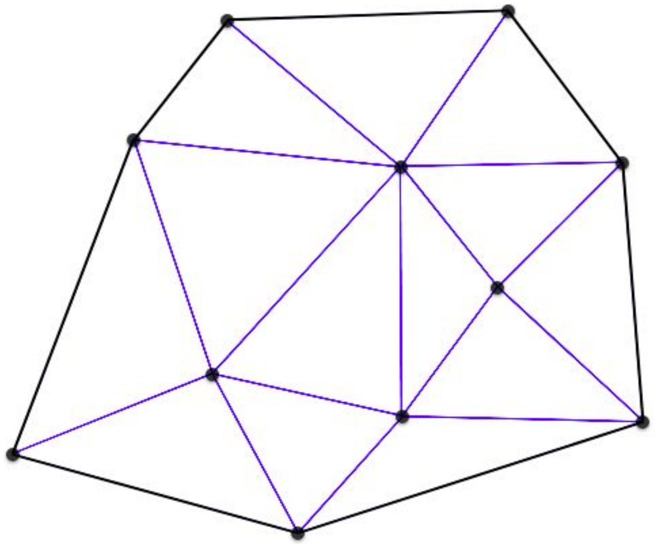
- In 2 dimension,
  - Expected Runtime: $O(n\sqrt{n})$
  - Space Complexity: $O(n)$

    - As addition of each new vertex creates $O(1)$ new triangles.

- The algorithm can also be extended to k-dimensions, with expected runtime of $O(n^{(1+1/k)})$.

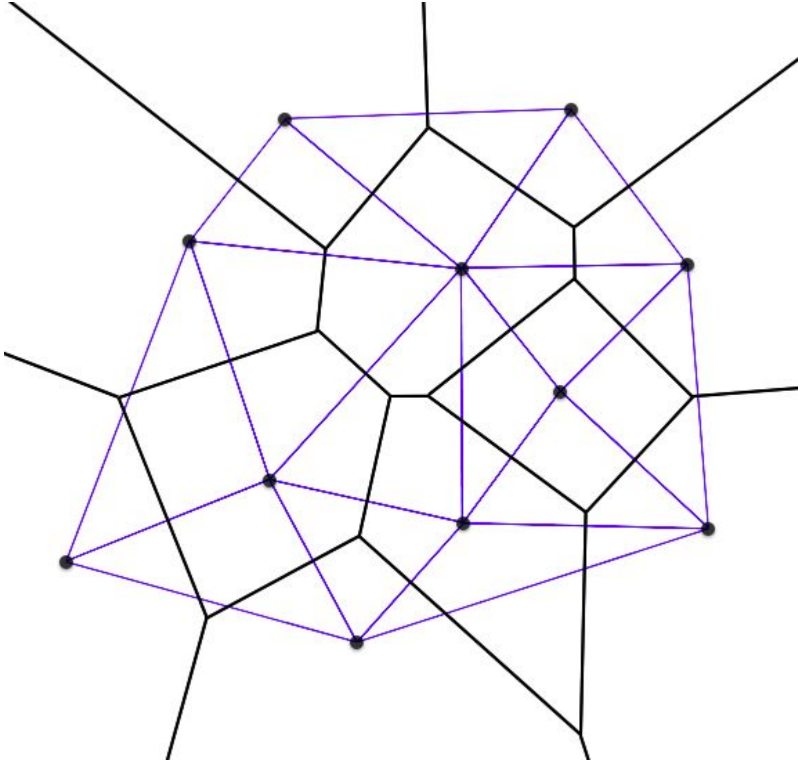
Convex Hull

Every edge that lies on the boundary of the convex hull of a vertex set and has no vertex in its interior is Delaunay.



Voronoi Diagram

- Vertices in Voronoi Diagram are the circumcenters of the triangles in Delaunay Triangulation.
- If we join the circumcenters of neighbouring triangles in Delaunay Triangulation, we get the Voronoi Diagram.



Demo

https://poojab01.github.io/DelaunayViz/