CPS234 Computational Geometry	**October 04, 2005**

Lecture 11: Voronoi diagram, Delaunay triangulation

Lecturer: Pankaj K. Agarwal *Scribe: Urmi Majumder*

11.1 Introduction

Voronoi diagram is the most popular among different geometric structures. In fact, it is one of the most versatile ones and has applications in geography, physics, astronomy, robotics and many more fields. It is also closely related to another important geometric structure, the Delaunay triangulation.

11.2 Voronoi diagram

11.2.1 Definition

Let $S \subseteq \mathbb{R}^2$ be a set of n distinct points(also called sites) in a plane.
For $p \in S$, we define the Voronoi region of p as,

$$\text{Vor}(p) = \{x \in R^2 : ||xp||^1 \leq ||xq||, \forall q \in S\}$$

We also define the Voronoi diagram of S $(VD(S))$ as the decomposition of $\mathbb{R}^2$ induced by Vor(p), $p \in S$[Figure 11.1].

We'll now take a closer look at the Voronoi diagram. First we'll study the structure of a single Voronoi cell. For two points p and q in the plane, we define the bisector l_{pq} of p and q as the perpendicular bisector of the line segment $\overline{pq}$. l_{pq} splits the plane into two half-planes. We denote the open half-plane that contain p by $h(p,q)$ and the open half-plane that contains q by $h(q,p)$. Notice that $x \in h(p,q)$ iff $||xp|| \leq ||xq||$[Figure 11.2a]. From this we can say,

$$\text{Vor}(p) = \bigcap_{q \in S \setminus \{q\}} h_{pq}$$

Consider the edge e_{pq} between Vor(p) and Vor(q). Note that,

$$\forall x \in e_{pq}, ||xp|| = ||xq|| < ||xr||, \forall r \in S \setminus \{p,q\}$$

Now consider the vertex v_{pqr}(point where Vor(p), Vor(q) and Vor(r) intersect). Let $v_{pqr} = x$. Then,

$$||xp|| = ||xq|| = ||xr|| < ||xs||, \forall s \in S \setminus \{p,q,r\}$$

v_{pqr} is also the circumcenter of p, q and r and the circle does not contain any other points from S.

[1] Note that $||xp||$ is the shorthand notation for Euclidean distance between the points x and p

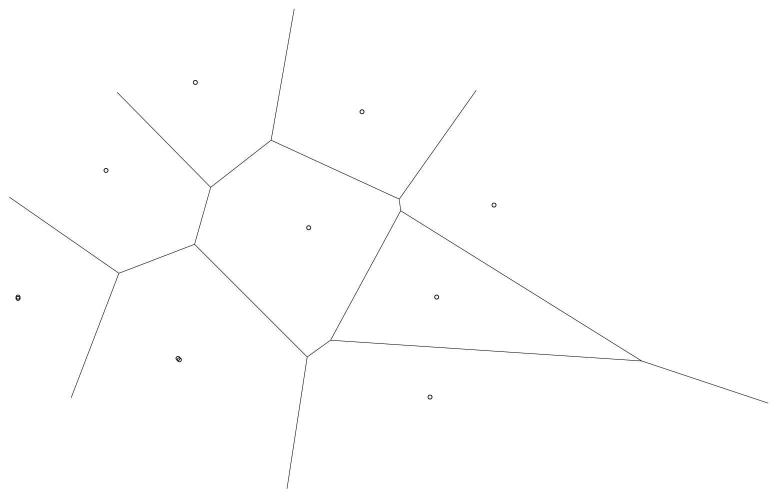


Figure 11.1: Voronoi diagram of S

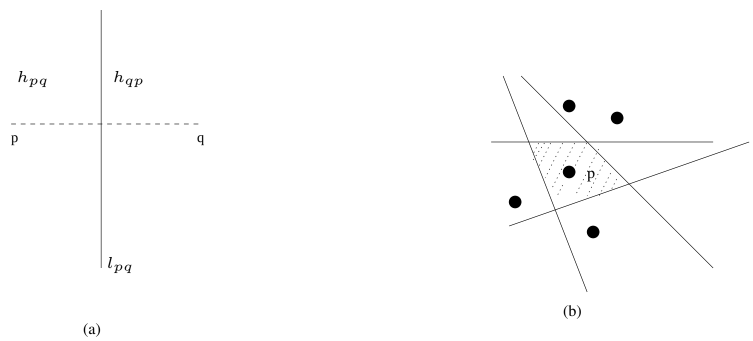


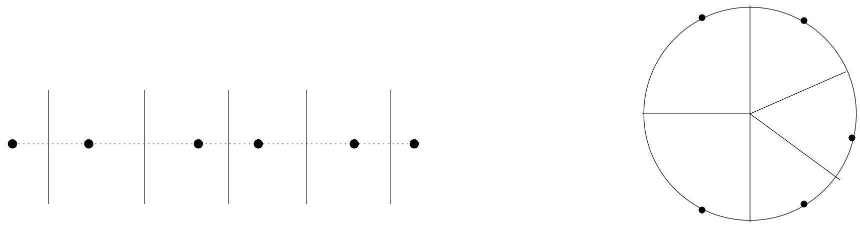
Figure 11.2: (a)Open Half-Planes of p and q (b)Voronoi Cell containing point p

Degenerate Cases

Voronoi diagram for collinear points For a set of n collinear points, the Voronoi diagram consists of $n-1$ parallel lines[Figure 11.3a].

Voronoi diagram for co-circular points For a set of co-circular points, the Voronoi diagram looks like pie slices[Figure 11.3b].

So, we'll make the general position assumption where (i)no three points are collinear and (ii)no four points are co-circular. This essentially means that no Voronoi polygon has parallel edges(from (i)) and degree of each vertex of a Voronoi polygon is at least three.



(a) (b)

Figure 11.3: (a)$VD(S)$, when all the points in S are collinear (b)$VD(S)$, when all the points in S are co-circular

11.2.2 Basic Properties

Size Complexity Size Complexity of $VD(S) = n_v + n_e + n_f$ where n_v is the # of vertices, n_e is the # of edges and n_f is the # of faces of $VD(S)$. Note that

$$n_f = n$$

For any connected planar embedded graph, the Euler formula gives us $n_v - n_e + n_f = 2$.
Moreover, every edge in the Voronoi diagram has exactly two vertices, so if we sum the degrees of all the vertices we get twice the number of edges. Because every vertex has degree at least three we get $2n_e \geq 3(n_v + 1)$ Solving the above equations, we get the following bounds

$$n_v \leq 2n - 5$$

$$n_e \leq 3n - 6$$

11.3 Delaunay triangulation

11.3.1 Definition

The Delaunay triangulation, $DT(S)$, is the dual graph of the Voronoi diagram $VD(S)$. This graph $DT(S)$ has a node for every Voronoi cell and has an edge between two nodes if the corresponding cells share an edge. Thus $\overline{pq}$ is an edge in $DT(S)$ iff Vor(p) and Vor(q) share an edge. Note that the degree of each vertex in Vor(S) is three and hence it is a triangulation[Figure 11.4].

11.3.2 Basic Properties

Some of the basic properties of Delaunay triangulation are listed below

i. $pq \in DT(S) \iff \partial\text{Vor}(p) \cap \partial\text{Vor}(q)$ is an edge of $VD(S)$

ii. $\Delta pqr \in DT(S) \Leftrightarrow v_{pqr}$ is a vertex of $VD(S)$

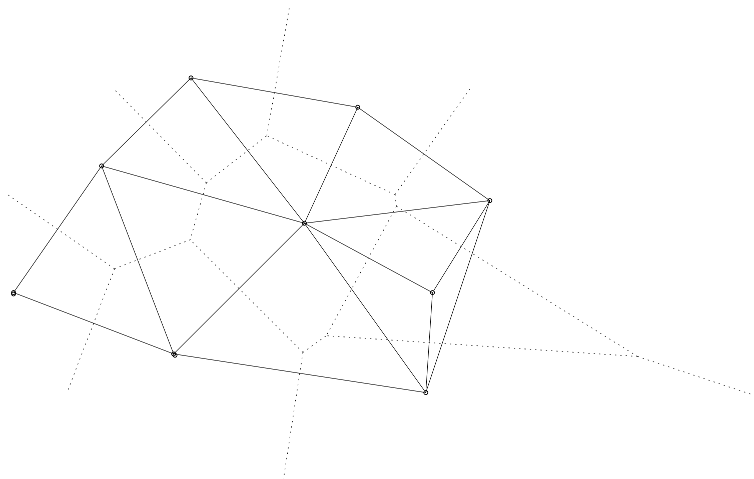


Figure 11.4: Delaunay triangulation of S

iii. v_{pqr} is the circumcenter of Δpqr

iv. $DT(S)$ is a triangulation of S such that the circumcenter of each triangle is empty i.e. no input point lies inside it.

v. A pair $pq \in DT(S) \Leftrightarrow \exists$ a circle C_{pq} passing through $\overline{pq}$ such that $interior(C_{pq}) \cap S = \emptyset$

11.4 Application

Voronoi diagram has several real world applications from anthropology to zoology. In addition, it has applications within the field of computer science, particularly computational geometry. We'll describe just a few examples.

11.4.1 Answering Nearest Neighbor Queries

A common type of query in Sensor Networks, Wireless Networks is to find the nearest neighbor object to a given point in space. This is used to maintain connectivity information. To answer this query, we first compute the Voronoi diagram and do some preprocessing for answering point location queries in $O(\lg n)$ time. Note that the query time in this case is $O(\lg n)$ and the space requirement. is $O(n)$.

We can answer the nearest neighbor queries efficiently with this procedure:
The Delaunay triangulation of the point set is represented as a weighted graph where the vertex set $V = S$, S being the point set. The weight of each edge for planar topology is $w(\overline{pq}) = ||pq||$. The goal is to compute the shortest path in this graph. Let the shortest path from p to q be denoted as $\delta(p, q)$. Now $\delta(p, q) \leq 4||pq||$, thus

Delaunay triangulation is widely used in routing because it is not required to maintain the entire connectivity information.

Spanners A spanner $G^{'}$ of a dense graph G is a small subgraph of G, such that a shortest path in G is approximated by a path in $G^{'}$. More precisely, given a graph G and parameter t, a t-spanner $G^{'}$ of G has the same vertices, and perhaps fewer edges, but the distance between a and b in $G^{'}$ is within a factor of t of the distance between any two vertices a and b in G. $DT(S)$ is a 2.43-spanner of $G(S)$. Spanners can be used for nearest neighbor queries.

11.4.2 Euclidean Minimum Spanning Tree

The **Euclidean Minimum Spanning Tree** or EMST(S) is a *minimum spanning tree* $G(S, S \times S)$ of a set of points Sin the plane (or more generally in $\mathbb{R}^n$), where the weight $w(p, q)$ of the edge between each pair of points p and q is the *Euclidean distance* between those two points ie. $w(p, q) = ||pq||$.

The simplest algorithm to find an EMST, given the n points, is to actually construct the complete graph on n vertices, which has $n(n-1)$ edges, compute each edge weight by finding the distance between each pair of points, and then run a standard minimum spanning tree algorithm on it. Since this graph has $O(n^2)$ edges, constructing it already requires $O(n^2)$ time. Even using a relatively simple minimum spanning tree algorithm such as *Prim's algorithm* with a *Fibonacci heap* requires only $O(n^2)$ additional time to find the minimum spanning tree of this graph, bringing the total time to $O(n^2)$ time. This solution also requires $O(n^2)$ space to store all the edges.

A better approach to finding the EMST in a plane is to note that it is a subgraph of every Delaunay triangulation of the n points, a much-reduced set of edges. The algorithm is given below

> Compute the Delaunay triangulation which is a planar graph, and there are no more than three times as many edges as vertices in any planar graph, this generates only $O(n)$ edges. Computing $DT(S)$ takes $O(n \lg n)$ time.
>
> Label each edge with its length.
>
> Run Prim's algorithm or any variant on it to find a minimum spanning tree. Since there are O(n) edges, this requires at most $O((n + n) \lg n)$ or $O(n \lg n)$ time.

The final result is an algorithm taking $O(n \lg n)$ expected time and $O(n)$ space.

An obvious application of Euclidean Minimum Spanning Trees is to find the cheapest network of wires or pipes to connect a set of places, assuming the links cost a fixed amount per unit length. Another application of EMSTs is to approximating the *Traveling Salesman Problem* on a set of points obeying the triangle inequality, such as a set of points in the plane. This realistic variation of the problem can be solved within a factor of 2 by computing the EMST, doing a walk along its boundary which outlines the entire tree, and then removing any duplicate vertices from this walk. This is a constant-factor approximation algorithm. EMST is also used in pattern recognition.

11.4.3 Mesh Generation and Surface Reconstruction

A key step of the finite element method for numerical computation is mesh generation. A domain is given and the goal is to partition it into simple "elements" meeting in well-defined ways. Meshes are often computed using *quadtrees* or by *Delaunay triangulation* of point sets.

A common method for the reconstruction of a geometric figure given a set of sample points is the use of a triangulation algorithm to connect the points and find the convex hull. *Delaunay triangulation* procedures have been used in the reconstruction of 3D geometric figures. The use of Delaunay triangulations is particularly suited when we do not want to force any constraints on the set of points to be connected. Besides, Delaunay triangulations have some interesting properties as optimal equiangularity and uniqueness (in $\mathbb{R}^2$).

11.4.4 Interpolation and Extrapolation

In many areas ranging from cartography to molecular imaging and modeling, one common problem is to fit a function or surface to a collection of scattered data points. Interpolation refers to finding values for points between the given points (i.e. inside their convex hull); if the function should extend over a wider region of the input domain the problem is instead referred to as extrapolation. The method chosen should depend on the properties one desires the resulting surface to have; many interpolation methods are based on *Voronoi diagrams* and *Delaunay triangulations*.

A piecewise constant approximation such as the one used in rainfall estimation, can be found by simply choosing the function value in each Voronoi cell to be that of the cell's generating site. The Delaunay triangulation itself provides a piecewise linear continuous function, defined within the convex hull of the input, that minimizes a certain energy function. *Natural neighbor interpolation* is defined for each point p by adding p as a site to the Voronoi diagram $VD(S)$ of the original sites set S, and averaging the sites' values weighted by the fraction of the cell for x previously covered by each other cell. This somewhat complicated procedure results in a continuous function smooth everywhere except at the original sites. Mathematically, we want to compute a function $f : S \longrightarrow \mathbb{R}$ by interpolation. Define $f(p) = \sum_{q \in S} f(q)w(q)$ where $w(q) = \frac{Area((\text{Vor}(p,S \cup p) \cap \text{Vor}(q,S))}{Area(\text{Vor}(p,S \cup p))}$, Vor$(p, S)$ being the Voronoi polygon of $p \in S$.

11.4.5 Motion Planning

In a typical motion planning problem, a two-dimensional map of the complicated region in which a robot will be operating, is given. Such a map includes a variety of polygonal obstacles that are to be avoided. To determine paths along which the robot can safely move through this environment, an approach based on the generalized Voronoi diagram for a planar region with specified obstacles is very popular among the motion planning community. Once this diagram has been constructed, one can search it to find robot paths that pass, with maximal clearance, around the obstacles.

The Algorithm

The two-dimensional region in which the robot moves will contain buildings and other types of barriers, each of which can be represented by a convex or concave polygonal obstacle.

To find the generalized Voronoi diagram for this collection of polygons, one can either compute the diagram exactly or use an approximation based on the simpler problem of computing the Voronoi diagram for a set of discrete points.

Once this complicated Voronoi diagram is constructed, eliminate those Voronoi edges which have one or both endpoints lying inside any of the obstacles. The remaining Voronoi edges form a good approximation of the generalized Voronoi diagram for the original obstacles in the map.

In this Voronoi diagram, locate the robot's starting and stopping points and then compute the Voronoi vertices which are closest to these two points. Use straight lines to connect the robot's starting and stopping points to these closest vertices. Special consideration is given to those situations in which one or both of the connecting straight lines pass through an obstacle.

Once the starting and stopping vertices on the Voronoi diagram have been determined one can use a standard search, such as Dijkstra's Algorithm, to find a "best" path which is a subset of the Voronoi diagram and which connects the starting and stopping vertices. This path can then be expanded to a path between the robot's original starting and stopping points. The method generates a route that for the most part remains equidistant between the obstacles closest to the robot and gives the robot a relatively safe path along which to travel.

11.4.6 Ecology

Geographic information systems(GIS) have become increasingly useful tools in many natural resource disciplines, including plant ecology. The ability to track vegetation change through time and to make predictions about future vegetation change are just two of the many possible uses of GIS. Delaunay triangulation is a part of most GIS and is often used for creating vegetation maps. For instance, in a surveying and mapping of plant communities on featureless terrain, a method of detailed vegetation mapping has been devised for use on undulating terrain where the combination of vegetation communities, topography, and other factors makes it difficult to locate a position on typical maps. The merged location and vegetation data set for individual points are then imported into a GIS to map point vegetation data accurately. Delaunay triangulation is used to produce a final vegetation map.