The geometry of circles: Voronoi diagrams, Möbius transformations, Convex Hulls, Fortune's algorithm, the cut locus and parametrization of shapes

David Dobkin William Thurston

1986

Abstract

The geometry of circles in the plane is inextricably tied with the group of Möbius transformations, which take circles to circles. This geometry can be seen in a more symmetric after transforming the plane to the sphere, by stereographic projection. Interpretations will be discussed for Voronoi diagrams, Delaunay triangulations, *etc.* from this point of view.

Fortune's algorithm for constructing the Voronoi diagram may be interpreted as a method of calculating the intersection of a set of half-spaces bounded by planes tangent to a sphere. The convex hull is ordered in terms of inclusion in the "horizon cone" for a point moving along a line tangent to the sphere.

Voronoi diagrams and convex hulls are closely related to the cut locus for a curve (or more general set) in the plane. A modification of Fortune's algorithm for calculating the cut locus of a smooth curve is discussed.

Notes for Dobkin-Thurston course in geometry and computer graphics

1.1 The Voronoi diagram and the Delaunay triangulation

One definition of the Voronoi diagram for a collection of points (called *sites*) in the plane describes it as the set of points in the plane which has a minimal distance from more than one site.

A better definition describes the *Voronoi diagram* as a certain mathematical structure, rather than as a set. From this point of view, the *Voronoi diagram* is a cell division, that is, a decomposition of the plane as a union of 2-*cells*, whose interiors are joint and homeomorphic to the open 2-disk or (synonomously) 2-cell. The boundary of any 2-cell is expressed as a union of 1-cells, whose interiors are disjoint and homeomorphic to the open 1-disk, and similarly, the boundary of each 1-cell is expressed as a union of 0-cells. Each cell of the Voronoi diagram is convex. Note that some of the cells in this situation are unbounded and noncompact. The Voronoi diagram as a cell-division is easy to define: the 2-cells consist of a collection of points with a common nearest neighbor, the 1-cells consist of a collection of points with a pair of nearest neighbors in common, and the 0-cells consist of those points with three or more nearest neighbors. We shall think of the Voronoi diagram with this second meaning. The subtlety of computing the Voronoi diagram arises from the problem of finding its structure, not from the problem of locating points once the structure is understood.

It is easy to check that the cells of the Voronoi diagram are all convex. Note that this property depends on the particular geometry of the plane. In fact, with a more general Riemannian metric in the plane (such as on a curved surface in space), the Voronoi diagram almost never has convex cells, and it is is not in general even a cell-division: Voronoi regions are not necessarily simply-connected, and two Voronoi regions can meet along a disconnected set.

It is also easy to check that when the sites are in general position (this means with the exception of an unlikely set of choices for sites) the vertices of the Voronoi diagram all are incident to three edges.

For any cell-division of a compact n-manifold without boundary, there is a dual cell structure, where each k-cell is replaced by an $n-k$-cell transverse to

it. For a situation such as the one we have, where some cells are unbounded, this same process yields a cell-division not for the entire manifold, but for a compact part of it. This process, in general, yields a curvilinear dual cell-division, even when you begin with a cell-division by convex cells.

In the case of the Voronoi diagram, it turns out that its dual, the Delaunay triangulation, can be straightened out so that it consists of actual cells. The Delaunay triangulation is actually a triangulation only in the generic case, when no vertices of the Voronoi diagram are incident to four or more edges.

There is another variation on the notion of the Voronoi diagram, known as the *most distant Voronoi diagram*. This is a decomposition of the plane according to which of the sites is furthest away. It depends only on those sites which are on the boundary of the convex hull of all sites.

1.2 Circles

The definition above of the Voronoi diagram was made in terms of distances in the plane. Another description, however, can be made in terms of circles:

The vertices of the Voronoi diagram are in one-to-one correspondence with collections of sites such that there is a circle passing through the sites in the collection, with no other sites either on the circle or in the interior of the circle.

Edges of the Voronoi diagram are in one-to-one correspondence with pairs of sites through which there is a circle intersecting no other sites and which have no other sites in their interior.

Finally, two-cells of the Voronoi diagram are in one-to-one correspondence with sites.

Thus, the combinatorics of the Voronoi diagram depends only on the geometry of circles in the plane.

In other words, we could forget about the notion of a straight line and forget about the notion of distance, but still be able to reconstruct the combinatorial structure of the Voronoi diagram by studying circles in the plane. In order to reconstruct the precise diagram, we need to know exactly where the vertices and edges are: this information does make use of the measure of distance in the plane, or at least, the knowledge of where the center of a circle is.

For brevity, let us say that a circle is a *maximal complementary circle* if its inside is in the complement of the union of sites, and if it is not contained inside any other circle with the same property. The union of the edges and

vertices of the Voronoi diagram is in one-to-one correspondence with maximal complementary circles. The actual points on the edges and vertices are the centers of these circles.

1.3 Fortune's algorithm

The algorithm of Steve Fortune for computing the Voronoi diagram using a simple scan line method can be seen clearly in terms of circles. The method is to construct inductively the combinatorial structure of the collection of maximal complementary circles below a line of the form $y =$ constant.

The more obvious scan line approach would have been to construct the Voronoi diagram itself up to a certain y-coordinate. In watching the circles instead of their centers, no backtracking is required.

For any value of y, there are ordinary vertices of the diagram, which come from maximal complementary circles touching three or more sites, and there are also "active" vertices, which touch two sites and the boundary line. There are also ordinary edges, which may have 0, 1 or 2 active endpoints, and active edges which separate a site from the scan line. At any time y, the active edges together with the active vertices form a linear graph, which one can visualize as waves moving upward.

Keep in mind that the edges of the graph are families of circles, not families of points, and the vertices of the graph are circles, not points. Mapped to edges in the plane, the ordinary Voronoi edges are straight, while the active Voronoi edges are curved (pieces of parabolas).

As the scan line is moved upward, two kinds of events can occur.

First, the scan line can cross a new site. In such a case, a new ordinary edge is created (with active endpoints) which shields the new site from its nearest neighbor below. This causes an active edge to be subdivided into three active edges. Such events are predicted ahead of time, if all sites are sorted ahead of time according to y-value and put in a priority queue.

Second, an active edge can shrink to a point, which becomes an ordinary vertex. For each active edge e, the time (if any) at which it will shrink to a vertex is precomputed from the geometry of the positions of the sites which e and its two neighbors separate from the scan line. If either of its two neighbors changes, then this information must be updated. This information can be kept on the priority queue as well.

Only $O(n)$ events are ever inserted in the priority queue, so it accounts for

$O(n \log(n))$ running time The linear graph of active edges and vertices must be kept up to date: this also takes $O(n \log(n))$ time for $O(n)$ total insertions and deletions if it is implemented with a dynamically balanced binary tree.

Fortune described this processing in terms of a remapping of the plane to itself; his description is equivalent to this one. The remapping amounts to associating with each maximal complementary circle the topmost point on it, rather than its center.

1.4 Project 1: $O(n \log n)$ implementation

The code which Fortune wrote does not actually implement his algorithm in $O(n \log(n))$ worst case time. This code does not take care to balance the binary tree representing the active linear graph. This is irrelevant in the case of a randomly-distributed collection of points, but it becomes a significant factor in the case of sites distributed around a smooth closed curve: in such a case, the tree tends to become very unbalanced, and the algorithm might take quadratic time.

Experiment with different data structures to represent both the active linear graph and the priority queue.

This may be one application where splaying sometimes beats in practice as a method of representing the active linear graph: in the case of a points around a smooth convex curve, all the insertions will be close to each other, and the deletions will be close to each other.

Test the different implementations with rpts, and backit having different parameter values. The qualitative character of the collection of points changes dramatically as the parameter is varied.

1.5 Project 2: Smaller size implementation

Write a version making more efficient use of memory. The program as it stands has a huge size, and a built-in limit of 10,000 points. One method is to use malloc to get memory as needed.

Another idea is to make a version which acts as a filter. It could accept as input a list of sites already sorted numerically by y-value, and output graphical information as it is becomes known. For random input, the size necessary should be cut down to roughly the square root of the size of keeping everything in memory at once.

1.6 Möbius transformations

There is a group of transformations which is fundamental to the study of properties of circles in the plane: the Möbius group. This group is generated by the transformation of *inversion* in a circle C: if C is centered at a point $X \in \mathbf{R}^2$ and has radius r, then inversion in C send a point y to the point y' which is on the ray from x through y at distance $r^2/d(x,y)$. In other words, y' is chosen so that the radius of the circle is the geometric mean of the distances from x to y and y'. Inversion is not quite well-defined: the point x is sent to infinity. However, if we adjoin a single point ∞ to the plane, this turns the plane into a sphere, and inversion is now defined everywhere and continuous. Inversion has the amazing properties that it sends a circle to a circle — provided that we allow a straight line as a special "limiting" case of a circle. The straight lines are precisely those circles which go through ∞. Inversion through C sends any straight line to a circle through x, and any circle through x to a straight line.

Any circle orthogonal to C is sent to itself. Using this fact, one can describe inversion without mentioning straight lines and distance. To calculate the image of a point y in the plane by inversion through C, you construct any two circles through y and orthogonal to C. The image of y is the second point y' of intersection of these two circles.

Inversions generate the group of Möbius transformations, but they do not constitute the whole group. The group of all Möbius transformations has a subgroup of index 2, consisting of those which preserve the orientation of the plane. These are the transformations which can be expressed as a composition of an even number of inversions. This group has another, very convenient, notation, involving complex numbers. If we think of the Euclidean plane as the complex numbers $\mathbf{C}$, then the group of orientation preserving Möbius transformations is the same as the group of fractional linear transformations

$$\frac{az+b}{cz+d},$$

where a, b, c and d are complex numbers such that the determinant of the matrix

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

is not zero. Composition of fractional linear transformations corresponds to multiplication of matrices, a great convenience. Two matrices which dif-

fer by multiplication by a scalar have the same effects as fractional linear transformations.

Another way to think of this is that the complex numbers, together with a single point at infinity, forms the complex projective "line" $\hat{\mathbf{C}}$ (since it is one-dimensional over $\mathbf{C}$). The fractional linear transformations are the same as projective transformations.

It turns out that any triple of distinct points in $\hat{\mathbf{C}}$ can be mapped to any other triple of distinct points by a unique fractional linear transformation. There are two Möbius transformations accomplishing this: they differ by an inversion through the circle determined by the triple. This fact is not hard to verify by solving for the coefficients a, b, c and d. One convenient case of this is the mapping that sends an arbitrary triple $z_i \quad [i = 1, 2, 3]$ to ∞, 0 and 1. If z_4 is any fourth point, this transformation sends z_4 to the cross-ratio

$$\mathrm{XR}(z_1, z_2, z_3, z_4) = \frac{(z_0 - z_3)(z_1 - z_4)}{(z_0 - z_4)(z_1 - z_3)}.$$

The cross-ratio can be used as a convenient test for the relation of z_4 to the oriented circle determined by z_1, z_2 and z_3. If these three points are in counterclockwise order, so that the circle is positively oriented, then z_4 is inside this circle if and only if it has a positive imaginary part. If the circle is negatively oriented, the test is reversed.

Because the Voronoi diagram is determined by properties of circles, the Voronoi diagram should be invariant by any Möbius transformation — the only catch is that the notion of the "inside" of a circle is not invariant by Möbius transformations. In the combinatorial description of the Voronoi diagram in terms of circles, if we substitute "outside" for "inside" wherever it occurs, we get the combinatorial description of the most distant Voronoi diagram. The Voronoi diagram of a collection of sites after a Möbius transformation can be obtained by combining cells of the original Voronoi diagram and cells of the most distant Voronoi diagram.

Although the combinatorics of the Voronoi diagram transform reasonably under Möbius transformations, the actual picture in the plane does not. The actual edges of the Voronoi diagram map to circles in general, not to straight lines. Vertices of the Voronoi diagram also do not map to vertices. For purposes of computation, this is not a problem; the actual vertices and edges are easy to compute if the combinatorics is known.

1.7 Stereographic projection and Convex hulls

There is a strangeness in the planar picture of Möbius transformations which has to do with points going to infinity, and infinity going to finite points.

The picture is much clearer when we pass to a three-dimensional view. *Stereographic projection* is a transformation which sends the plane to a sphere, adding an extra point (the north pole) representing ∞. It is defined by resting a sphere S on the plane, then using projection through the uppermost point on S (the north pole).

Stereographic projection transforms transforms circles in the plane to circles on the sphere. One proof of this fact comes from its interpretation as a Möbius transformation in three dimensions: if you take a sphere S' centered at the north pole of S and tangent to the plane, inversion in S' sends S to a plane (since S passes through the center of inversion). The plane is tangent to S' where S and S' are tangent; in other words, this is our original plane, and the transformation is stereographic projection. To prove that a circle C in the plane goes to a circle on S, let D be any sphere which intersects the plane in C. Then the image of D by the inversion is a sphere, and the image of C is the intersection of D with S, so it must be a circle.

Möbius transformations of the sphere in $\mathbf{R}^3$ extend to projective transformations of 3-space (actually, projective 3-space $\mathbf{RP}^3$). To see the correspondence between Möbius transformations and projective transformations which preserve the sphere, note that a circle on the sphere determines a plane in space which contains it, and any plane which intersects the sphere determines a circle. A projective transformation can be constructed from a Möbius transformation by its action on these planes.

When a circle is transformed by a Möbius transformation, its center does not transform to the center of the image circle (in general). The three-dimensional picture does give a way to record the center of a circle invariantly. A circle on the sphere determines a point in $\mathbf{R}^3$: namely, if you take all the lines orthogonal to the circle and tangent to the sphere, they form a cone; they intersect in a single point, the apex of the cone. The circle is the "horizon" of the sphere from the point of view of this apex. Thus, the collection of points in $\mathbf{RP}^3$ outside the sphere are in one-to-one correspondence with circles on the sphere.

When a set of sites is transformed to the sphere, the Delaunay triangulation corresponds to the convex hull of the set in question. The boundary of the convex hull is made up of planes which cannot be pushed inward any

further: these intersect precisely in the maximal complementary circles.

The Voronoi diagram corresponds to the dual convex hull, formed by the intersection of the "inside" half spaces bounded by planes tangent to the sphere at the sites. (In general, for any subset A of a vector space V whose convex hull contains the origin, the *dual convex hull* of A is the set of vectors in V whose inner product is not greater than 1 for any element of A. For example, if the convex hull of a collection of points is an octahedron, its dual convex hull is the dual polyhedron, the cube).

Actually, the picture on the sphere has lost the distinction between inside and outside of a circle. To reconstruct the actual voronoi diagram, take the dual convex hull of the set of sites, form the intersection with the region of $\mathbf{R}^3$ below the north pole, and project to the plane. The vertices of the dual convex hull actually project to the true centers of the circles: the picture is not only combinatorially correct, it is correct on the nose.

There is a related three-dimensional picture for the Delaunay triangulation, coming from the paraboloid $z = x^2 + y^2$. In this form, you project the plane straight upward to the paraboloid and form its convex hull. The paraboloid picture is equivalent to the sphere picture: to get from one to the other, you can take a projective transformation which sends the north pole of the sphere to a point at infinity in $\mathbf{RP}^3$, where the vertical lines all meet. Stereographic projection goes over to vertical projection.

1.8 Fortune's algorithm in 3 dimensions

Fortune's algorithm in 3 dimensions becomes the following.

We are given a collection of sites on the sphere, and we are to find the dual convex hull. We build one of the much-touted aerospace planes which will take off like airplanes but fly like rockets at high altitude. We take off on a runway at the north pole, continuing in a straight line into space. As we fly away, we look backward at the earth, and keep track of what is visible in the cone inside the earth's horizon (which is a circle). The horizon on the earth is a circle, always passing orthogonally through the runway where we took off, which gradually sweeps over the earth's surface. Fortune's order of processing events is to keep track of portions of the dual convex hull as they enter this cone of view of the earth.

To extend the metaphor a bit further, we can think of the dual convex hull of the collection of sites as the region which is not visible to radars at any of the sites.

Unfortunately, the field of view never encompasses more than half of the earth's surface with this approach, but with our ever-increasing military budgets and the infinite technological advances which our leaders assure us will be accomplished by SDI (star wars), this is no problem. Let the aerospace plane continue on through infinity and come back from the other side. It continues to look behind itself with a superspy camera, and its field of view enlarges, sweeping over the entire surface of the earth. The plane returns, and makes a smooth landing.

The point of this method is that no part of a face of the dual convex hull is viewed until the point of tangency is viewed.

The cone under consideration may be thought of as generated by a circle sweeping over the earth's surface, always passing through the north pole and tangent to a given line there. The cone is the set of rays tangent to the earth and orthogonal to the circle, pointing to one side of the circle. Note that this cone never goes above the plane tangent to the earth at the north pole. For this reason, the method computes the Voronoi diagram and not the most distant Voronoi diagram.

1.9 Project 3: write a 3-dimensional version of Fortune's algorithm

Rewrite Fortune's code to make use of this three-dimensional picture. Construct the entire dual convex hull, not just the part corresponding to the Voronoi diagram. This can be done by putting the runway at an actual site (or in the planar picture, transforming $\mathbf{C}$ by a Möbius transformation so that a site is at ∞).

1.10 Alternative order of processing

The scenario above suggests an alternative. Instead of following a trajectory tangent to the earth's surface, the plane can follow any trajectory so long as its field of vision is always enlarging. The most straightforward method is to fly straight upward (provided we can correct problems with the shuttle so we have a vehicle which can take off vertically). The algorithm works just as before, by keeping track of portions of the dual convex hull which fall inside the cone from the shuttle to the earth. The shuttle proceeds through infinity, and lands at the antipodal point. In doing this, the cone sweeps

over a half-space bounded by the plane tangent to the point of landing; if the point of landing is a site, then the entire dual convex hull is obtained

In Fortune's terminology, this amounts to remapping the voronoi diagram so that each point is moved radially outward from the origin by a distance equal to its distance from the nearest site.

This approach has a clear application in the case that the set of sites is infinite: there is no problem of where to start, just where to stop. You stop when you have done enough.

Instead of a linear graph of active edges and vertices, there is now a circular graph of active edges and vertices, but the theory is much the same.

1.11 Project 4: Radial version

Write a version of Fortune's code to process events radially outward, either in 3 dimensions or in 2 dimensions.

1.12 Project 5: Edge testing

A triangulation is a Delaunay triangulation iff for each edge, the circumscribed circle for either adjacent triangle does not contain the other. If an edge fails this condition, then it can be removed; in such a case, the resulting quadrilateral is always convex, and the transverse edge inserted. If this process is continued, it will eventually terminate (as is easy to see from the convex hull picture).

Is there an efficient algorithm which makes use only of the edge test? In other words, assume one is given a triangulation of a convex subset of the plane. (For any collection of sites in the plane, there is a straightforward algorithm to find a triangulation for its convex hull: essentially Graham's algorithm for finding their convex hull; assume such a triangulation is input for the algorithm). To avoid cheating, design a program which is told only the combinatorial structure of the triangulation (not the location of vertices) and results of edge tests on edges. The program may change the triangulation at any time by an elementary move or rotation of any edge, and ask for edge tests at any time. Is there a strategy which takes less than $O(n^2)$ time? Perhaps $O(n \log(n)$ or even $O(n)$?

One potential strategy is to maximize independence of tests and moves. Form a queue of edges to be checked, initially having every edge not on the boundary. Each time an edge fails the test, perform a rotation on it, delete

(or mark for deletion) the four neighboring edges from their current position in the queue, and append them to the tail of the queue.

Test empirically how this, or other strategies of your devising, performs (measured by the number of tests and the number of rotations). Check it with initial points of varying qualitative type, as in rpts and backit.

2 The cut locus

The *cut locus* of a curve γ in the plane is the set of points x inside γ (or more precisely, in the bounded component of the complement of γ) such that there is more than one point on γ with least distance to x. This definition is precisely the same as the definition for the Voronoi diagram, if γ is replaced by a discrete set of points. The terminology "cut locus" comes from differential geometry; it is also called the *skeleton* of a curve.