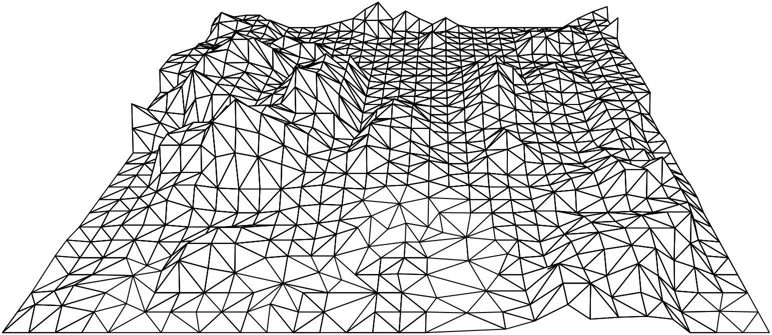
Delaunay Triangulation (chapter 9)

Elisha Sacks

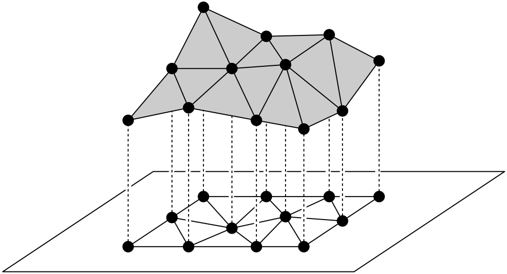
Additional material on Delaunay triangulation and meshing
Cheng, Dey, and Shewchuk. *Delaunay Mesh Generation*, Chapman and Hall, 2012.

Terrain Approximation



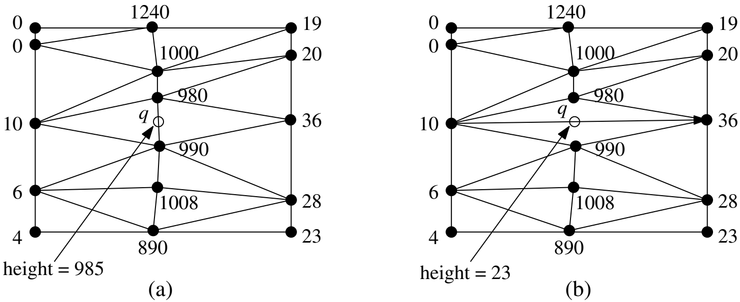
- Altitudes are measured at scattered points.
- The terrain is approximated with a triangle mesh.
- We will discuss smooth approximation in a later class.

Domain Triangulation



- Triangulate the projections of the points onto the *xy* plane.
- The corresponding 3D triangles comprise the mesh.

Triangulation Quality



- Triangles with large angles poorly interpolate the gradient.
- Small angles cause numerical problems, e.g in finite elements.
- The Delaunay triangulation maximizes the smallest angle.
- Delaunay refinement algorithms remove large angles and other bad shapes by adding vertices to Delaunay triangulations.

Gradient Interpolation Error

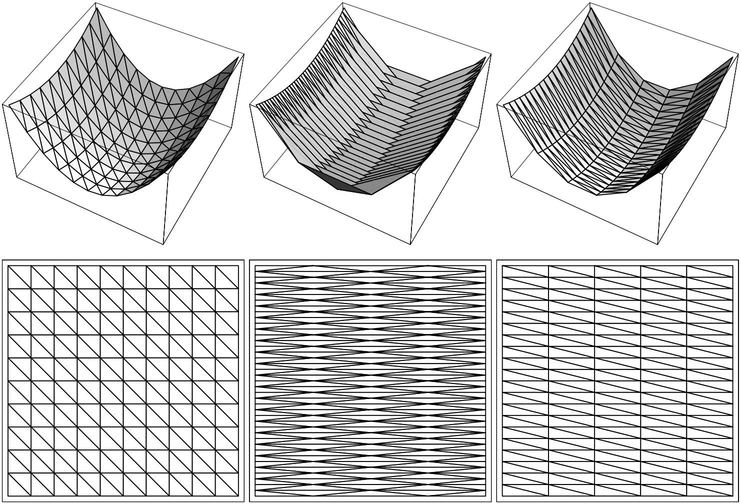
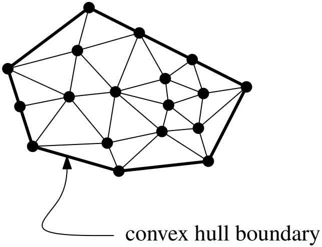


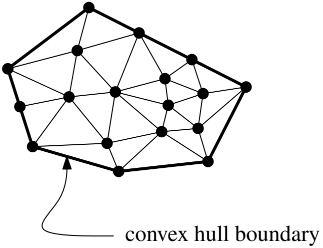
Figure 1.3: An illustration of how large angles, but not small angles, can ruin the interpolated gradients. Each triangulation uses 200 triangles to render a paraboloid.

Point Set Triangulation



- A *triangulation* of a point set is a subdivision whose vertices are the points and that has a maximal number of edges.
- The bounded faces are triangles because polygonal faces can be triangulated.
- The unbounded face is the complement of the convex hull of the points. A hull edge that contains points corresponds to multiple subdivision edges.

Complexity



Theorem 9.1 A triangulation of n points of which k are on the convex hull has $t = 2n - k - 2$ triangles and $e = 3n - k - 3$ edges.

Proof $e = (3t + k)/2$ because a triangle has 3 edges, the unbounded face has k edges, and an edge is incident on 2 faces.

The number of faces is $t + 1$, so $t = e - n + 1$ by the Euler formula.

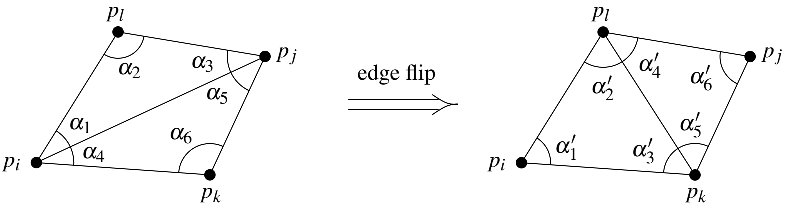
Substitute $e = (3t + k)/2$ to obtain $t = 2n - k - 2$.

Substitute t into $e = (3t + k)/2$ to obtain $e = 3n - k - 3$.

Angle Optimal Triangulation

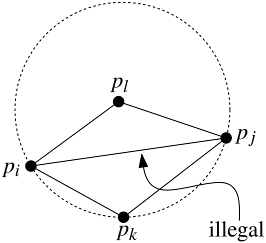
- The angle sequence of a triangulation is a list of the angles of its triangles in increasing order.
- Angle sequences are ordered lexicographically.
- A triangulation is angle optimal if no triangulation has a larger angle sequence.
- The smallest angle in the mesh is maximized.

Edge Flips



- Consider a triangulation with triangles $p_j p_i p_k$ and $p_i p_j p_l$.
- The edge $p_i p_j$ is illegal if the polygon $p_i p_k p_j p_l$ is convex and $\min \alpha_i < \min \alpha'_i$.
- An edge flip replaces $p_i p_j$ with $p_k p_l$, which increases the angle sequence.
- A triangulation is legal when it has no illegal edges.
- Flipping all the illegal edges yields a legal triangulation.
- An angle optimal triangulation is legal.

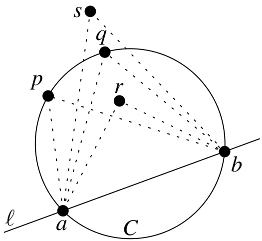
Illegal Edge Test



Lemma 9.4 An edge $p_i p_j$ is illegal iff p_l is in $C(p_i, p_j, p_k)$; equivalently, p_k is in $C(p_i, p_j, p_l)$.

- We will prove this using Thale's theorem.
- We saw the point-in-circle predicate in an earlier lecture.
- If the four points lie on a circle, neither edge is illegal.

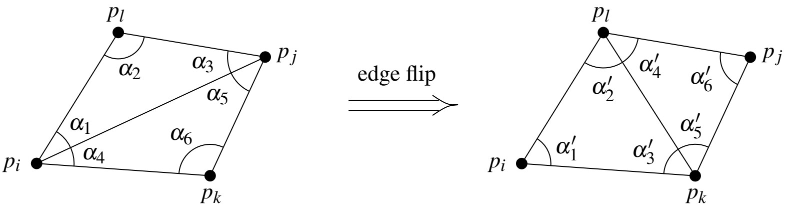
Thale's Theorem



Theorem 9.2 Let ℓ be a line through points a and b on a circle C. Let p, q, r, s be points on the same side of ℓ with p and q on C, r inside C, and s outside C.

$$\angle arb > \angle apb = \angle aqb > \angle asb$$

Proof of Lemma 9.4

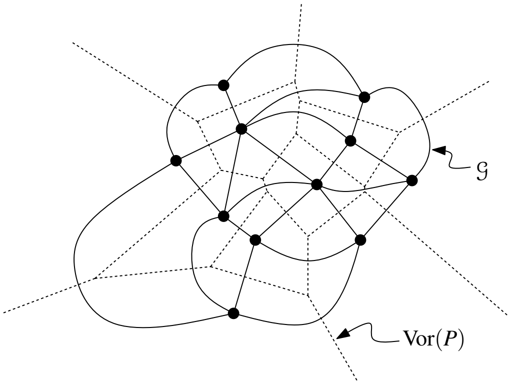


- Let p_l be in $C(p_i, p_j, p_k)$.
- $p_i p_k p_j p_l$ is convex, so $p_i p_j$ can be flipped.
- Every angle α'_i is larger than some angle α_j.
  - $\alpha'_1 > \alpha_1$ because $\alpha'_1 = \alpha_1 + \alpha_4$.
  - $\alpha'_2 > \alpha_5$ because p_l is in $C(p_i, p_j, p_k)$ and p_j is on it.
  - The other angles are analogous.
- Hence, $\min \alpha'_i > \min \alpha_i$ and $p_i p_j$ is illegal.
- For p_l outside $C(p_i, p_j, p_k)$ and $p_i p_k p_j p_l$ convex, a similar proof shows that $\min \alpha'_i < \min \alpha_i$ and $p_i p_j$ is not illegal.

Delaunay Triangulation

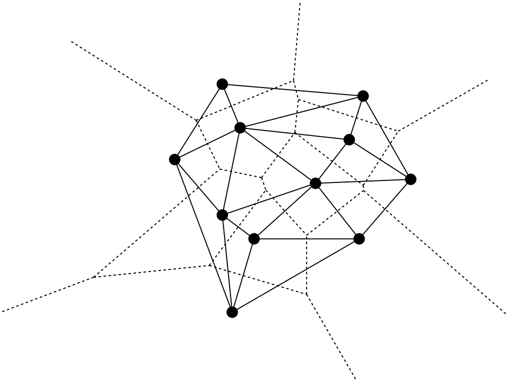
- Legal triangulations are closely tied to Voronoi diagrams.
- The dual graph of a Voronoi diagram is planar.
- A planar embedding yields the legal triangulations.
- These triangulations are called Delaunay triangulations.

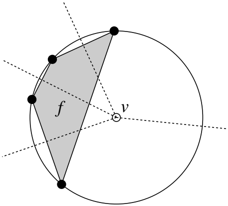
Dual Graph



- Consider the dual graph $\mathcal{G}$ of a Voronoi diagram $Vor(P)$.
- The vertices of $\mathcal{G}$ are the Voronoi cells.
- The edges of $\mathcal{G}$ connect the cells that share an edge.
- The duals of the edges incident on a vertex form a loop.

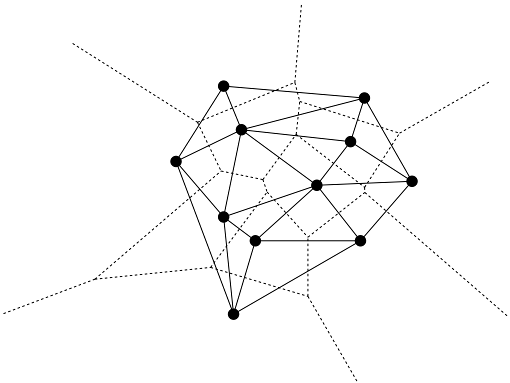
Delaunay Subdivision





- The Delaunay subdivision is the straight-line embedding whose vertices are the sites.
- The edge loops are convex polygons.

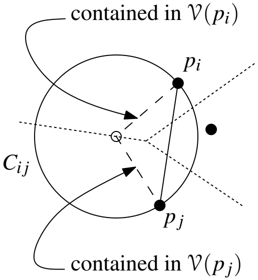
Empty Circle Conditions



Three sites are vertices of a face iff their circumcircle is empty.
Proof This is the condition for a Voronoi vertex.

Two sites form an edge iff they are on an empty circle.
Proof This is the condition for the dual edge to be Voronoi.

Planarity

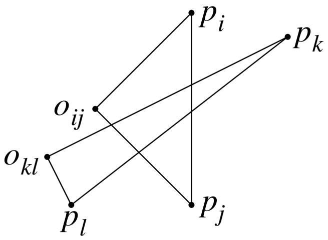


Theorem 9.5 The Delaunay subdivision is a plane graph.

Proof

- An edge $p_i p_j$ is in the Delaunay subdivision iff there exists an empty circle C_{ij} with p_i and p_j on its boundary.
- Let $t_{ij} = p_i p_j o_{ij}$ with o_{ij} the center of C_{ij}.
- The triangle t_{ij} is empty, o_{ij} is on the $p_i p_j$ Voronoi edge, $p_i o_{ij} \subset \mathcal{V}(p_i)$, and $p_j o_{ij} \subset \mathcal{V}(p_j)$.

Planarity Proof (continued)



- Suppose $p_i p_j$ intersects $p_k p_l$ with C_{kl}, t_{kl}, and o_{kl}.
- Since p_k and p_l are outside t_{ij}, $p_k p_l$ also intersects an edge incident on o_{ij} (here $o_{ij} p_j$).
- Likewise, $p_i p_j$ intersects an edge incident on o_{kl} (here $o_{kl} p_k$).
- The two triangles intersect at four points.
- An edge incident on o_{ij} intersects an edge incident on o_{kl} (here $o_{ij} p_j$ and $o_{kl} p_k$).
- Contradiction: these edges are in different cells, hence disjoint.

Delaunay Triangulation

- A Delaunay triangulation is a triangulation of the faces of the Delaunay subdivision.
- The sites are in *general position* when no four lie on an circle whose interior is empty of sites.
- The Delaunay triangulation is unique because the faces of the Delaunay subdivision are already triangles.
- A triangulation of the sites is legal iff it is Delaunay.
- Sites in general position have a unique legal triangulation.
- Degenerate sites has multiple legal triangulations with the same minimum angle.
- Every legal triangulation is angle optimal.

Delaunay Lemma

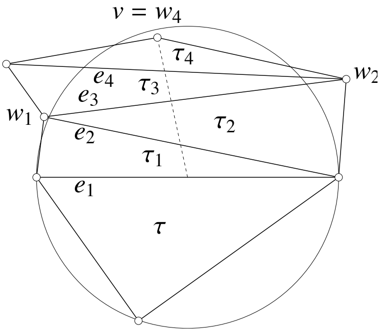
Theorem 9.8 A triangulation is legal iff it is Delaunay.

Proof A Delaunay triangulation is trivially legal.

- Consider an edge $p_i p_j$ incident on triangles $p_i p_j p_k$ and $p_j p_i p_l$.
- The circumcircle of $p_i p_j p_k$ is empty.
- The edge is legal because p_l is not in the circumcircle.

We will prove the converse differently than the textbook.

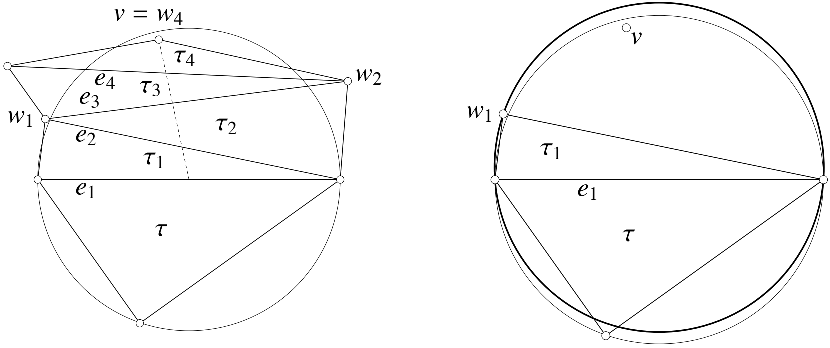
Legal Implies Delaunay



We will derive a contradiction from the assumption that a legal triangulation has a vertex v in the circumcircle of a triangle τ.

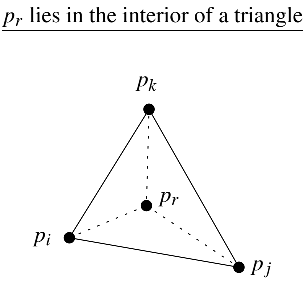
- Let e_1 be the edge of τ that separates its interior from v.
- Pick a coordinate system with e_1 horizontal and v above it.
- Let l be a vertex-free line segment from v to a point on e_1.
- Let $e_1, \ldots, e_m$ be the edges that intersect l in vertical order.
- Let w_i be the vertex above e_i that forms a triangle τ_i with it.

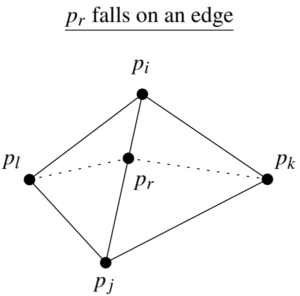
Legal Implies Delaunay (continued)



- Observe that $v = w_m$.
- w_1 is not in the circumcircle of τ by legality.
- The circumcircle of τ_1 contains the portion of the circumcircle of τ above e_1, so it contains v (right figure). **Why?**
- The circumcircle of τ_m contains v by induction.
- Contradiction: $v = w_m$ is a vertex of τ_m.

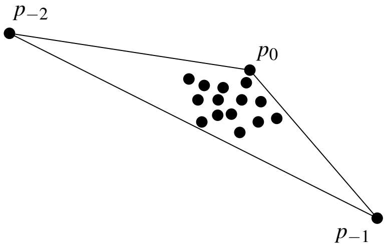
Delaunay Triangulation Algorithm





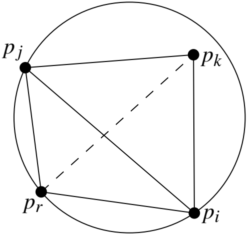
1. Construct $p_{-2}p_{-1}p_0$ and a point location graph.
2. Insert $p_1, \ldots, p_{n-1}$ in random order.
   - 2.1 Use the graph to find the triangle $p_i p_j p_k$ that contains p_r.
   - 2.2 Split it into $p_r p_i p_j$, $p_r p_j p_k$, and $p_r p_k p_i$.
   - 2.3 Call $\text{legalize}(p_r, p_i p_j)$, $\text{legalize}(p_r, p_j p_k)$, and $\text{legalize}(p_r, p_k p_i)$.
   - 2.4 Update the graph.
3. Remove the triangles incident on p_{-2} and p_{-1}.

Bounding Triangle



- The input points are $p_0, \ldots, p_{n-1}$ with p_0 the highest.
- Dummy point p_{-1} is below and to the right of the input.
- Dummy point p_{-2} is above and to the left of the input.
- The points $p_1, \ldots, p_{n-1}$ are in the triangle $p_{-2}p_{-1}p_0$.
- p_{-2} and p_{-1} are outside the circumcircles of the input points.
- Predicates involving p_{-2} and p_{-1} are computed symbolically.

Removing Illegal Edges



$\mathrm{legalize}(p_r, p_i p_j)$

If $p_i p_j$ is illegal:

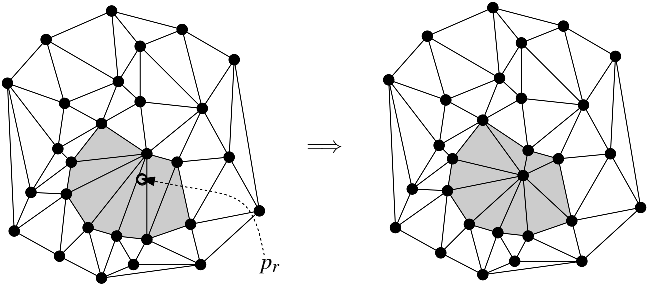
Let $p_j p_i p_k$ be opposite $p_i p_j p_r$.

Flip $p_i p_j$ to $p_r p_k$.

$\mathrm{legalize}(p_r, p_i p_k)$

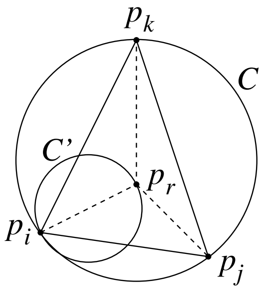
$\mathrm{legalize}(p_r, p_k p_j)$.

Correctness



- **Lemma 9.10** The new edges are Delaunay.
- Legalize terminates because flips increase the angle sequence.
- Conclusion: legalize restores the Delaunay property.

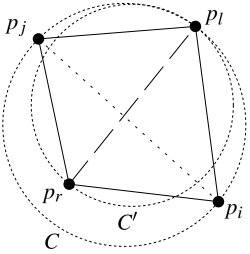
Proof of Lemma 9.10



Initial new edges are Delaunay.

- The circumcircle C of $p_i p_j p_k$ is empty before p_r is inserted.
- Let $C' \subset C$ be the circle with diameter $p_r p_i$.
- This circle is empty, so $p_r p_i$ is Delaunay.
- Likewise $p_r p_j$ and $p_r p_k$.

Proof of Lemma 9.10 (continued)



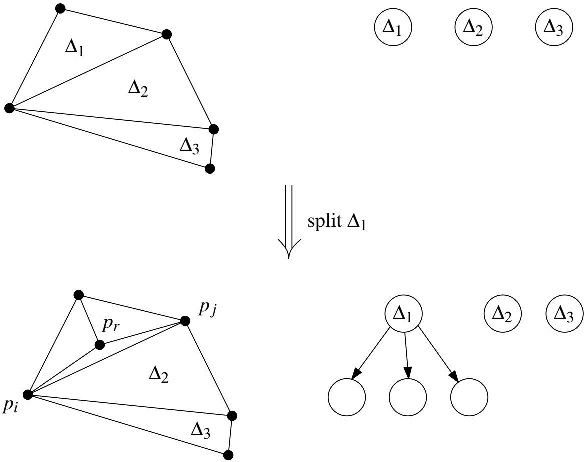
Legalize creates Delaunay edges.

- Let legalize flip the edge $p_i p_j$ to $p_r p_l$.
- The circumcircle C of $p_i p_j p_l$ contains p_r and no other point.
- Let $C' \subset C$ be the circle with diameter $p_r p_l$.
- This circle is empty, so $p_r p_l$ is Delaunay.

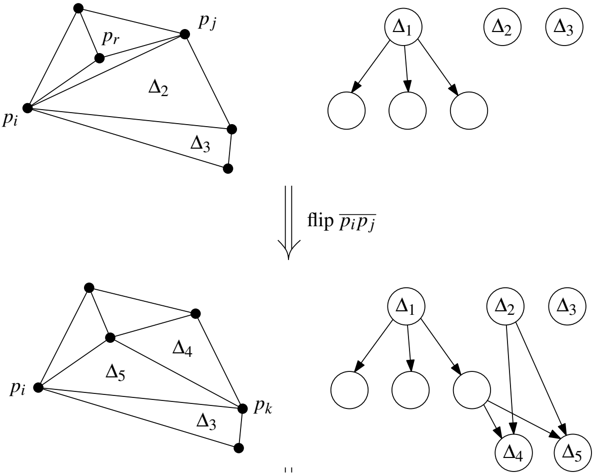
Search Graph

- Nodes contain triangles.
- The leaf nodes comprise the current triangulation.
- The internal nodes are from prior triangulations.
- An internal node has two or three children.
- Its triangle is a subset of the union of their triangles.
- Triangle splits and edge flips create internal nodes.

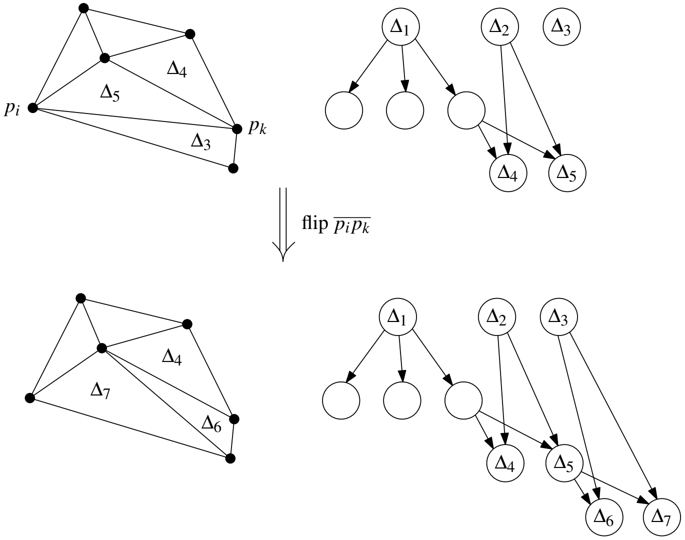
Search Graph Update 1



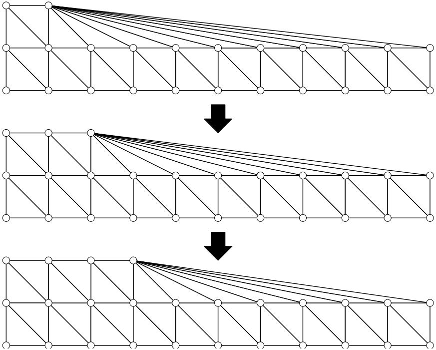
Search Graph Update 2



Search Graph Update 3

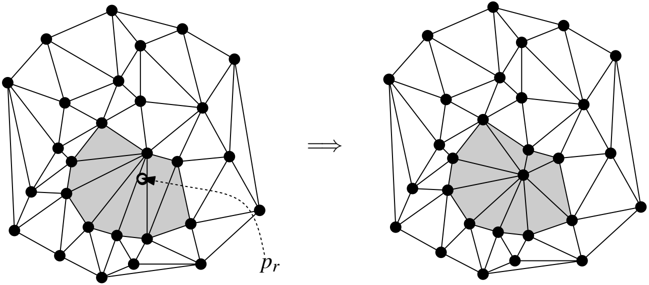


Time Complexity



- The worst case time complexity for n points is $O(n^2)$.
- The expected complexity is $O(n \log n)$.
- We will prove this for a variant of the textbook algorithm.
- The variant is simpler, faster, and easier to analyze.
- The textbook gives a proof for its algorithm in Section 9.4.

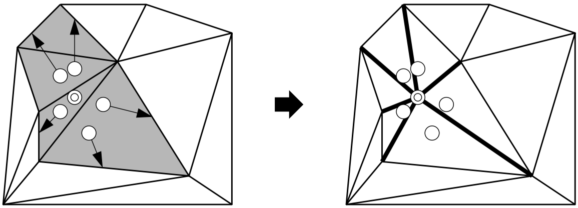
Remeshing



Alternative to splitting triangles and flipping edges

- If a point is in the circumcircle of a triangle, they *conflict*.
- The triangles that conflict with p_r form a star-shaped region.
- Locate one triangle then find the rest by graph traversal.
- Remove them all.
- Connect p_r to each edge of the star-shaped region.

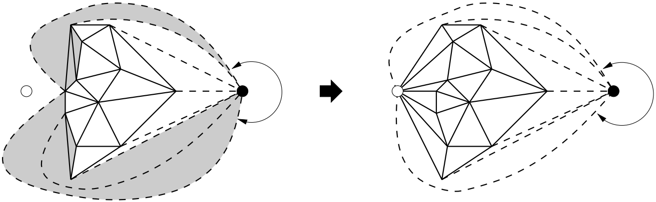
Conflict Sets



Alternative to search graph for point location

- Each uninserted point stores the triangle that contains it.
- Each triangle stores the points that it contains.
- The initial triangle contains all the uninserted points.
- This data is updated when each point p_r is inserted.
  1. Let the triangle t contain the point q.
  2. Find the edge ab of t where the ray $p_r q$ exits.
  3. Let s be the triangle incident on ba.
  4. If s conflicts with p_r, replace t by s and go to step 2.
  5. Assign q to the triangle $p_r ab$ and vice versa.

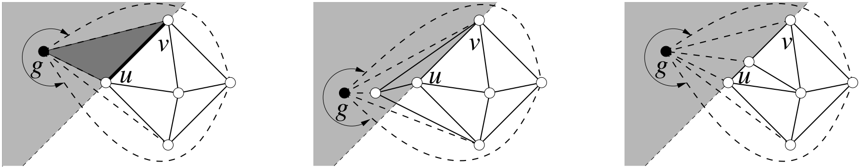
Ghost Triangles



Alternative to a bounding triangle with symbolic predicates

- Define a *ghost vertex* g at infinity (black circle).
- Each convex hull edge ab generates a ghost triangle bag.
- Remeshing works with ghost triangles.
- Example: 3 ghost triangles and 3 triangles (shaded) are replaced by 2 ghost triangles and 6 triangles.

Ghost Triangle Conflict Sets



A ghost triangle *uvg* conflicts with a vertex *a* in two cases.

1. *a* is in the *uv* half space (middle).
2. *a* is on *uv* (right).

The conflict set update for *q* starts in a real triangle *t* and ends in a real triangle or the first time it crosses into a ghost triangle.

Ghost edges neither have nor need equations.

Alternate Delaunay Triangulation Algorithm

1. Randomly permute the sites to obtain $p_1, \ldots, p_n$.
2. Construct $p_1 p_2 p_3$, ghost vertex g, ghost triangles $p_2 p_1 g$, $p_3 p_2 g$, and $p_1 p_3 g$, and initialize the conflict sets.
3. Insert $p_4, \ldots, p_n$.
    3.1 Find the conflict set of p_r by graph traversal.
    3.2 Remove the conflict set and remesh the resulting cavity.
    3.3 Create the conflict sets of the cavity points and triangles.
4. Remove the ghost triangles.

Expected Time Complexity

Let T_i denote the Delaunay triangulation of $p_1, \ldots, p_i$.

1. T_i contains $2i - 2$ triangles and $3i - 3$ edges.
Proof There are $2i - k - 2$ regular triangles and $3i - k - 3$ regular edges by Theorem 9.1 with k the number of hull edges. There are k ghost edges and k ghost triangles.

2. Inserting p_i creates fewer than 6 triangles on average.
Proof The number of triangles equals the degree of p_i in T_i. There are $6i - 6$ edge endpoints by 1, so the average degree of a non-ghost vertex is bounded by $(6i - 6)/i < 6$.

3. The time excluding conflict set updates is $O(n)$.
Proof The time is linear in the number of triangles created and deleted. The former is bounded by $6n$ and the latter is smaller.

4. p_i conflicts with fewer than 4 triangles on average.
Proof The conflicting triangles are deleted when p_i is inserted. The number deleted is two less than the number inserted because there are $2i - 2$ triangles in T_i by 1.

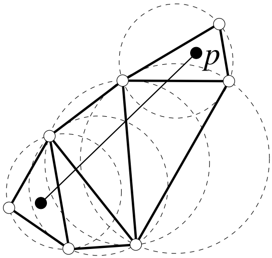
Expected Time Complexity (continued)

5. Inserting p_i creates less than $12(n-i)/i$ conflicts on average.
Proof We bound the number of conflicts that disappear when p_i is removed from T_i (backward analysis). A triangle disappears when p_i is one of its vertices, which occurs with probability $3/i$ for a regular triangle and $2/i$ for a ghost triangle. A conflict disappears when its triangle disappears, which has probability at most $3/i$. The expected number of conflicts in T_i equals the sum of the expected number over the $n-i$ uninserted points, which is less than $4(n-i)$ by 4. The expected number of conflicts that disappear is less than $4(n-i)(3/i) = 12(n-i)/i$.

6. The expected conflict set update time is $O(n \log n)$.
Proof Each step along the ray $p_r q$ traverses a triangle that conflicts with p_r. Thus the time is bounded by the number of conflicts that are created, which is

$$12 \sum_{i=1}^{n} \frac{n-i}{i} = 12n \sum_{i=1}^{n} \frac{1}{i} - 12n = O(n \log n).$$

Walking Algorithm



Alternative to search graph or conflict sets for point location

- Insert points based on locality (quadtree order).
- Locate the triangle that contains the point p.
  1. Let t be the last triangle created and q be a point in t.
  2. If t contains p, return t.
  3. Replace t by the adjacent triangle that qp intersects.
  4. Go to step 2.
- No theoretical bound, but good in practice.

Simple Data Structures

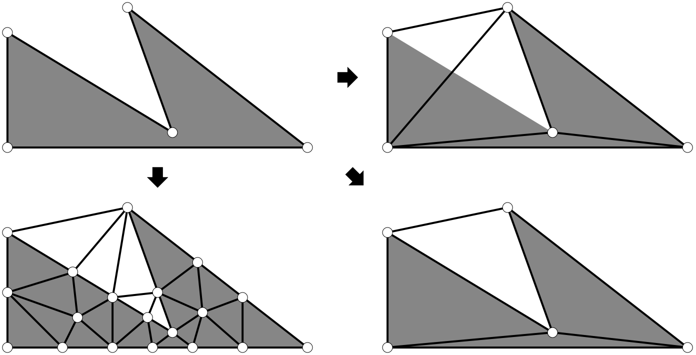
Alternative to subdivision representation of the triangulation

1. Point: (x, y) coordinates.
2. Designated ghost point.
3. Edge: two ordered points.
4. Triangle: three ordered points.
5. Map from edge to incident triangle.

Practical Delaunay Triangulation Algorithm

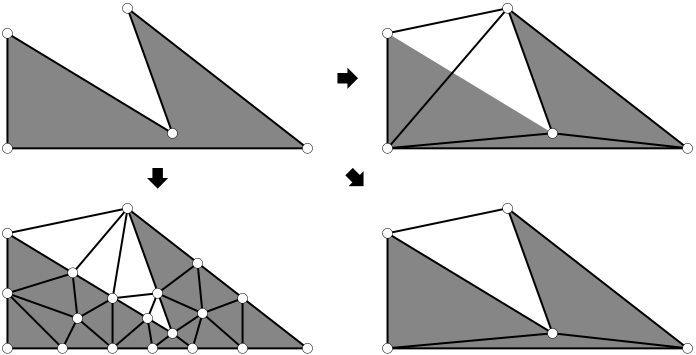
1. Construct $p_1 p_2 p_3$ and three ghost triangles.
2. Insert $p_4, \ldots, p_n$ in quadtree order.
   2.1 Find the triangle s that contains p_r by walking.
   2.2 Find the conflict set of p_r by graph traversal.
   2.3 Remove the conflict set and remesh the resulting cavity.
3. Remove the ghost triangles.

Steiner Delaunay Triangulation



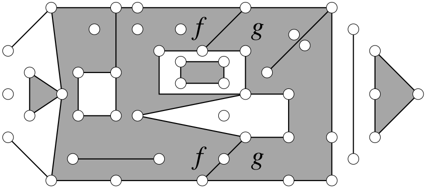
- Delaunay triangulation extends to polygons.
- Triangulating the vertices fails when edges are not Delaunay.
- This problem can be solved by subdividing the edges.
- The triangulation can be improved by adding interior points.
- A Delaunay triangulation with extra points is called *Steiner*.

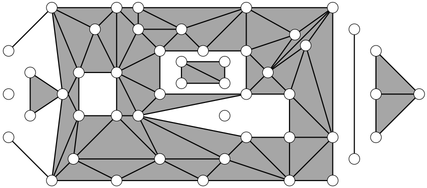
Constrained Delaunay Triangulation



- A *constrained Delaunay triangulation* forces the polygon edges to be triangulation edges.
- The other edges are legal.
- The algorithm is quite complicated.

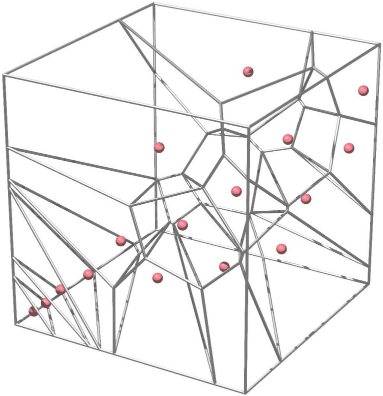
Constrained Delaunay Triangulation (continued)





The input can be any set of vertices, edges, and triangles.

3D Voronoi Diagram



- The Voronoi diagram of 3D sites is a spatial subdivision.
- The cells are convex and can be unbounded.
- The facet between two cells lies on the bisector of their sites.
- Three facets meet at an edge where three sites are equidistant.
- Four edges meet at a vertex where four sites are equidistant.

3D Voronoi Diagram Construction

- The space complexity for n sites is n^2.
- There are complicated optimal n^2 algorithms.
- We will see a simple Delaunay triangulation algorithm.
- The Voronoi diagram is easily obtained as the dual.

3D Delaunay Triangulation

- The Delaunay triangulation of 3D points is a decomposition of their convex hull into tetrahedra with empty circumspheres.
- It is the dual of the Voronoi diagram: vertices map to cells, edges to facets, triangles to edges, and tetrahedra to vertices.
- The practical algorithm easily transfers to 3D.
- A ghost tetrahedron has one real and three ghost triangles.
- The cavity is a star-shaped cell.
- The simple data structures generalize easily.
- The expected time complexity with conflicts sets is $O(n^2)$.
- The walking algorithm crosses triangles and remains superior.