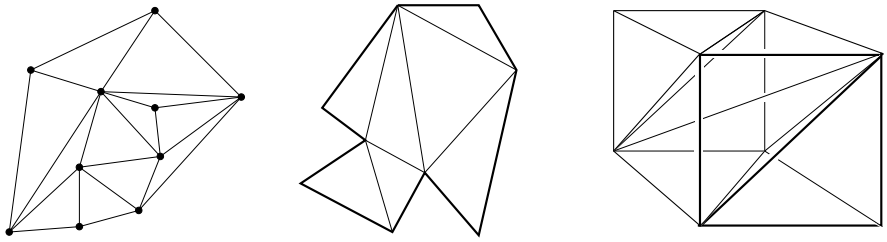
29 TRIANGULATIONS AND MESH GENERATION

Marshall Bern, Jonathan R. Shewchuk, and Nina Amenta

INTRODUCTION

A triangulation is a partition of a geometric domain, such as a point set, polygon, or polyhedron, into simplices that meet only at shared faces. (For a point set, the triangulation covers the convex hull.) Triangulations are important for representing complicated geometry by piecewise simple geometry and for interpolating numerical fields. The first four sections of this chapter discuss two-dimensional triangulations: Delaunay triangulation of point sets (Section 29.1); triangulations of polygons, including constrained Delaunay triangulations (Section 29.2); other optimal triangulations (Section 29.3); and mesh generation (Section 29.4). The last three sections treat triangulations of surfaces embedded in $\mathbb{R}^3$ (Section 29.5), triangulations (composed of tetrahedra) of polyhedra in $\mathbb{R}^3$ (Section 29.6), and triangulations in arbitrary dimension $\mathbb{R}^d$ (Section 29.7).

FIGURE 29.0.1
Triangulations of a point set, a simple polygon, and a polyhedron.



29.1 DELAUNAY TRIANGULATION

The Delaunay triangulation is the most famous and useful triangulation of a point set. Chapter 27 discusses this construction in conjunction with the Voronoi diagram.

GLOSSARY

Empty circle: No input points strictly inside the circle.

Delaunay triangulation (DT): All triangles have empty circumcircles.

Completion: Adding edges to a polyhedral subdivision to make a triangulation.

Edge flipping: Replacing an edge by a crossing edge; used to compute a DT.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

BASIC FACTS

Let $S = \{s_1, s_2, \ldots, s_n\}$ (for "sites") be a set of points in the Euclidean plane $\mathbb{R}^2$. The Delaunay triangulation (DT) is a triangulation of S defined by the ***empty circle condition***: a triangle $s_i s_j s_k$ appears in the DT only if its circumscribing circle (***circumcircle***) has no point of S strictly inside it.

The ***Delaunay subdivision*** is a subdivision of the convex hull of S into polygons with cocircular vertices and empty circumcircles. The Delaunay subdivision is the planar dual of the Voronoi diagram, meaning that an edge $s_i s_j$ appears in the subdivision if and only if the Voronoi cells of s_i and s_j share a boundary edge. If no four points in S are cocircular, the Delaunay subdivision is a triangulation of S. If four or more points in S lie on a common empty circle, the Delaunay subdivision has one or more faces with more than three sides. These can be triangulated to ***complete*** a Delaunay triangulation of S. The triangulations that complete these faces can be chosen arbitrarily, so the DT is not always unique.

There is a connection between a Delaunay subdivision in $\mathbb{R}^2$ and a convex polytope in $\mathbb{R}^3$. If we ***lift*** S onto the paraboloid with equation $z = x^2 + y^2$ by mapping $s_i = (x_i, y_i)$ to $(x_i, y_i, x_i^2 + y_i^2)$, then the Delaunay subdivision turns out to be the projection of the lower convex hull of the lifted points. See Figure 27.1.2.

ALGORITHMS

There are a number of practical planar DT algorithms [For95], including edge flipping, incremental construction, plane sweep, and divide and conquer. The last three algorithms can be implemented to run in $O(n \log n)$ time. We describe only the edge flipping algorithm, even though its worst-case running time of $O(n^2)$ is not optimal, because it is most relevant to our subsequent discussion.

The edge flipping algorithm starts from any triangulation of S and then locally optimizes each edge. Let e be an internal edge (not on the boundary of the convex hull of S) and Q_e be the triangulated quadrilateral formed by the two triangles sharing e. Q_e is *reversed* if the two angles opposite the diagonal sum to more than 180°, or equivalently, if each triangle's circumcircle encloses the opposite vertex. If Q_e is reversed, we "flip" it by exchanging e for the other diagonal of Q_e.

```
Compute an initial triangulation of S
Place all internal edges into a queue
while the queue is not empty do
    Remove an edge e from the queue
    if quadrilateral Q_e is reversed then
        Flip e; add the four outside edges of Q_e to the queue fi od
```

An initial triangulation can be computed by a plane-sweep algorithm that adds the points of S by x-coordinate order (breaking ties by y-coordinate), as shown in Figure 29.1.1. Upon each addition, the algorithm walks around the convex hull of the already-added points, connecting the new vertex to the vertices it can see.

The following theorem guarantees the success of edge flipping: a triangulation in which no quadrilateral is reversed must be a DT. This theorem can be proved with the lifting map: a reversed quadrilateral lifts to a reflex edge, and a surface without reflex edges must be the lower convex hull.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

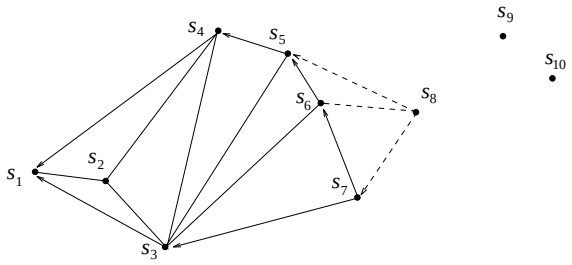


FIGURE 29.1.1
A generic step in computing an initial triangulation.

OPTIMALITY PROPERTIES

Certain measures of quality are improved by flipping a reversed quadrilateral [BE95]. For example, the minimum angle in a triangle of Q_e must increase. Hence, a triangulation that maximizes the minimum angle cannot have a reversed quadrilateral, implying that it is a DT. Among all triangulations of the input points, some DT:

- maximizes the minimum angle (moreover, lexicographically maximizes the list of angles ordered from smallest to largest);
- minimizes the maximum radius of the triangles' circumcircles;
- minimizes the maximum radius of the triangles' smallest enclosing circles;
- maximizes the sum of the radii of the triangles' inscribed circles;
- minimizes the "potential energy" (sum of area-weighted squared gradients) of an interpolated piecewise-linear surface; and
- minimizes the surface area of a piecewise-linear surface for elevations scaled sufficiently small.

Two additional properties of the DT: Delaunay triangles are acyclically ordered by distance from any fixed reference point. The distance between any pair of vertices, measured along edges of the DT, is at most a constant (less than 1.998) times the Euclidean distance between them [Xia13].

WEIGHTED DELAUNAY TRIANGULATIONS

Voronoi diagrams can be defined for various distance measures (Section 27.3), and some of them induce Delaunay-like triangulations by duality. Here we mention one generalization that retains most of the rich mathematical structure. Assign each point $s_i = (x_i, y_i)$ in S a real weight w_i. The ***weighted Delaunay triangulation*** of S is the projection of the lower convex hull of the points $(x_i, y_i, x_i^2 + y_i^2 - w_i)$. With a small (perhaps negative) weight, a site can fail to appear in the weighted Delaunay triangulation, because the corresponding point in $\mathbb{R}^3$ lies inside the convex hull. Hence, in general the weighted Delaunay triangulation is a graph on a subset of the sites S. In the special case that the weights are all zero (or all equal), the weighted Delaunay triangulation is the DT.

A ***regular triangulation*** is a triangulation in $\mathbb{R}^2$ found as a projection of the lower surface of a polytope in $\mathbb{R}^3$. Not all triangulations are regular; see Section 16.3 for a counterexample. Every regular triangulation can be expressed as a weighted Delaunay triangulation, because the w_i weights are arbitrary. (The problem of finding suitable weights can be expressed as a linear program.)

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

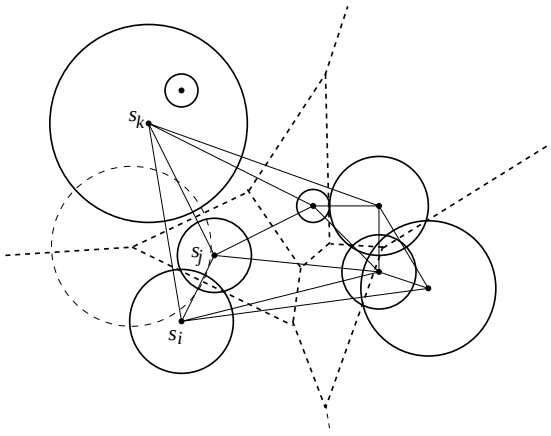


FIGURE 29.1.2
Power diagram (dashed) and weighted Delaunay triangulation. The dashed circle is the orthogonal circle for triangle $s_i s_j s_k$.

The planar dual of the weighted Delaunay triangulation is the ***power diagram***, a Voronoi diagram in which the distance from a site $s_i \in S$ to a point in $\mathbb{R}^2$ is the square of the Euclidean distance minus w_i. We can regard the sites in a power diagram as circles, with the radius of site i being $\sqrt{w_i}$. See Figure 29.1.2. In weighted Delaunay triangulations, the analogue of the empty circle condition is the ***orthogonal circle condition***: a triangle $s_i s_j s_k$ appears in the triangulation only if the circle that crosses circles i, j, and k at right angles penetrates no other site's circle more deeply.

29.2 TRIANGULATIONS OF POLYGONS

We now discuss triangulations of more complicated inputs: polygons and planar straight-line graphs. We start with the problem of computing any triangulation at all; then we progress to constrained Delaunay triangulations.

GLOSSARY

Simple polygon: Boundary is a loop made of edges without self-intersections.

Monotone polygon: Intersection with any vertical line is one segment.

Constrained Delaunay triangulation: Allows input edges as well as vertices. Triangles have empty circumcircles, meaning no *visible* input vertices.

TRIANGULATIONS OF SIMPLE POLYGONS

Triangulating a simple polygon is both an interesting problem in its own right and an important preprocessing step in other computations. For example, the following problems are known to be solvable in linear time once the input polygon P is triangulated: computing link distances from a given source, finding a monotone path within P connecting two given points, and computing the portion of P illuminated by a given line segment.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

How much time does it take to triangulate a simple polygon? For practical purposes, one should use either an $O(n \log n)$ deterministic algorithm (such as the one given below for the more general case of planar straight-line graphs) or a slightly faster randomized algorithm (such as one with running time $O(n \log^* n)$ described by Mulmuley [Mul94]).

However, for theoretical purposes, achieving the ultimate running time was for several years an outstanding open problem. After a sequence of interim results, Chazelle [Cha91] devised a linear-time algorithm. Chazelle's algorithm, like previous algorithms, reduces the problem to that of computing the ***horizontal visibility map*** of P—the partition obtained by shooting horizontal rays left and right from each of the vertices. The "up-phase" of this algorithm recursively merges coarse visibility maps for halves of the polygon (polygonal chains); the "down-phase" refines the coarse map into the complete horizontal visibility map.

TRIANGULATIONS OF PLANAR STRAIGHT-LINE GRAPHS

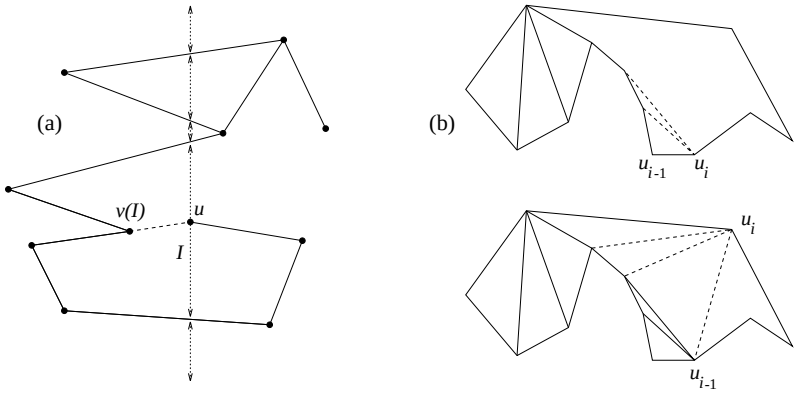
Let G be a planar straight-line graph (PSLG). We describe an $O(n \log n)$ algorithm [PS85] that triangulates G in two stages, called regularization and triangulation. Regularization adds edges to G so that each vertex, except the first and last, has at least one edge extending to the left and one extending to the right. Conceptually, we sweep a vertical line ℓ from left to right across G while maintaining the list of intervals of ℓ between successive edges of G. For each vertical interval I in ℓ, we remember a vertex $v(I)$ visible to all points of I: this vertex is either an endpoint of one of the two edges bounding I or a vertex between these edges, lacking a right edge. When we hit a vertex u with no left edge, we add the edge $\{u, v(I)\}$, where I is the interval containing u, as shown in Figure 29.2.1(a). After the left-to-right sweep, we sweep from right to left, adding right edges to vertices lacking them.

Start vertical line ℓ to the left of all vertices in G
for each vertex u of G from left to right **do**
  if u has no left edges **and** u isn't the first vertex **then**
    Add edge $\{u, v(I)\}$ where I is the interval containing u **fi**
  Delete u's left edges from interval list
  Insert u's right edges with $v(I) \leftarrow u$ for each new vertical interval I **od**
  if u has no right edges **then**
    Set $v(I) \leftarrow u$ for the interval I that contains u **fi**
Repeat the steps above for vertices from right to left

After the regularization stage, each bounded face of G is a ***monotone polygon***, meaning that every vertical line intersects the face in at most one interval. We consider the vertices $u_1, u_2, \ldots, u_n$ of a face in left-to-right order, using a stack to store the not-yet-triangulated vertices (a reflex chain) to the left of the current vertex u_i. If u_i is adjacent to u_{i-1}, the topmost vertex on the stack, as shown in the upper illustration of Figure 29.2.1(b), then we pop vertices off the stack and add diagonals from these vertices to u_i, until the vertices on the stack—u_i on top—again form a reflex chain. If u_i is instead adjacent to the leftmost vertex on the stack, as shown in the lower picture, then we can add a diagonal from each vertex on the stack, and clear the stack of all vertices except u_i and u_{i-1}.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

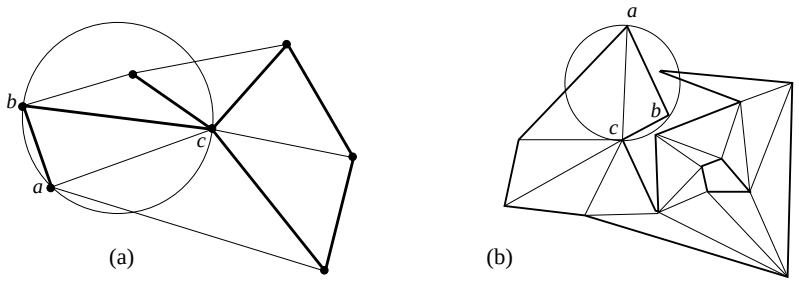
FIGURE 29.2.1
(a) *Sweep-line algorithm for regularization.* (b) *Stack-based triangulation algorithm.*



CONSTRAINED DELAUNAY TRIANGULATIONS

The constrained Delaunay triangulation [LL86] provides a way to force the edges of a planar straight-line graph G into the DT. A point p is ***visible*** to point q if line segment pq does not intersect any edge or vertex in G, except maybe at its endpoints. A triangle abc with vertices from G appears in the ***constrained Delaunay triangulation*** (CDT) if its circumcircle encloses no vertex of G visible to some point in the interior of abc, and moreover, no edge of G intersects the interior of abc. If G is a graph with vertices but not edges, then the CDT is the ordinary, unconstrained Delaunay triangulation. If G is a polygon or polygon with holes, as in Figure 29.2.2(b), then the CDT retains only the triangles interior to G.

FIGURE 29.2.2
Constrained Delaunay triangulations of (a) *a PSLG and* (b) *a polygon with a hole.*



The edge flipping algorithm works for CDTs, with the modification that edges of G are never placed on the queue or flipped. There are also $O(n \log n)$-time algorithms for the CDT [Sei88, Che89], and even $O(n)$-time algorithms for the case that G is just a simple polygon [KL93, CW98]. A slightly slower incremental edge insertion algorithm is usually used in practice [SB15]. See Section 67.2 for pointers to software for computing the constrained Delaunay triangulation.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

29.3 OPTIMAL TRIANGULATIONS

We have already seen two types of optimal triangulations: the DT and the CDT. Some applications, however, demand triangulations with properties other than those optimized by DTs and CDTs. Table 29.3.1 summarizes some results.

GLOSSARY

Edge insertion: Local improvement operation, more general than edge flipping.

Local optimum: A solution that cannot be improved by local moves.

Greedy triangulation: Repeatedly add the shortest non-crossing edge.

Steiner triangulation: Extra vertices, not in the input, are allowed.

TABLE 29.3.1 Optimal triangulation results. (Constrained) Delaunay triangulations maximize the minimum angle and optimize many other objectives.

PROPERTY	INPUTS	ALGORITHMS	TIME
various properties	polygons	dynamic programming [Kli80]	$O(n^3)$
constrained Delaunay	polygons	divide and conquer [KL93, CW98]	$O(n)$
minimize total edge length	polygons	approximation algorithms [Epp94, LK98]	$O(n \log n)$
Delaunay	point sets	various algorithms [For95]	$O(n \log n)$
minimize maximum angle	point sets	fast edge insertion [ETW92]	$O(n^2 \log n)$
minmax slope terrain	point sets	edge insertion [BEE$^+$93]	$O(n^3)$
minmax edge length	point sets	MST induces polygons [ET91]	$O(n^2)$
greedy edge length	point sets	dynamic Voronoi diagram [LL92]	$O(n^2)$
constrained Delaunay	PSLGs	various algorithms [Sei88, Che89, SB15]	$O(n \log n)$

OPTIMAL TRIANGULATIONS OF SIMPLE POLYGONS

Many problems in finding an optimal triangulation of a simple polygon can be solved in $O(n^3)$ time by a dynamic programming algorithm of Klincsek [Kli80]. For example, the algorithm solves any problem that assigns a weight to each possible triangle and/or edge and asks to find the triangulation that minimizes or maximizes the minimum or maximum weight or the sum of the weights. Examples include the minimum/maximum angle in the triangulation, the minimum/maximum length of an edge in the triangulation, and the sum of edge lengths in the triangulation. The triangulation that minimizes the sum of edge lengths is called the *minimum weight triangulation* and is discussed further below.

The running time can be improved for some nonconvex polygons. Let p be the number of pairs of vertices of the polygon that can "see" each other; that is, the line segment connecting them is in the polygon. The optimal triangulation can be found in $O(n^2 + p^{3/2})$ time by first computing a *visibility graph* [BE95].

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

EDGE FLIPPING AND EDGE INSERTION

The edge flipping DT algorithm can be modified to compute other locally optimal triangulations of point sets. For example, if we redefine "reversed" to mean a quadrilateral triangulated with the diagonal that forms the larger maximum angle, then edge flipping can be used to minimize the maximum angle. For the minmax angle criterion, however, edge flipping computes only a local optimum, not necessarily the true global optimum.

Although edge flipping seems to work well in practice [ETW92], its theoretical guarantees are very weak: the running time is not known to be polynomially bounded and the local optimum it finds may be greatly inferior to the true optimum.

A more general local improvement method, called ***edge insertion*** [BEE+93, ETW92] exactly solves certain minmax optimization problems, including minmax angle and minmax slope of a piecewise-linear interpolating surface.

Assume that the input is a planar straight-line graph G, and we are trying to minimize the maximum angle. Starting from some initial triangulation of G, edge insertion repeatedly adds a candidate edge e that subdivides the maximum angle. (In general, edge insertion always breaks up a worst triangle by adding an edge incident to its "worst vertex.") The algorithm then removes the edges that are crossed by e, forming two polygonal holes alongside e. Holes are retriangulated by repeatedly removing ***ears*** (triangles with two sides on the boundary, as shown in Figure 29.3.1) with maximum angle smaller than the old worst angle $\angle cab$. If retriangulation succeeds, then the overall triangulation improves and edge bc is eliminated as a future candidate. If retriangulation fails, then the overall triangulation is returned to its state before the insertion of e, and e is eliminated as a future candidate. Each candidate insertion takes time $O(n)$, giving a total running time of $O(n^3)$.

Compute an initial triangulation with all $\binom{n}{2}$ edge slots unmarked
while $\exists$ an unmarked edge e cutting the worst vertex of worst triangle abc **do**
    Add e and remove all edges crossed by e
    Try to retriangulate the polygonal holes by removing ears better than abc
    if retriangulation succeeds **then**
        mark bc
    else mark e and undo e's insertion **fi od**

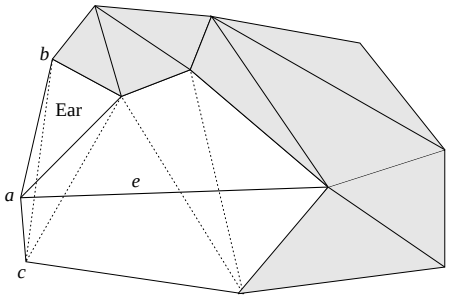


FIGURE 29.3.1
The maximum angle $\angle cab$ is subdivided by the insertion of an edge e, causing the deletion of the dashed edges. The algorithm retriangulates the two polygonal holes alongside e by removing sufficiently good ears. (From [BE95]*, with permission.)*

Edge insertion can compute the minmax "eccentricity" triangulation or the minmax slope surface [BEE+93] in time $O(n^3)$. By inserting candidate edges in a certain order, one can improve the running time to $O(n^2 \log n)$ for minmax angle [ETW92] and maxmin triangle height.

MINIMUM WEIGHT TRIANGULATIONS

Several natural optimization criteria can be defined using edge lengths [BE95]. The most famous such criterion—called ***minimum weight triangulation***—asks for a triangulation of a planar point set minimizing the total edge length. We have already seen that for simple polygons it can be computed in $O(n^3)$ time, but the dynamic programming algorithm does not apply to point sets. Since the problem was posed in 1970, the suspicion that it is NP-hard grew but was only confirmed in 2006, by Mulzer and Rote [MR08]. (It is still not known whether the problem is in NP, because of technicalities related to computer arithmetic with radicals: it is not known how to compare sums of Euclidean lengths in polynomial time.)

However, there is an algorithm that is quite fast in practice for most point sets [DKM97]. This algorithm uses a local criterion to find edges sure to be in the minimum weight triangulation. These edges break the convex hull of the point set into regions, such as simple polygons or polygons with one or two disconnected interior points, that can be triangulated optimally using dynamic programming.

The best polynomial-time approximation algorithm for minimum weight triangulation, by Levcopoulos and Krznaric [LK98], gives a solution within a constant multiplicative factor of the optimal length. Remy and Steger [RS09] give an approximation scheme that finds a $(1+\epsilon)$-approximation for any fixed $\epsilon > 0$, but runs in quasi-polynomial $n^{O(\log^8 n)}$ time. Eppstein [Epp94] gives a constant-factor approximation ratio for minimum weight Steiner triangulation, in which extra vertices are allowed.

A commonly used heuristic for minimum weight triangulation is ***greedy triangulation***. This algorithm adds edges one at a time, each time choosing the shortest edge that is not already crossed. The greedy triangulation can be viewed as an optimal triangulation in its own right, because it lexicographically minimizes the sorted vector of edge lengths. For planar point sets, the greedy triangulation approximates the minimum weight triangulation by a factor of $O(\sqrt{n})$, and it can be computed in time $O(n^2)$ by dynamic maintenance of a bounded Voronoi diagram [LL92].

Another natural criterion asks for a triangulation minimizing the maximum edge length. Edelsbrunner and Tan [ET91] show that such a triangulation—like the DT—must contain the edges of the minimum spanning tree (MST). The MST, together with the edges of the convex hull, divides the polygon into regions that are weakly-simple polygons (each interior is a topological disk, but an edge might appear multiple times on the boundary). This geometric lemma gives the following $O(n^2)$-time algorithm: compute the MST, then triangulate the resulting weakly-simple polygons optimally using dynamic programming.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

OPEN PROBLEMS

1. Explain the empirical success and limits of edge flipping for non-Delaunay optimization criteria—both solution quality and running time.
2. Can the minimum weight triangulation of a convex polygon be computed in $o(n^3)$ time?
3. Find a polynomial-time approximation scheme for the minimum weight triangulation.
4. Show that the minimum weight Steiner triangulation exists; that is, rule out the possibility that more and more Steiner points decrease the total edge length forever.

29.4 PLANAR MESH GENERATION

A ***mesh*** is a decomposition of a geometric domain into ***elements***, usually triangles or quadrilaterals in $\mathbb{R}^2$. (For brevity, we ignore a large literature on quadrilateral mesh generation.) Meshes are used to discretize functions, especially solutions to partial differential equations. Piecewise linear discretizations are by far the most popular. Practical mesh generation problems tend to be application-specific: one desires small elements where a function changes rapidly and larger elements elsewhere. However, certain goals apply fairly generally, and computational geometers have formulated problems incorporating these considerations. Table 29.4.1 summarizes these results; below we discuss some of them in detail.

GLOSSARY

Steiner point: An added vertex that is not an input point.

Conforming mesh: Elements exactly cover the input domain.

Quadtree: A recursive subdivision of a bounding square into smaller squares.

TABLE 29.4.1 Some mesh generation results.

PROPERTY	INPUTS	ALGORITHMS	SIZE
no small or obtuse angles	polygons	grid [BGR88], quadtree [BEG94]	$O(1) \cdot$ optimal
no obtuse angles	polygons	disk packing [BMR94]	$O(n)$
no small angles	most PSLGs	Delaunay [Rup95]	$O(1) \cdot$ optimal
no obtuse angles	PSLGs	disk packing & propagation [Bis16]	$O(n^{2.5})$
no large angles	PSLGs	propagating horns [Mit93, Tan94, Bis16]	$O(n^2)$
no extreme dihedral angles	polyhedra	octree [MV92]	$O(1) \cdot$ optimal
no extreme dihedral angles	smooth 3D	octree [LS07]	no bound

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

NO SMALL ANGLES

Sharp triangles can degrade appearance and accuracy, so most mesh generation methods attempt to avoid small and large angles. (There are exceptions: properly aligned sharp triangles prove quite useful in simulations of fluid flow.) In finite element methods, elements with small angles sometimes cause the associated stiffness matrices to be badly conditioned, and elements with large angles can lead to large discretization errors.

Baker et al. [BGR88] give a grid-based algorithm for triangulating a polygon so the mesh has no obtuse angles and all the *new* angles—a sharp angle in the input cannot be erased—measure at least 14°. Bern et al. [BEG94] use quadtrees instead of a uniform grid and prove the following ***size optimality*** guarantee: the number of triangles is $O(1)$ times the minimum number in any no-small-angle triangulation of the input. The minimum number of triangles required depends not just on the number of input vertices n, but also on the geometry of the input. The simple example where the input is a long skinny rectangle shows why the number of output triangles depends upon the geometry.

Ruppert [Rup95], building on work of Chew, devised a ***Delaunay refinement*** algorithm with the same guarantee. The main loop of Ruppert's algorithm attempts to add the circumcenter (the center of the circumcircle) of a too-sharp triangle as a new vertex. If the circumcenter "encroaches" upon a boundary edge, meaning that it falls within the boundary edge's diameter circle, then the algorithm subdivides the boundary edge instead of adding the triangle circumcenter. Ruppert's algorithm accepts PSLGs as input, not merely polygons, but it is guaranteed to work only if the input PSLG has no acute angles. For PSLG inputs with small angles, unlike for polygon inputs, it is impossible to devise a mesh generation algorithm guaranteed to create no *new* small angles.

The size optimality guarantees for Delaunay refinement algorithms follow from a stronger guarantee: at each mesh vertex v, every adjoining mesh edge has length within a constant factor of the "local feature size" at v, which is a local measure of the distance between PSLG features (see also Chapter 35).

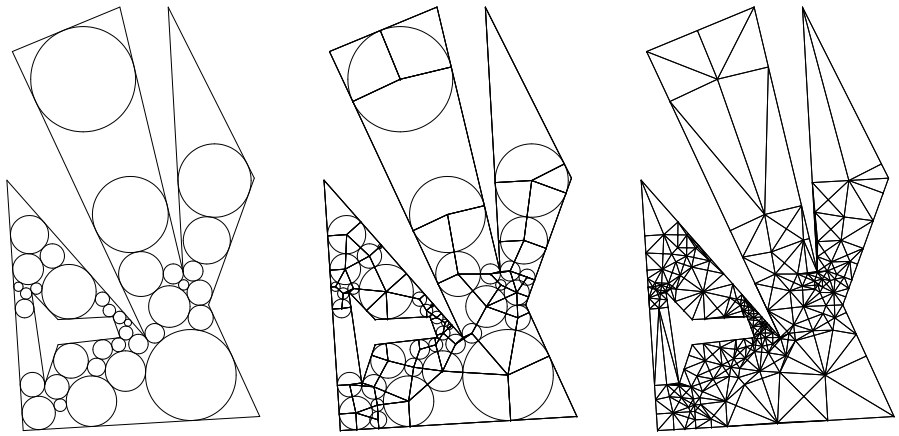
NO LARGE ANGLES

A lower bound on each triangle's smallest angle implies an upper bound on its largest angle. A weaker restriction is to prohibit large angles (close to 180°) only. The strictest bound on large angles that does not also imply a bound on small angles is to ask for no obtuse angles, that is, all angles are at most 90°. Surprisingly, it is possible to triangulate any polygon (possibly with holes) with only $O(n)$ nonobtuse triangles. Figure 29.4.1 illustrates an algorithm of Bern, Mitchell, and Ruppert [BMR94]: the domain is packed with nonoverlapping disks until each uncovered region has either 3 or 4 sides; radii to tangencies are added to split the domain into small polygons; and finally these polygons are triangulated with right triangles.

The problem is substantially harder for PSLGs than for polygons. There are PSLGs for which every nonobtuse triangulation has $\Omega(n^2)$ triangles. An algorithm of Bishop [Bis16] constructs a nonobtuse triangulation of a PSLG with $O(n^{2.5})$ triangles. Closing the gap between the $\Omega(n^2)$ lower bound and the $O(n^{2.5})$ upper

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

FIGURE 29.4.1
Nonobtuse triangulation steps. (From [BMR94], [BE95], *with permission.)*



bound is an open problem.

By relaxing the bound on the largest angle from 90° to something larger, one can improve the worst-case complexity of the mesh. Mitchell [Mit93] gives an algorithm that uses $O(n^2 \log n)$ triangles to guarantee that all angles measure less than 157.5°. The algorithm traces a cone of possible angle-breaking edges, called a *horn*, from each vertex (including Steiner points introduced on input edges) with a larger angle. Horns propagate around the PSLG until meeting an exterior edge or another horn. By adding some more horn-stopping "traps," Tan [Tan94] improves the angle bound to 132° and the complexity bound to $O(n^2)$. This complexity bound is tight; there are PSLGs for which a smaller complexity is not possible. Bishop [Bis16] combines the disk-packing algorithm of Bern et al. with propagation of Steiner points, yielding an angle bound of $90° + \epsilon$ and a complexity bound of $O(n^2/\epsilon^2)$ for any fixed $\epsilon > 0$.

CONFORMING DELAUNAY TRIANGULATIONS

For some applications, it suffices to have a mesh that is Delaunay and respects the input edges. A ***conforming Delaunay triangulation*** of a PSLG is a triangulation in which extra vertices—***Steiner points***—are added to the input, until the Delaunay triangulation of the vertices "conforms" to the input, meaning that each input edge is a union of Delaunay edges.

It is easy to verify that every nonobtuse triangulation is Delaunay. The nonobtuse triangulation algorithms discussed above currently offer the best size complexity of any known conforming Delaunay triangulation algorithms—namely, $O(n)$ for polygon inputs and $O(n^{2.5})$ for PSLG inputs. In principle, finding a conforming Delaunay triangulation of a PSLG might be easier than finding a nonobtuse triangulation, but the $\Omega(n^2)$ lower bound on triangulation complexity applies to conforming Delaunay triangulations as well (for worst-case PSLGs).

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

OPEN PROBLEMS

1. Does every PSLG have a conforming Delaunay triangulation of size $O(n^2)$? Or, more strongly, a nonobtuse triangulation of size $O(n^2)$?
2. Can the algorithms for triangulations with no large (or obtuse) angles be generalized to inputs with curved boundaries?

29.5 SURFACE MESHES

Sitting between two- and three-dimensional triangulations are triangulated surface meshes, which typically enclose 3D solids. Surface meshes are used heavily in computer graphics and in applications such as boundary element methods.

RESTRICTED DELAUNAY TRIANGULATIONS

Many ideas in guaranteed-quality mesh generation extend to surface meshes with the help of the ***restricted Delaunay triangulation*** (RDT) of Edelsbrunner and Shah [ES97], a geometric structure that extends the Delaunay triangulation to curved surfaces. The RDT is always a subcomplex of the three-dimensional Delaunay triangulation. It has proven itself as a mathematically powerful tool for surface meshing and surface reconstruction.

The RDT is defined by dualizing the restricted Voronoi diagram. The ***restricted Voronoi diagram*** of a point set $S \subset \Sigma$ with respect to a surface $\Sigma \subset \mathbb{R}^3$ is a cell complex much like the standard Voronoi diagram, but the restricted Voronoi cells contain only points on the surface Σ. The restricted Delaunay triangulation of S with respect to Σ is the subcomplex of S's 3D Delaunay triangulation containing every Delaunay face whose Voronoi dual face intersects Σ. Typically, the RDT is a subset only of the Delaunay triangles and their edges and vertices; the RDT contains no tetrahedra except in "degenerate" circumstances that can be prevented by infinitesimally perturbing Σ. Another way to characterize the RDT is by observing that its triangles have empty circumscribing spheres whose centers lie on Σ.

If Σ is a smooth surface and the point set S is sampled sufficiently densely from Σ, then the RDT is a triangulation of Σ. In particular, it can be shown that the RDT is homeomorphic to Σ and is a geometrically good approximation of Σ. (Note that if the point set S is not dense enough, the RDT may be a mess that fails to approximate Σ or even to be a manifold.) Partly for these reasons, RDTs have become a standard tool in provably good surface reconstruction and surface meshing algorithms [CDS12]. A surface meshing algorithm of Boissonnat and Oudot [BO05] offers guarantees similar to those of Ruppert's algorithm in the plane. The algorithm can be extended to generate tetrahedral meshes in a volume enclosed by a smooth surface [ORY05]. These algorithms have a simple interaction with the representation of Σ: they require an oracle that, given an arbitrary line segment, returns a point where the line segment intersects Σ. This oracle can be implemented efficiently in practice for many different surface representations.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

INTRINSIC DELAUNAY AND SELF-DELAUNAY MESHES

Another way to define a Delaunay-like surface triangulation is to dualize the Voronoi diagram induced by the ***intrinsic distance metric*** in Σ, defined to be the Euclidean length of the shortest path between each pair of points in Σ, where the paths are restricted to lie on Σ. (This metric is sometimes called the ***geodesic distance***, but when Σ is not smooth, the intrinsic shortest path is not necessarily a geodesic.) If we dualize the ***intrinsic Voronoi diagram*** of a point set $S \subset \Sigma$, we obtain an ***intrinsic Delaunay triangulation*** in which each dual triangle is ***intrinsic Delaunay***, which implies that there is an empty geodesic ball on the surface Σ whose boundary passes through the triangle's vertices. The intrinsic Delaunay triangulation is not always a "proper" triangulation—it might include loop edges or multi-edges.

If Σ is itself a triangulated surface and S is its vertex set, the intrinsic Delaunay triangulation of S might or might not coincide with the triangles that make up Σ (even if all of the triangles of Σ are Delaunay in the usual 3D sense). If they do coincide, so the triangles comprising Σ are all intrinsic Delaunay, we say that Σ is ***self-Delaunay***.

Bobenko and Springborn [BS07] show that the intrinsic Delaunay triangulation on a piecewise linear surface Σ in $\mathbb{R}^3$ is unique, and that the corresponding Laplace–Beltrami operator always has positive weights, even if the intrinsic triangulation is not proper; this is useful in geometry processing. Dyer, Zhang and Möller [DZM07] give an algorithm that takes a triangulated surface Σ in $\mathbb{R}^3$ and produces a geometrically similar self-Delaunay mesh, using a combination of flipping and vertex insertion. A recent preprint by Boissonnat, Dyer and Ghosh [BDG13] describes an algorithm that takes a smooth k-dimensional manifold Σ in $\mathbb{R}^d$ as input and produces a self-Delaunay surface triangulation, for any integers $0 < k < d$.

29.6 THREE-DIMENSIONAL POLYHEDRA

In this section we discuss the triangulation (or ***tetrahedralization***) of 3D polyhedra. A polyhedron P is a flat-sided solid, usually assumed to be connected and to satisfy the following nondegeneracy condition: around any point on the boundary of P, a sufficiently small ball contains one connected component of each of the interior and exterior of P. With this assumption, the numbers of vertices, edges, and faces (facets) of P are linearly related by Euler's polyhedron formula (adjusted for genus if appropriate).

GLOSSARY

Dihedral angle: The angle separating two polyhedral faces meeting at a shared edge, measured on a plane normal to the shared edge.

Reflex edge: An edge with interior dihedral angle greater than 180°.

Convex polyhedron: A polyhedron without reflex edges.

General polyhedron: Multiple components, cavities, and handles (higher genus) are permitted.

Circumsphere: The sphere through the vertices of a tetrahedron.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

BAD EXAMPLES

Three dimensions is not as nice as two. Triangulations of the same input may contain different numbers of tetrahedra. For example, a triangulation of an n-vertex convex polyhedron may have as few as $n-3$ or as many as $\binom{n-2}{2}$ tetrahedra. Below et al. [BLR00] proved that finding the minimum number of tetrahedra needed to triangulate (without Steiner points) a convex polyhedron is NP-complete. And when we move to nonconvex polyhedra, we get an even worse surprise: some cannot even be triangulated without Steiner points.

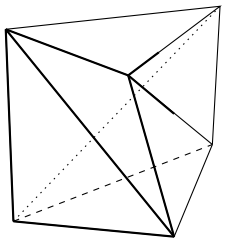


FIGURE 29.6.1
A twisted prism cannot be triangulated without Steiner points.

Schönhardt's polyhedron, shown in Figure 29.6.1, is the simplest example of a polyhedron that cannot be triangulated. Ruppert and Seidel [RS92] prove that it is NP-complete to determine whether a polyhedron can be triangulated without Steiner points, or to test whether k Steiner points suffice.

Chazelle [Cha84] gives an n-vertex polyhedron that requires $\Omega(n^2)$ Steiner points. This polyhedron is a box with thin wedges removed from the top and bottom faces (Figure 29.6.2). The tips of the wedges nearly meet at the hyperbolic surface $z = xy$. From a top view, the wedges appear to subdivide the polyhedron into $\Theta(n^2)$ small squares. We can exhibit $\Theta(n^2)$ points (one in each square) such that no two can see each other within the polyhedron, implying that every subdivision of the polyhedron into convex cells has $\Omega(n^2)$ cells.

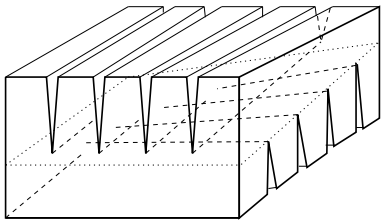


FIGURE 29.6.2
A polyhedron that requires $\Omega(n^2)$ tetrahedra. (From [BE95]*, with permission.)*

TRIANGULATIONS OF POLYHEDRA

Any polyhedron can be triangulated with $O(n^2)$ tetrahedra, matching the lower bound. An algorithm called ***vertical decomposition*** shoots vertical walls up and down from each edge of the polyhedron boundary; walls stop when they reach some other part of the boundary. The tops and bottoms of the resulting "cylinders" are then triangulated to produce $O(n^2)$ triangular prisms, which can each be trian-

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

gulated with a single interior Steiner point. A better algorithm first plucks off "pointed vertices" with unhindered "caps" [CP90]. Such a vertex, together with its incident faces, forms an empty convex cone. This algorithm uses $O(n + r^2)$ tetrahedra, where r is the number of reflex edges of the original polyhedron.

An alternative algorithm [Cha84] divides the polyhedron into convex solids by incrementally bisecting each reflex angle with a plane that extends away from the reflex angle in all directions until it first contacts the polyhedron boundary. This algorithm produces at most $O(nr + r^{7/3})$ tetrahedra [HS92].

SPECIAL POLYHEDRA

Any n-vertex convex polyhedron can be triangulated with at most $2n - 7$ tetrahedra by "starring" from a vertex. The region between two convex polyhedra (the convex hull of the union, minus the polyhedra), with a total of n vertices, can be triangulated without any Steiner points. If Steiner points are allowed, $O(n)$ tetrahedra suffice. The union of three convex polyhedra can also be tetrahedralized without Steiner points. The region between a convex polyhedron and a terrain can be triangulated with $O(n \log n)$ tetrahedra, and in fact, some such regions require $\Omega(n \log n)$ tetrahedra [CS94].

THREE-DIMENSIONAL MESH GENERATION

Mesh generation for three-dimensional solids is an important practical problem. Current approaches include advancing front methods, Delaunay refinement, octrees (the 3D generalization of quadtrees), bubble meshing, and mesh improvement methods, with none of them being clearly dominant. Some Delaunay and octree methods offer theoretical guarantees. A typical goal is to take a polyhedron input and generate a mesh of tetrahedra that are nicely shaped by some criterion—loosely speaking, they are not "skinny"—in which the number of tetrahedra is small.

Unfortunately, current state-of-the-art algorithms cannot guarantee that all the tetrahedra will have good angles, at least not in theory. (Often it is not difficult to produce good tetrahedra in practice, but there are no guarantees.) Most applications of tetrahedral meshes desire tetrahedra that have no dihedral angle close to 0° or 180° (which implies no solid angle is close to 0° or 360°). Unfortunately, known algorithms for meshing polyhedra can guarantee only very weak bounds on dihedral angles—bounds so weak they are typically not even stated, only proven to exist. Quite a few such algorithms have been published.

Many other algorithms offer no dihedral angle bounds, but instead guarantee that no tetrahedron will have a large ***radius-edge ratio***, which is the radius of the tetrahedron's circumscribing sphere divided by the length of its shortest edge. This guarantee rules out most skinny tetrahedron shapes, but ***slivers*** may remain. A sliver is a tetrahedron whose four vertices lie nearly on a common circle (although the vertices are not exactly coplanar), so that two of its dihedral angles are close to 180° and four are close to 0°. Although bounds on radius-edge ratios may seem disappointing compared to bounds on dihedral angles, they lead to algorithms that are popular in practice because the bounds are tight enough to eliminate most bad tetrahedra and slivers are easy to eliminate in practice, albeit not in theory.

The earliest example of an algorithm offering weak dihedral angle bounds re-

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

mains important as a theoretical exemplar. Mitchell and Vavasis [MV92] give an octree method that generalizes the quadtree mesh generation algorithm of Bern et al. [BEG94] to polyhedra. In addition to proving that their algorithm generates tetrahedra whose dihedral angles are bounded, Mitchell and Vavasis guarantee that the number of tetrahedra in the mesh is asymptotically "optimal," in the sense that it is within a constant factor of the best possible number for any mesh satisfying the same angle bounds.

Most (but not all) published algorithms that offer bounds on radius-edge ratios use Delaunay triangulations. Shewchuk [She98, CDS12] generalizes Ruppert's 2D Delaunay refinement algorithm to take a polyhedron (with no dihedral angle smaller than 90°) and generate a mesh whose tetrahedra's radius-edge ratios do not exceed 2. Variants of this algorithm have been implemented and found to work well in practice, for instance in Si's software *TetGen* [Si15].

Some Delaunay-based algorithms also offer weak dihedral angle bounds. Slivers can be removed by carefully perturbing the point set, either geometrically [Che97, ELM+00] or by adding small weights to the points and using a weighted Delaunay triangulation, a technique called ***sliver exudation*** [Ede01, LT01, CDS12]. In all algorithms for meshing polyhedra that offer dihedral angle bounds, the bounds are too weak to be reassuring to practitioners, but sliver exudation often performs substantially better in practice than the theory suggests [EG02]. Guaranteed sliver elimination with bounds strong enough to be meaningful in practice remains an important unsolved problem.

There is a special case where it is possible to obtain strong bounds on dihedral angles: when the input domain is not a polyhedron, but a region bounded by a smooth surface. By deforming a body-centered cubic lattice or octree, the ***isosurface stuffing*** algorithm of Labelle and Shewchuk [LS07] generates a mesh of tetrahedra whose dihedral angles are bounded between 10.7° and 164.8°.

Constrained Delaunay triangulations extend uneasily to $\mathbb{R}^3$, because as Schönhardt's and Chazelle's polyhedra remind us, not every polyhedron even has a triangulation without Steiner points. Shewchuk [She08] considers adding vertices on the polyhedron's edges. Given a polyhedron X, call a tetrahedron t *constrained Delaunay* if $t \subseteq X$, t's vertices are vertices of X, and t has a circumsphere that encloses no vertex of X visible from any point in the relative interior of t. A CDT of X is a triangulation of X whose tetrahedra are all constrained Delaunay. Call an edge of X *strongly Delaunay* if there exists a closed ball that contains the edge's two vertices but contains no other vertex of X. Shewchuk's *CDT Theorem* states that if every edge of X is strongly Delaunay, then X has a CDT. If a polyhedron does not have a CDT, one can subdivide its edges with new vertices until it has one. Shewchuk and Si [SS14] use the CDT Theorem and this notion of constrained Delaunay triangulation as part of a Delaunay refinement algorithm (implemented in *TetGen*) that is particularly effective for polyhedra that have small input angles.

OPEN PROBLEMS

1. Can the region between k convex polytopes, with n vertices in total, be (Steiner) triangulated with $O(n + k^2)$ tetrahedra?
2. Give an *input-sensitive* tetrahedralization algorithm for polyhedra, for example, one that uses only $O(1)$ times the smallest possible number of tetrahedra.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

3. Give a polynomial bound (or even a simple-to-state bound depending upon geometry) on the number of Steiner points needed to make all edges of a polyhedron strongly Delaunay.
4. Give an algorithm for computing tetrahedralizations of point sets or polyhedra, such that each tetrahedron contains its own circumcenter. This condition guarantees a desirable matrix property for a finite-volume formulation of an elliptic partial differential equation [Ber02].

29.7 ARBITRARY DIMENSION

We now discuss triangulation algorithms for arbitrary dimension $\mathbb{R}^d$. In our big-O expressions, we consider the dimension d to be fixed.

GLOSSARY

Polytope: A bounded intersection of halfspaces in $\mathbb{R}^d$.

Face: A subpolytope of any dimension—a vertex, edge, 2D face, 3D face, etc.

Simplex: The convex hull of $d+1$ affinely independent points in $\mathbb{R}^d$.

Circumsphere: The hypersphere through the vertices of a simplex.

Flip: A local operation, sometimes called a geometric bistellar operation, that exchanges two different triangulations of $d+2$ points in $\mathbb{R}^d$.

TRIANGULATIONS OF POINT SETS

Delaunay triangulations and weighted Delaunay triangulations generalize to $\mathbb{R}^d$. Every simplex in the DT has a circumsphere (a circumscribing hypersphere) that encloses no input points. The lifting map generalizes as well, so any convex hull algorithm in dimension $d+1$ can be used to compute d-dimensional DTs. The incremental insertion algorithm also generalizes to $\mathbb{R}^d$; however, the edge flipping algorithm does not. Flips themselves do generalize to $\mathbb{R}^d$: a flip in $\mathbb{R}^d$ replaces a triangulation of $d+2$ points in convex position with another triangulation of the same points. For example, 5 points in convex position in $\mathbb{R}^3$ can be triangulated by two tetrahedra sharing a face or by three tetrahedra sharing an edge. Unfortunately, the natural generalization of the flip algorithm, which starts from an arbitrary triangulation in $\mathbb{R}^3$ and performs appropriate flips, can get stuck before reaching the DT [Joe89]. The most popular algorithm for computing DTs in three or more dimensions is randomized incremental insertion.

The lifting map can be used to show (from the Upper Bound Theorem for Polytopes) that an n-vertex DT in $\mathbb{R}^d$ contains at most $O(n^{\lceil d/2 \rceil})$ simplices. For practical applications such as interpolation, surface reconstruction, and mesh generation, however, the DT rarely attains its worst-case complexity. The DT of random points within a volume or on a convex surface in $\mathbb{R}^3$ has linear expected complexity, but on a nonconvex surface can have near-quadratic complexity [Eri03]. DT

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

complexity can also be bounded by geometric parameters such as the ratio between longest and shortest pairwise distances [Eri03].

Most 2D DT optimality properties do not generalize to higher dimensions. One exception: the DT minimizes the maximum radius of the simplices' smallest enclosing spheres. The ***smallest enclosing sphere*** of a simplex is always either the circumscribing sphere of the simplex or the smallest circumscribing sphere of some face of the simplex.

Of interest in algebraic geometry as well as computational geometry is the ***flip graph*** or ***triangulation space***, which has a vertex for each distinct triangulation of a point set and an edge for each flip. The flip graph is sometimes defined so that flips that remove or insert input vertices are permitted; for example, a tetrahedron in $\mathbb{R}^3$ can be split into four tetrahedra by the insertion of a vertex in its interior. If the flip graph includes these flips, then the flip graph of the regular triangulations of a point set has the structure of the skeleton of a high-dimensional polytope called the ***secondary polytope*** [BFS90, GKZ90]. Therefore, the flip graph is connected.

Unfortunately, the flip graph of *all* triangulations of a point set (including nonregular triangulations) is not well understood. Santos [San00] showed that for some point sets in $\mathbb{R}^5$ this flip graph is not connected, even if the points are in convex position (making it moot whether flips that remove or insert input vertices are permitted, as every triangulation includes all the input vertices). Moreover, in $\mathbb{R}^6$ the flip graph may even have an isolated vertex. The question remains open in $\mathbb{R}^3$ and $\mathbb{R}^4$.

The following is known about Steiner triangulations of point sets in $\mathbb{R}^d$. It is always possible to add $O(n)$ Steiner points, so that the DT of the augmented point set has size only $O(n)$. Moreover, there is always a nonobtuse Steiner triangulation containing at most $O(n^{\lceil d/2 \rceil})$ simplices, all of which are ***path simplices***: each includes a path of d pairwise orthogonal edges [BCER95].

TRIANGULATIONS OF POLYTOPES

Triangulations of polytopes in $\mathbb{R}^d$ arise in combinatorics and algebra [GKZ90, Sta80]. Several algorithms are known for triangulating the hypercube, but there is a gap between the algorithm that produces the least number of simplices and the best lower bound on the number of simplices [OS03]; see Section 16.7.2. It is known that the region between two convex polytopes—a nonconvex polytope—can always be triangulated without Steiner points [GP88]; see Section 16.3.2. Below et al. [BBLR00] have shown that there can be a significant difference (linear in the number of vertices) in the minimum numbers of simplices in a triangulation and a ***dissection*** of a convex polytope (of dimension 3 or greater), which is a partition of a polytope into simplices whose faces may meet only partially (for example, a triangle bordering two other triangles along one of its sides).

OPEN PROBLEMS

1. Is the flip graph of the triangulations of a point set or polytope in $\mathbb{R}^3$ or $\mathbb{R}^4$ always connected?
2. What is the maximum number of triangulations of a set of n points in $\mathbb{R}^d$?

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

For bounds on this maximum in $\mathbb{R}^2$, see Sharir and Sheffer [SS11] for an upper bound and Dumitrescu et al. [DSST13] for a lower bound.

3. Narrow the gap between the upper and lower bounds on the minimum number of simplices in a triangulation of the d-cube.

29.8 SOURCES AND RELATED MATERIAL

SURVEYS

For more complete descriptions and references, consult the following sources.

[Aur91]: Generalizations of the Voronoi diagram and Delaunay triangulation.

[Ber02]: A survey of mesh generation algorithms.

[CDS12]: A book on Delaunay triangulations and Delaunay refinement algorithms for mesh generation.

[DRS10]: A book on the combinatorics of triangulations, especially regular triangulations, in arbitrary dimensions.

[Ede01]: A book on geometry and topology relevant to triangular and tetrahedral mesh generation.

The web is a rich source on mesh generation and triangulation; see Chapter 67.

RELATED CHAPTERS

Chapter 16: Subdivisions and triangulations of polytopes
Chapter 23: Computational topology of graphs on surfaces
Chapter 27: Voronoi diagrams and Delaunay triangulations
Chapter 30: Polygons
Chapter 35: Curve and surface reconstruction

REFERENCES

[Aur91] F. Aurenhammer. Voronoi diagrams—A survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23:345–405, 1991.

[BBLR00] A. Below, U. Brehm, J.A. De Loera, J. Richter-Gebert. Minimal simplicial dissections and triangulations of convex 3-polytopes. *Discrete Comput. Geom.*, 24:35–48, 2000.

[BCER95] M. Bern, L.P. Chew, D. Eppstein, and J. Ruppert. Dihedral bounds for mesh generation in high dimensions. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms*, pages 189–196, 1995.

[BDG13] J.-D. Boissonnat, R. Dyer, and A. Ghosh. Constructing intrinsic Delaunay triangulations of submanifolds. Technical Report RR-8273, INRIA, Sophia Antipolis, 2013.

[BE95] M. Bern and D. Eppstein. Mesh generation and optimal triangulation. In D.-Z. Du and F.K. Hwang, editors, *Computing in Euclidean Geometry*, 2nd edition, pages 47–123, World Scientific, Singapore, 1995.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

[BEE+93] M. Bern, H. Edelsbrunner, D. Eppstein, S.A. Mitchell, and T.-S. Tan. Edge-insertion for optimal triangulations. *Discrete Comput. Geom.*, 10:47–65, 1993.

[BEG94] M. Bern, D. Eppstein, and J.R. Gilbert. Provably good mesh generation. *J. Comput. Syst. Sci.*, 48:384–409, 1994.

[Ber02] M. Bern. Adaptive mesh generation. In T. Barth and H. Deconinck, editors, *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, pages 1–56, Springer-Verlag, Berlin, 2002.

[BFS90] L. Billera, P. Filliman, and B. Sturmfels. Constructions and complexity of secondary polytopes. *Adv. Math.*, 83:155–179, 1990.

[BGR88] B.S. Baker, E. Grosse, and C.S. Rafferty. Nonobtuse triangulation of polygons. *Discrete Comput. Geom.*, 3:147–168, 1988.

[Bis16] C.J. Bishop. Nonobtuse triangulations of PSLGs. *Discrete Comput. Geom.*, 56:43–92, 2016.

[BLR00] A. Below, J.A. De Loera, and J. Richter-Gebert. Finding minimal triangulations of convex 3-polytopes is NP-hard. In *Proc. 11th ACM-SIAM Sympos. Discrete Algorithms*, pages 65–66, 2000.

[BMR94] M. Bern, S.A. Mitchell, and J. Ruppert. Linear-size nonobtuse triangulation of polygons. In *Proc. 10th Sympos. Comput. Geom.*, pages 221–230, ACM Press, 1994.

[BO05] J.-D. Boissonnat and S. Oudot. Provably good sampling and meshing of surfaces. *Graphical Models*, 67:405–451, 2005.

[BS07] A.I. Bobenko and B.A. Springborn. A discrete Laplace–Beltrami operator for simplicial surfaces. *Discrete Comput. Geom.*, 38:740–756, 2007.

[CDS12] S.-W. Cheng, T.K. Dey, and J.R. Shewchuk. *Delaunay Mesh Generation*. CRC Press, Boca Raton, 2012.

[Cha84] B. Chazelle. Convex partitions of polyhedra: A lower bound and worst-case optimal algorithm. *SIAM J. Comput.*, 13:488–507, 1984.

[Cha91] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete Comput. Geom.*, 6:485–524, 1991.

[Che89] L.P. Chew. Constrained Delaunay triangulations. *Algorithmica*, 4:97–108, 1989.

[Che97] L.P. Chew. Guaranteed-quality Delaunay meshing in 3D. In *Proc. 13th Sympos. Comput. Geom.*, pages 391–393, ACM Press, 1997.

[CP90] B. Chazelle and L. Palios. Triangulating a nonconvex polytope. *Discrete Comput. Geom.*, 5:505–526, 1990.

[CS94] B. Chazelle and N. Shouraboura. Bounds on the size of tetrahedralizations. In *Proc. 10th Sympos. Comput. Geom.*, pages 231–239, ACM Press, 1994.

[CW98] F.Y.L. Chin and C.A. Wang. Finding the constrained Delaunay triangulation and constrained Voronoi diagram of a simple polygon in linear time. *SIAM J. Computing*, 28:471–486, 1998.

[DKM97] M.T. Dickerson, J.M. Keil, and M.H. Montague. A large subgraph of the minimum weight triangulation. *Discrete Comput. Geom.*, 18:289–304, 1997.

[DRS10] J.A. De Loera, J. Rambau, and F. Santos. *Triangulations: Structures for Algorithms and Applications*. Springer-Verlag, Berlin, 2010.

[DSST13] A. Dumitrescu, A. Schulz, A. Sheffer, and C. D. Tóth. Bounds on the maximum multiplicity of some common geometric graphs. *SIAM J. Discrete Math.*, 27:802–826, 2013.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.

[DZM07] R. Dyer, H. Zhang, and T. Möller. Delaunay mesh construction. In *Proc. 5th Sympos. Geom. Processing*, pages 273–282, Eurographics, 2007.

[Ede01] H. Edelsbrunner. *Geometry and Topology for Mesh Generation*. Cambridge University Press, 2001.

[EG02] H. Edelsbrunner and D. Guoy. An experimental study of sliver exudation. *Engineering with Computers*, 18:229–240, 2002.

[ELM+00] H. Edelsbrunner, X.-Y. Li, G. Miller, A. Stathopoulos, D. Talmor, S.-H. Teng, A. Üngör, and N.J. Walkington. Smoothing and cleaning up slivers. In *Proc. 32nd ACM Sympos. Theory Comput.*, pages 273–278, 2000.

[Epp94] D. Eppstein. Approximating the minimum weight triangulation. *Discrete Comput. Geom.*, 11:163–191, 1994.

[Eri03] J. Erickson. Nice point sets can have nasty Delaunay triangulations. *Discrete Comput. Geom.*, 30:109–132, 2003.

[ES97] H. Edelsbrunner and N.R. Shah. Triangulating topological spaces. *Internat. J. Comput. Geom. Appl.*, 7:365–378, 1997.

[ET91] H. Edelsbrunner and T.-S. Tan. A quadratic time algorithm for the minmax length triangulation. In *Proc. 32nd IEEE Sympos. Found. Comput. Sci.*, pages 414–423, 1991.

[ETW92] H. Edelsbrunner, T.-S. Tan, and R. Waupotitsch. A polynomial time algorithm for the minmax angle triangulation. *SIAM J. Sci. Statist. Comput.*, 13:994–1008, 1992.

[For95] S.J. Fortune. Voronoi diagrams and Delaunay triangulations. In F.K. Hwang and D.-Z. Du, editors, *Computing in Euclidean Geometry*, 2nd edition, pages 225–265, World Scientific, Singapore, 1995.

[GKZ90] I.M. Gelfand, M.M. Kapranov, and A.V. Zelevinsky. Newton polytopes of the classical discriminant and resultant. *Adv. Math.*, 84:237–254, 1990.

[GP88] J.E. Goodman and J. Pach. Cell decomposition of polytopes by bending. *Israel J. Math.*, 64:129–138, 1988.

[HS92] J. Hershberger and J. Snoeyink. Convex polygons made from few lines and convex decompositions of polyhedra. In *Proc. 3rd Scand. Workshop Algorithm Theory*, vol. 621 of *LNCS*, pages 376–387, Springer-Verlag, Berlin, 1992.

[Joe89] B. Joe. Three-dimensional triangulations from local transformations. *SIAM J. Sci. Stat. Comput.*, 10:718–741, 1989.

[KL93] R. Klein and A. Lingas. A linear-time randomized algorithm for the bounded Voronoi diagram of a simple polygon. *Internat. J. Comput. Geom. Appl.*, 6:263–278, 1996.

[Kli80] G.T. Klincsek. Minimal triangulations of polygonal domains. *Annals Discrete Math.*, 9:121–123, 1980.

[LK98] C. Levcopoulos and D. Krznaric. Quasi-greedy triangulations approximating the minimum weight triangulation. *J. Algorithms*, 27:303–338, 1998.

[LL86] D.T. Lee and A.K. Lin. Generalized Delaunay triangulation for planar graphs. *Discrete Comput. Geom.*, 1:201–217, 1986.

[LL92] C. Levcopoulos and A. Lingas. Fast algorithms for greedy triangulation. *BIT*, 32:280–296, 1992.

[LS07] F. Labelle and J.R. Shewchuk. Isosurface stuffing: Fast tetrahedral meshes with good dihedral angles. *ACM Trans. Graph.*, 26:57, 2007.

[LT01] X.-Y. Li and S.-H. Teng. Generating well-shaped Delaunay meshes in 3D. In *Proc. 12th ACM-SIAM Sympos. Discrete Algorithms*, pages 28–37, 2001.

[Mit93] S.A. Mitchell. Refining a triangulation of a planar straight-line graph to eliminate large angles. In *Proc. 34th IEEE Sympos. Found. Comput. Sci.*, pages 583–591, 1993.

[MR08] W. Mulzer and G. Rote. Minimum weight triangulation is NP-hard. *J. ACM*, 55:11, 2008.

[Mul94] K. Mulmuley. *Computational Geometry: An Introduction through Randomized Algorithms*. Prentice-Hall, Englewood Cliffs, 1994.

[MV92] S.A. Mitchell and S.A. Vavasis Quality mesh generation in three dimensions. In *Proc. 8th Sympos. Comput. Geom.*, pages 212–221, ACM Press, 1992.

[ORY05] S. Oudot, L. Rineau, and M. Yvinec. Meshing volumes bounded by smooth surfaces. In *Proc. 14th International Meshing Roundtable*, pages 203–219, Springer, Berlin, 2005.

[OS03] D. Orden and F. Santos. Asymptotically efficient triangulations of the d-cube. *Discrete Comput. Geom.*, 30:509–528, 2003.

[PS85] F.P. Preparata and M.I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, 1985.

[RS92] J. Ruppert and R. Seidel. On the difficulty of tetrahedralizing 3-dimensional non-convex polyhedra. *Discrete Comput. Geom.*, 7:227–253, 1992.

[RS09] J. Remy and A. Steger. A quasi-polynomial time approximation scheme for minimum weight triangulation. *J. ACM*, 56:15, 2009.

[Rup95] J. Ruppert. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. Algorithms*, 18:548–585, 1995.

[San00] F. Santos. A point set whose space of triangulations is disconnected. *J. Amer. Math. Soc.*, 13:611–637, 2000.

[SB15] J.R. Shewchuk and B.C. Brown. Fast segment insertion and incremental construction of constrained Delaunay triangulations. *Comput. Geom.*, 48:554–574, 2015.

[Sei88] R. Seidel. Constrained Delaunay triangulations and Voronoi diagrams with obstacles. In H. S. Poingratz and W. Schinnerl, editors, *1978–1988 Ten Years IIG*, pages 178–191, Institute for Information Processing, Graz University of Technology, 1988.

[She98] J.R. Shewchuk. Tetrahedral mesh generation by Delaunay refinement. In *Proc. 14th Sympos. Comput. Geom.*, pages 86–95, ACM Press, 1998.

[She08] J.R. Shewchuk. General-dimensional constrained Delaunay triangulations and constrained regular triangulations, I: Combinatorial properties. *Discrete Comput. Geom.*, 39:580–637, 2008.

[Si15] H. Si. TetGen, a Delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Software*, 41:11, 2015.

[SS11] M. Sharir and A. Sheffer. Counting triangulations of planar point sets. *Electr. J. Comb.*, 18(1), 2011.

[SS14] J.R. Shewchuk and H. Si. Higher-quality tetrahedral mesh generation for domains with small angles by constrained Delaunay refinement. In *Proc. 13th Sympos. Comput. Geom.*, pages 290–299, ACM Press, 2014.

[Sta80] R.P. Stanley. Decompositions of rational convex polytopes. *Annals Discrete Math.*, 6:333–342, 1980.

[Tan94] T.-S. Tan. An optimal bound for conforming quality triangulations. In *Proc. 10h Sympos. Comput. Geom.*, pages 240–249, ACM Press, 1994.

[Xia13] G. Xia. The stretch factor of the Delaunay triangulation is less than 1.998. *SIAM J. Comput.*, 42:1620–1659, 2013.

Preliminary version (July 28, 2017). To appear in the Handbook of Discrete and Computational Geometry, J.E. Goodman, J. O'Rourke, and C. D. Tóth (editors), 3rd edition, CRC Press, Boca Raton, FL, 2017.