Chapter 9

Weighted Delaunay refinement for PLCs with small angles

The Delaunay refinement algorithm DelTetPLC in the previous chapter requires that the input PLC satisfy the projection condition, which rules out linear cells adjoining each other at dihedral angles or plane angles less than 90°. For many engineering applications, this restriction is not acceptable. Unfortunately, DelTetPLC, like Ruppert's original algorithm, can fail to terminate because of ping-pong encroachment (recall Figure 6.12). Ruppert's "modified segment splitting using concentric circular shells," described in Section 6.6, extends easily to spherical shells, and it copes reasonably well with segments that meet at small plane angles in tetrahedral meshes. It is not a complete solution, because of the seditious edges discussed in Section 6.6.

Three-dimensional domains introduce the additional danger of ping-pong encroachment among adjoining polygons, which is substantially harder to control than mutual encroachment among segments, especially if many polygons share a single segment. Efforts have been made to adapt spherical or cylindrical shells for the protection of segments where such polygons meet at small dihedral angles. However, algorithms for constructing Steiner Delaunay triangulations of three-dimensional PLCs often place undesirably many vertices in the vicinity of small domain angles and create undesirably short edges—even when they are not concerned with the quality of the tetrahedra.

This chapter offers an alternative approach to small domain angles that takes advantage of the properties of the weighted Delaunay triangulation, and does not overrefine as algorithms that use shells often do. We present here a new Delaunay refinement algorithm (it has not previously appeared in the literature) that generates a graded mesh in which no tetrahedron has a radius-edge ratio greater than 2, except possibly tetrahedra that adjoin a vertex or segment where two linear cells in the PLC meet at an acute angle. The tactic is to place positively weighted vertices at those meeting points. A weighted Delaunay triangulation tends to preferentially create edges and triangles adjoining more strongly weighted vertices, thereby helping to enforce domain conformity at the cost of locally sacrificing quality.

Because we use weighted vertices and a weighted Delaunay triangulation, we insert new vertices at orthocenters of simplices instead of their circumcenters. Orthocenters are

further from the highly weighted vertices and closer to the unweighted vertices than circumcenters are. The weighted vertices act as protecting balls that prevent new vertices from being inserted too close to an apex of a small angle. The cycle of mutual encroachment in Figure 6.12 cannot occur because the apex is protected by a ball in which no new vertex can appear. Although we must compromise on tetrahedron quality near the apex, we guarantee that no tetrahedron has an orthoradius greater than twice the length of its shortest edge. The value of this guarantee is weakest where the vertex weights are greatest. Tetrahedra that do not adjoin small domain angles do not have weighted vertices, so their radius-edge ratios are at most 2, as usual.

9.1 The ideas behind weighted Delaunay refinement

Recall that each weighted vertex $v[\omega_v]$ is represented by a ball $B_v = B(v, \sqrt{\omega_v})$. We call these balls *protecting balls*. The weighted Delaunay refinement algorithm creates a weighted vertex with the intention that no vertex shall ever be inserted inside its protecting ball. It never assigns a vertex a negative weight. Most of the vertices that the algorithm creates are *unweighted*, meaning they have weight zero; we conceive of their protecting balls as being single points.

Weighted Delaunay refinement is similar to ordinary Delaunay refinement, but encroachment is defined in terms of orthospheres instead of circumspheres, and new vertices are inserted at orthocenters instead of circumcenters. Roughly speaking—we will be more precise later—a simplex σ with vertices from a weighted point set $S[\omega]$ is encroached if some other vertex in $S[\omega]$ is orthogonal or closer than orthogonal to the diametric orthoball of σ. In particular, an unweighted vertex must lie in σ's diametric orthoball to encroach upon σ. An encroached simplex is split with a new vertex at its orthocenter. The new vertex is not too close to any other vertex, because no vertex lies in the interior of that simplex's diametric orthoball, as no vertex has negative weight.

Several other observations are crucial to understanding the algorithm. First, if the intersection of the interiors of the protecting balls centered at σ's vertices is nonempty, then σ has an imaginary orthoradius. It follows that no unweighted vertex can encroach upon σ. The refinement stage of the algorithm inserts only unweighted vertices, so simplices with imaginary orthoradii are invulnerable to encroachment or to removal from the weighted Delaunay triangulation. In terms of the parabolic lifting map, the entirety of σ^+ is below the paraboloid, so no vertex inserted on the paraboloid can remove σ^+ from the convex hull. Our algorithm begins with a protection stage that identifies segments where polygons meet at acute dihedral angles and covers them with overlapping protecting balls, thereby ensuring that their subsegments cannot be split.

Second and conversely, if the intersection of the interiors of the protecting balls centered at σ's vertices is empty, then σ's orthocenter does not lie inside its vertices' protecting balls. If in addition σ is weighted Delaunay, then σ's orthocenter does not lie inside any protecting ball whatsoever. Therefore, splitting an encroached simplex never entails the risk of inserting a new vertex inside a protecting ball. Neither does splitting a tetrahedron with positive orthoradius.

Third, a tetrahedron is split if its *orthoradius-edge ratio*, its orthoradius divided by the length of its shortest edge, exceeds a threshold $\bar{\rho}$. If a tetrahedron has no weighted vertex, its orthoradius is its circumradius. Therefore, most tetrahedra in the final mesh have good circumradius-edge ratios, but tetrahedra that adjoin the apex of an acute domain angle might not. This is a reasonable compromise, as it is sometimes impossible to achieve high quality near small domain angles.

The weighted Delaunay refinement algorithm, called DelTetAcutePLC, takes as input a function $\lambda : |\mathcal{P}| \to \mathbb{R}$, called a *size field*, that reflects the user's locally desired spacing of vertices in the domain. The algorithm might be forced to generate much shorter edges in the vicinity of small angles and small geometric features, but it will not leave behind edges that are substantially longer than the user requests. A user desiring the sparsest possible mesh can simply set $\lambda \equiv \infty$. We require that $\inf_{x \in |\mathcal{P}|} \lambda(x) > 0$.

DelTetAcutePLC begins with a protection stage called Protect that centers protecting balls on vertices and segments in $\mathcal{P}$ where linear cells meet at acute angles. A protected segment is covered by a union of protecting balls. The initial vertex set $S[\omega]$ contains a weighted vertex $v[r^2]$ for each protecting ball $B(v, r)$ and an unweighted vertex for each vertex in $\mathcal{P}$ that is not protected. When the protection stage is done, DelTetAcutePLC constructs the weighted Delaunay triangulation $\operatorname{Del} S[\omega]$ and executes a refinement stage called Refine, which inserts unweighted vertices into $\operatorname{Del} S[\omega]$ to enforce domain conformity and eliminate poor-quality tetrahedra.

DelTetAcutePLC($\mathcal{P}, \lambda, \bar{\rho}$)

1. $S[\omega] \leftarrow$ Protect($\mathcal{P}, \lambda$).
2. Construct $\operatorname{Del} S[\omega]$.
3. $S[\omega] \leftarrow$ Refine($\mathcal{P}, S[\omega], \lambda, \bar{\rho}$).
4. Return the mesh $\{\sigma \in \operatorname{Del} S[\omega] : \sigma \subseteq |\mathcal{P}|\}$.

9.2 Protecting vertices and segments

The protection stage, implemented by the Protect pseudocode below, first identifies every vertex v in $\mathcal{P}$ that adjoins two linear cells that fail the projection condition (Definition 8.3), and for each such vertex constructs a protecting ball $B_v = B(v, \min\{\lambda(v), f(v)/(2\sqrt{2})\})$, where f is the local feature size function. Then, it identifies every segment in $\mathcal{P}$ that is an edge of two polygons in $\mathcal{P}$ that fail the projection condition, and covers each such segment with protecting balls (with the subroutine Cover). We call these specially treated vertices and segments *acute*. Clearly, both vertices of an acute segment are acute.

Protect($\mathcal{P}, \lambda$)

1. Let $S[\omega]$ be an empty point set.
2. For each vertex v in $\mathcal{P}$, determine whether v is acute. If so, add the weighted point $v[\omega_v]$ to $S[\omega]$ to represent the protecting ball $B_v = B(v, \sqrt{\omega_v})$, where $\sqrt{\omega_v} = \min\{\lambda(v), f(v)/(2\sqrt{2})\}$. Otherwise, add the unweighted point v to $S[\omega]$.

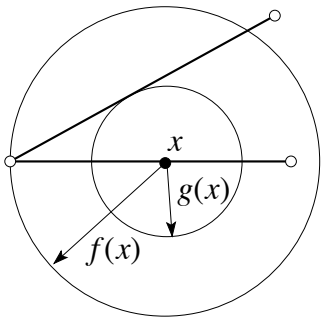


Figure 9.1: Local gap size $g(x)$ versus local feature size $f(x)$.

3. For each segment uv in $\mathcal{P}$, if uv is acute, call Cover$(S[\omega], u, v)$.
4. Return the weighted point set $S[\omega]$.

To set the radii of the protecting balls on segments, we use the notion of the local gap size.

Definition 9.1 (local gap size)**.** For every $x \in |\mathcal{P}|$, *the local gap size* $g(x)$ is the smallest radius $r > 0$ such that $B(x, r)$ intersects two linear cells in $\mathcal{P}$, one of which does not contain x.

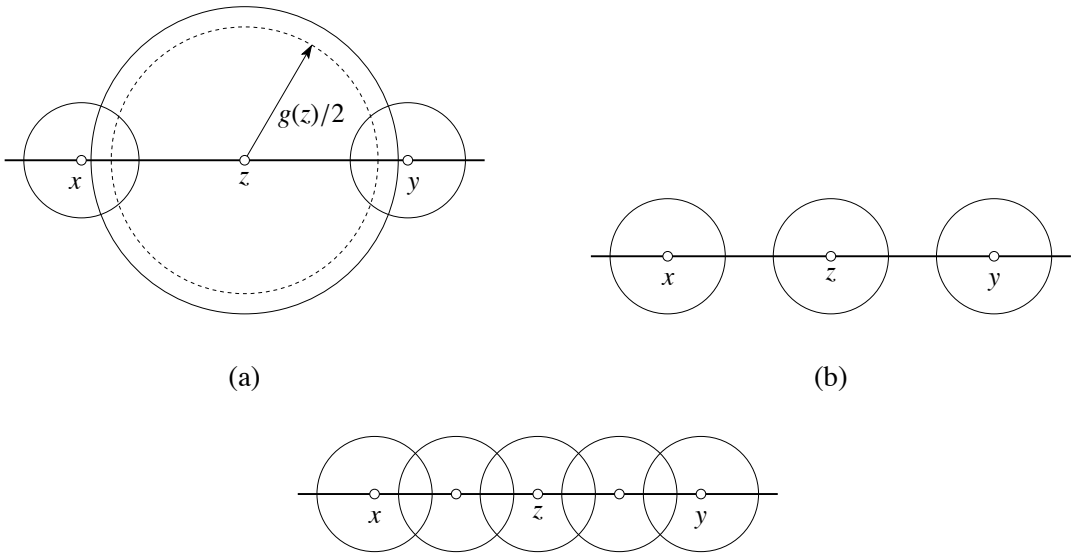
Figure 9.1 illustrates the difference between the local gap size and the local feature size. At a vertex v in $\mathcal{P}$, the two notions coincide: $g(v) = f(v)$. Observe that g is not continuous: it approaches zero as x approaches a vertex, but is nonzero at the vertex. However, g is 1-Lipschitz in the interior of a segment.

We require the radius of every protecting ball with center x to be less than or equal to $g(x)/2$. Hence, no protecting ball centered on the interior of a segment s may reach more than halfway to a segment other than s, nor halfway to a polygon that does not include s. This restriction limits the ability of protecting balls to encroach upon subsegments and subpolygons, and ensures that no three protecting balls have interiors with a common intersection.

A recursive procedure Cover covers each acute segment in $\mathcal{P}$ with protecting balls such that any two consecutive balls are orthogonal. Let B_x and B_y be two balls that have already been placed on a segment. Cover$(S[\omega], x, y)$ begins by computing the unique ball on xy orthogonal to both B_x and B_y. If that ball is not too big, it is accepted to cover the remainder of xy. Otherwise, Cover centers a smaller ball at a point z in the middle of the gap between B_x and B_y and recursively calls itself to cover xz and zy. To ensure that it will not produce an unduly small ball, Cover maintains the invariant that the gap between nonorthogonal balls is always at least $2\sqrt{2} - 2 \doteq 0.828$ times the radius of the smaller of the two balls that bookend the gap. Figure 9.2 illustrates its workings.

Cover$(S[\omega], x, y)$

1. Compute the center z on xy of the ball orthogonal to B_x and B_y with the relation
$$d(x, z) = \frac{d(x, y)^2 + \text{radius}(B_x)^2 - \text{radius}(B_y)^2}{2d(x, y)}.$$



(a) (b) (c)

Figure 9.2: (a) The middle solid ball is orthogonal to B_x and B_y. Because its radius exceeds $g(z)/2$, we (b) use a ball of radius proportional to the gap size and (c) recursively cover xz and zy.

Compute its radius $Z = \sqrt{d(x,z)^2 - \text{radius}(B_x)^2}$.

2. If $Z \leq \min\{\lambda(z), g(z)/2\}$, add the weighted point $z[Z^2]$ to $S[\omega]$ to represent the protecting ball $B(z, Z)$ and return.
3. Let $G = d(x,y) - \text{radius}(B_x) - \text{radius}(B_y)$ be the length of the gap between B_x and B_y. Let w be the point in the middle of the gap; i.e. $d(x,w) = \text{radius}(B_x) + G/2$. Let $W = \min\{\lambda(w), g(w)/2, G/(4\sqrt{2} - 2)\}$.
4. Add the weighted point $w[W^2]$ to $S[\omega]$ to represent the protecting ball $B(w, W)$, call Cover$(S[\omega], x, w)$, and call Cover$(S[\omega], w, y)$.

Cover ensures that protecting balls with consecutive centers on a segment are orthogonal. With this property, the subset of $\mathcal{P}$ in the union of the protecting balls can be triangulated with Delaunay tetrahedra whose aspect ratios have an upper bound that depends on the domain angles—that is, so that the protecting balls do not include arbitrarily thin slivers (see the notes at the end of the chapter). For simplicity, we will not make use of this property.

Protect and Cover must compute local gap sizes at points on acute vertices and segments. These can be determined by computing the distances from a point to all the linear cells in $\mathcal{P}$ that do not contain it and taking the minimum.

We show that Protect terminates and does not create unduly short edges by deriving a lower bound on the radius of every protecting ball. The following proposition is a preliminary step to that bound.

Proposition 9.1. PROTECT *and* COVER *maintain the invariant that if two balls have consecutive centers on a segment but are not orthogonal, there is a gap between them whose length is at least* $2\sqrt{2}-2 \doteq 0.828$ *times the radius of the smaller ball. The gap can accommodate a ball that is at least as large as the smaller neighbor and not closer than orthogonal to either neighbor.*

PROOF. For an acute segment uv, step 2 of PROTECT creates a ball B_u whose radius is at most $f(u)/(2\sqrt{2}) \le d(u,v)/(2\sqrt{2})$ and a ball B_v whose radius is also at most $d(u,v)/(2\sqrt{2})$. The gap between the two balls has length at least $(2\sqrt{2}-2)\,d(u,v)/(2\sqrt{2})$, so the invariant holds before COVER$(S[\omega],u,v)$ is called.

Step 3 of COVER places a ball of radius at most $G/(4\sqrt{2}-2)$ in the center of a gap of length G, leaving smaller gaps of length at least $(2\sqrt{2}-2)\,G/(4\sqrt{2}-2)$ on either side of it. Again the invariant holds.

If we place at the center of the gap a ball with the same radius as the smaller ball adjoining the gap, the distance from its center to the center of the smaller ball is at least $\sqrt{2}$ times their radius. Therefore, it is not closer than orthogonal to either neighbor. □

Proposition 9.2. *Let uv be an acute segment in $\mathcal{P}$. The radius of every protecting ball B_z centered on uv is between $c \cdot \inf_{x\in(uv\setminus(B_u\cup B_v))\cup\{u,v\}} \min\{\lambda(x), g(x)/2\}$ and $\min\{\lambda(z), g(z)/2\}$, where $c = \left(4\sqrt{2}-2\right)^{-1}\left(1-1/\sqrt{2}\right)^{1/2} > 1/7$. If $\inf_{x\in uv}\lambda(x)$ is positive,* COVER$(S[\omega],u,v)$ *terminates, whereupon uv is covered by protecting balls such that any two balls with consecutive centers are orthogonal.*

PROOF. Every ball placed by step 2 of PROTECT or step 2 or 4 of COVER satisfies the upper bound by construction. The radii of B_u and B_v, placed by step 2 of PROTECT, also satisfy the lower bound by construction.

Any ball placed by step 4 of COVER is centered at a point $w \in uv \setminus (B_u \cup B_v)$ at the midpoint of a gap of width G, and has radius either $W = \min\{\lambda(w), g(w)/2\}$, satisfying the lower bound, or $W = G/(4\sqrt{2}-2)$. Let us find a lower bound for W in the latter case. By Proposition 9.1, the gap is next to a ball B_y of radius at most $G/(2\sqrt{2}-2)$; let y be the center of B_y. The fact that COVER reached step 4 means that a ball B_z constructed by step 1 was rejected for being too large; specifically, $\text{radius}(B_z) > \min\{\lambda(z), g(z)/2\}$. As the center z of B_z lies in the gap, $d(y,z) \le G + \text{radius}(B_y)$. As B_y and B_z are orthogonal, $\text{radius}(B_z)^2 = d(y,z)^2 - \text{radius}(B_y)^2 \le G^2 + 2G\cdot\text{radius}(B_y) \le G^2 + G^2/(\sqrt{2}-1) = \sqrt{2}G^2/(\sqrt{2}-1)$. Combining inequalities yields $W \ge c\cdot\text{radius}(B_z) > c\cdot\min\{\lambda(z), g(z)/2\}$.

Every ball placed on uv by step 2 of COVER is at least as large as one of its two neighbors by Proposition 9.1. It satisfies the lower bound because its neighbor does.

The infimum $\inf_{x\in(uv\setminus(B_u\cup B_v))\cup\{u,v\}} g(x)$ is positive for every PLC. If $\inf_{x\in uv}\lambda(x)$ is positive, there is a constant lower bound on the size of every protecting ball, so COVER terminates. By design, it can terminate only when any two balls with consecutive centers are orthogonal. □

See the end of Section 9.4 for a discussion of the mysterious expression $\inf_{x\in(uv\setminus(B_u\cup B_v))\cup\{u,v\}} g(x)$ and its relationship to the small angles of $\mathcal{P}$.

In practice, it may be wise to modify step 2 of COVER to make Z somewhat smaller than $g(z)/2$ (perhaps $g(z)/3$) to make room for more Steiner points to be inserted between two segments meeting at a small angle, and thereby make room to create tetrahedra with good radius-edge ratios in the gap.

9.3 The refinement stage

After the protection stage ends, the refinement stage triangulates $\mathcal{P}$. Each protecting ball B_x is represented by a weighted point $x[\omega_x]$ in $S[\omega]$, where $\omega_x = \text{radius}(B_x)^2$. Unprotected vertices in $\mathcal{P}$ are represented by unweighted points in $S[\omega]$. DELTETACUTEPLC constructs the weighted Delaunay triangulation $\text{Del}\, S[\omega]$, and the procedure REFINE refines it by inserting new vertices, all unweighted. We recall the notions of subsegments, subpolygons, and encroachment from Chapter 8, but we adjust their definitions to treat weighted vertices.

Definition 9.2 (subsegment encroachment)**.** A vertex v *encroaches upon* a subsegment s if v is not a vertex of s and v has a nonpositive power distance from the diametric orthoball of s.

Definition 9.3 (subpolygon encroachment)**.** For a polygon $h \in \mathcal{P}$, if no subsegment of h is encroached, then the two-dimensional weighted Delaunay triangulation $\text{Del}\,(S[\omega] \cap h)$ includes a Steiner triangulation of h. The *subpolygons* of h are the triangles in this Steiner triangulation, whether they appear in $\text{Del}\, S[\omega]$ or not. A vertex $v \in S[\omega]$ *encroaches upon* a subpolygon σ of h if the power distance between $v[\omega_v]$ and the diametric orthoball of σ is nonpositive, unless v lies on $\text{aff}\, h$ and the power distance is zero. (The exception is analogous to the exception for cocircular vertices in Definition 8.2. If the weighted Delaunay subdivision of $S[\omega] \cap h$ contains a polygon that is not a triangle, each weighted Delaunay triangulation subdivides the polygon into triangles, and those triangles should not be encroached upon by each other's vertices.)

There are three differences between the REFINE stage and the Delaunay refinement algorithm in Chapter 8. First, encroachment is determined by orthoballs; second, new vertices are inserted at orthocenters. Even subsegments are split at their orthocenters rather than their midpoints.

SPLITWEIGHTEDSUBSEGMENT($e, S[\omega]$)

Insert the orthocenter of e into $S[\omega]$.

Observe that every subsegment of an acute segment has a diametric orthoball of imaginary radius and is never encroached, so no such subsegment is ever passed to SPLITWEIGHTEDSUBSEGMENT. Observe also that some segments are not acute but have one or even two vertices that are acute, and therefore weighted; it is these segments that necessitate inserting subsegment orthocenters rather than midpoints.

SPLITWEIGHTEDSUBPOLYGON($\sigma, S[\omega]$)

1. Let c be the orthocenter of σ.

2. If c encroaches upon a subsegment e, call SplitWeightedSubsegment($e, S[\omega]$) and return. Otherwise, insert c into $S[\omega]$.

The procedure for splitting encroached subpolygons is similar to that of Chapter 8: if no subsegment is encroached and a vertex v encroaches upon some subpolygon of a polygon h, the Monotone Power Lemma (Lemma 7.5) in Chapter 7 shows that v also encroaches upon the subpolygon σ of h that contains the orthogonal projection of v onto h. We split σ by inserting a new vertex at σ's orthocenter. Splitting σ rather than an arbitrary encroached subpolygon makes it possible to guarantee a better bound on the quality of the tetrahedra.

The third difference is that a tetrahedron is split if its orthoradius-edge ratio exceeds $\bar{\rho}$.

SplitWeightedTetrahedron($\tau, S[\omega]$)

1. Let c be the orthocenter of τ.
2. If c encroaches upon a subsegment e, call SplitWeightedSubsegment($e, S[\omega]$) and return.
3. If c encroaches upon a subpolygon of some polygon $h \in \mathcal{P}$, let σ be a subpolygon of h that contains the orthogonal projection of c onto h. Call SplitWeightedSubpolygon($\sigma, S[\omega]$) and return.
4. Insert c into $S[\omega]$.

The procedure Refine is the entry point for the refinement stage. Termination is guaranteed only if the bound $\bar{\rho}$ on the largest permissible orthoradius-edge ratio is 2 or greater. For simplicity, we omit from the pseudocode the necessary record-keeping of the subsegments, the subpolygons, and the mesh, addressed by the pseudocode in Chapter 8.

Refine($\mathcal{P}, S[\omega], \lambda, \bar{\rho}$)

1. While some vertex v in $S[\omega]$ encroaches upon a subsegment e, call SplitWeightedSubsegment($e, S[\omega]$) and repeat Step 1.
2. If some vertex v in $S[\omega]$ encroaches upon a subpolygon of a polygon $h \in \mathcal{P}$, let σ be a subpolygon of h that contains the orthogonal projection of v onto h. Call SplitWeightedSubpolygon($\sigma, S[\omega]$) and go to Step 1.
3. If $\mathrm{Del}\, S[\omega]$ contains a tetrahedron $\tau \subseteq |\mathcal{P}|$ whose orthoradius-edge ratio exceeds $\bar{\rho}$ or whose orthoradius exceeds $\lambda(c)$ where c is the orthocenter of τ, then call SplitWeightedTetrahedron($\tau, S[\omega]$) and go to Step 1.
4. Return $S[\omega]$.

SplitWeightedSubpolygon is invoked only when there is no encroached subsegment, in which case for every polygon h in $\mathcal{P}$, the subcomplex $\{\sigma \in \mathrm{Del}\, S[\omega] : \sigma \subseteq h\}$ triangulates h and is a quarantined complex. By the Orthocenter Containment Lemma (Lemma 7.7), h contains the orthocenters of all its subpolygons. Therefore, when SplitWeightedSubpolygon inserts the orthocenter of a subpolygon $\sigma \subseteq h$, the orthocenter refines h.

Likewise, SplitWeightedTetrahedron is invoked only when there is no encroached subsegment or subpolygon, in which case the subcomplex $\{\tau \in \mathrm{Del}\, S[\omega] : \tau \subseteq |\mathcal{P}|\}$ is a quarantined complex. By the Orthocenter Containment Lemma, $|\mathcal{P}|$ contains the orthocenter of every tetrahedron in it, so it is safe to insert the orthocenters.

9.4 A proof of termination and good grading

In this section, we prove that if $\bar{\rho} \geq 2$, then, DelTetAcutePLC terminates and returns a mesh of $\mathcal{P}$ in which no tetrahedron has an orthoradius-edge ratio greater than $\bar{\rho}$, hence no tetrahedron with unweighted vertices has a circumradius-edge ratio exceeding $\bar{\rho}$. We derive a lower bound on the distances between points in S, thereby showing that DelTetAcutePLC terminates and produces nicely graded meshes.

The edge lengths in the final mesh are proportional to the local feature size function for a modified version of $\mathcal{P}$ we call a *eunuch structure*. Let C be the set of vertices at centers of protecting balls, and let U be the union of the interiors of all the protecting balls. Let $\mathcal{E} = \{g \setminus U : g \in \mathcal{P}\} \cup C$ be a set of cells defined by removing from each linear cell every protecting ball, then adding the protecting ball centers as vertices. $\mathcal{E}$ is not generally a complex, as the boundary of a cell in $\mathcal{E}$ is not necessarily a union of cells in $\mathcal{E}$. It is important to keep it this way. Observe that $\mathcal{E}$ satisfies the projection condition, because if two adjoining cells in $\mathcal{P}$ fail the projection condition, their counterparts in $\mathcal{E}$ do not adjoin each other at all.

The cells in $\mathcal{E}$ are not necessarily polyhedra, but $\mathcal{E}$ has a well-defined local feature size function, denoted $f_{\mathcal{E}}$. Multiplied by the right constant, this function is a lower bound on the edge lengths in the mesh generated by DelTetAcutePLC (see Theorem 9.4). All vertices in that mesh lie on cells in $\mathcal{E}$. Thus $f_{\mathcal{E}}$ gives useful intuition on how mesh vertices are distributed.

The analysis follows the pattern established in Chapters 6 and 8. A vertex is of type i if it lies on a linear i-cell in $\mathcal{P}$ but not on a lower-dimensional cell. Every vertex in the mesh has a type, and so does every rejected orthocenter that SplitWeightedSubpolygon or SplitWeightedTetrahedron declines to insert because of encroachment. Vertices in $\mathcal{P}$ are of type 0. Other vertices lying on segments are of type 1, including vertices placed on segments by Cover and vertices inserted by SplitWeightedSubsegment. Vertices inserted or rejected by SplitWeightedSubpolygon are of type 2. Vertices inserted or rejected by SplitWeightedTetrahedron are of type 3. Protecting balls are centered on some vertices of types 0 and 1.

We take the notion of insertion radius from Chapters 6 and 8, but change the definition. The *insertion radius* r_x of a vertex x of type 0 or a weighted vertex (of type 0 or type 1), created during the protection stage, is the Euclidean distance from x to the nearest other vertex created during the protection stage. For a vertex x inserted during the refinement stage—every unweighted vertex not of type 0—the power distance replaces the Euclidean distance: r_x is the square root of the power distance from x to the vertex in S (weighted or unweighted) that is nearest to x by power distance, at the instant before x is inserted into S or rejected. In other words, $r_x^2 = \min_{y \in S} \pi(x, y) = \min_{y \in S} \left(d(x, y)^2 - \omega_y\right)$. When a vertex is inserted at the center of an orthoball of a weighted Delaunay simplex, the insertion radius of the vertex is equal to the radius of the orthoball.

Each unweighted vertex inserted during the refinement stage has a *parent* vertex, whose insertion radius helps to place a lower bound on the child's. Centers of protecting balls do not have parents. The parent of an unweighted type 1 or 2 vertex x, inserted at the orthocenter of an encroached subsegment or subpolygon, is the closest encroaching vertex by power distance; this vertex might be in S or might be a rejected circumcenter. The parent

of a type 3 vertex x, inserted at the orthocenter of a tetrahedron τ, is the most recently inserted vertex of τ's shortest edge. If both vertices were present from the start, choose one arbitrarily.

The following proposition places a lower bound on the insertion radius r_x of a vertex x in terms of $f_{\mathcal{E}}(x)$, $\lambda(x)$, and the insertion radius r_p of x's parent p. Recall that the relationship between r_x and r_p is the crux of the proof by induction of Proposition 8.5.

Proposition 9.3. *Let x be a vertex. Let p be its parent, if x has one.*

(i) *If x is a type 0 vertex or a weighted vertex, then $r_x \geq f_{\mathcal{E}}(x)$.*

(ii) *If x is an unweighted vertex of type $i \in \{1, 2\}$ and p is of type $j \leq i$, then $r_x \geq \sqrt{3} f_{\mathcal{E}}(x)/2$.*

(iii) *If x is an unweighted vertex of type $i \in \{1, 2\}$ and p is of type $j > i$, then $r_x \geq r_p/\sqrt{2}$.*

(iv) *If x is of type 3, then $r_x > \bar{\rho} r_p$ or $r_x > \lambda(x)$.*

Proof. If x is a type 0 vertex or a weighted vertex, $r_x \geq f_{\mathcal{E}}(x)$ because at the end of the stage Protect, r_x is the Euclidean distance from x to the nearest weighted vertex, and $f_{\mathcal{E}}(x)$ cannot exceed that value as $\mathcal{E}$ contains every weighted vertex.

If x is of type 3, it is the orthocenter of a tetrahedron τ whose orthoradius-edge ratio $\rho(\tau)$ exceeds $\bar{\rho}$ or whose orthoradius exceeds $\lambda(x)$. The insertion radius of x is the radius $r_x = \sqrt{\pi(x, p)}$ of τ's orthoball because the vertices of τ are orthogonal to the orthoball but, as τ is weighted Delaunay, no mesh vertex is closer than orthogonal to the orthoball. In the case where τ is split because its orthoradius exceeds $\lambda(x)$, clearly $r_x > \lambda(x)$. In the case where $\rho(\tau) > \bar{\rho}$, the parent p of x is the most recently inserted endpoint of τ's shortest edge. As no vertex in $S[\omega]$ is negatively weighted, r_p is no greater than the length ℓ of τ's shortest edge. Therefore, $r_x = \rho(\tau)\ell > \bar{\rho} r_p$.

If x is an unweighted vertex of type $i \in \{1, 2\}$, it is the orthocenter of an encroached subsegment or subpolygon σ. Let X be its orthoradius. If p is of type $j > i$, then p is a simplex orthocenter rejected for encroaching upon σ. Because σ was not encroached before p was rejected, $r_x = X$. The simplex σ contains the orthogonal projection of p onto aff σ by the algorithm's design. Therefore, σ has a vertex a such that $\angle pxa \leq 90°$, which implies that

$$d(p, a)^2 \;\leq\; d(p, x)^2 + d(x, a)^2 \;\leq\; X^2 + d(x, a)^2.$$

It follows that

$$r_p^2 \;\leq\; \pi(p, a) \;=\; d(p, a)^2 - \omega_a \;\leq\; X^2 + d(x, a)^2 - \omega_a \;=\; X^2 + \pi(x, a) = 2X^2 \;=\; 2r_x^2,$$

so $r_x \geq r_p/\sqrt{2}$ as claimed.

If x is an unweighted vertex of type $i \in \{1, 2\}$ and p is of type $j \leq i$, then let h be the linear cell of least dimension in $\mathcal{P}$ that includes the simplex σ split by x, and let h' be the linear cell of least dimension in $\mathcal{P}$ that contains p. Whether or not h and h' are disjoint, their counterparts $h \setminus U, h' \setminus U \in \mathcal{E}$ are disjoint, where U is the union of the interiors of the protecting balls. To see this, observe that if h' adjoins h, the encroachment implies that together they do not satisfy the projection condition, so $h \cap h'$ is covered by U.

Unweighted vertices are never inserted in protecting balls, so $x \in h \setminus U$, and if p is unweighted, $p \in h' \setminus U$. If p is weighted, it is a vertex in $\mathcal{E}$ with a protecting ball whose radius does not exceed half the local gap size, so $\sqrt{\omega_p} \leq g(p)/2 \leq d(x,p)/2$. In either case, by the definition of local feature size, $f_{\mathcal{E}}(x) \leq d(x,p)$, so

$$r_x^2 \;=\; \pi(x,p) \;=\; d(x,p)^2 - \omega_p \;\geq\; d(x,p)^2 - \left(\frac{d(x,p)}{2}\right)^2 \;\geq\; \frac{3}{4} f_{\mathcal{E}}(x)^2,$$

thus $r_x \geq \sqrt{3} f_{\mathcal{E}}(x)/2$ as claimed. □

We use Proposition 9.3 to show that the algorithm generates graded meshes with strong lower bounds on the edge lengths. We wish to incorporate the effect of the size field in addition to the local feature size. Define the field

$$\mu(x) = \min\left\{ f_{\mathcal{E}}(x),\, \inf_{y \in |\mathcal{P}|} (C_3 \lambda(y) + d(x,y)) \right\},$$

where $C_3 = (\bar{\rho} + \sqrt{2} + 1)/(\bar{\rho} - 2)$ (recall Proposition 8.5). Observe that the second expression in the braces is 1-Lipschitz—by design, it is the largest 1-Lipschitz function that is nowhere greater than $C_3\lambda$. The field μ is 1-Lipschitz because it is a minimum of two 1-Lipschitz functions. It captures the combined influence of the local feature size and the size field on the edge lengths in the mesh.

Theorem 9.4. *Let $\mathcal{P}$ be a PLC embedded in $\mathbb{R}^3$. If $\bar{\rho} \geq 2$ and $\inf_{x \in |\mathcal{P}|} \lambda(x) > 0$, then* DelTetAcutePLC$(\mathcal{P}, \bar{\rho})$ *terminates and returns a Steiner weighted Delaunay triangulation of $\mathcal{P}$ in which no tetrahedron has an orthoradius-edge ratio greater than $\bar{\rho}$. Therefore, tetrahedra with no weighted vertices have circumradius-edge ratios no greater than $\bar{\rho}$. For any two vertices p and q in the mesh, $d(p,q) \geq \mu(p)/(C_1 + 1)$ where $C_1 = (3 + \sqrt{2})\bar{\rho}/(\bar{\rho} - 2)$.*

Proof. By Proposition 9.2, the stage Protect terminates.

The inequalities stated in Proposition 9.3 are nearly the same as in Proposition 8.4, so we can reprise the proofs of Propositions 8.5 and 8.6 with μ replacing the local feature size f. Proposition 8.5 states that the insertion radius r_x of every vertex x of type i satisfies $r_x \geq \mu(x)/C_i$, where C_0, C_1, C_2, and C_3 are specified in the statement of Proposition 8.5. There are two changes to the algorithm for which we must verify that these invariants still hold. First, if a vertex x is inserted at the orthocenter of a tetrahedron because its orthoradius exceeds $\lambda(x)$, then $r_x > \lambda(x) \geq \mu(x)/C_3$ as stated. Second, the inequality $r_x \geq \sqrt{3} f_{\mathcal{E}}(x)/2$ from Proposition 9.3 is looser than its counterpart from Proposition 8.4, but it is tight enough to imply that $r_x > \mu(x)/C_2 > \mu(x)/C_1$.

By Proposition 8.6, for any two vertices in the mesh, $d(p,q) \geq \mu(p)/(C_1 + 1)$. By the Packing Lemma (Lemma 6.1), Refine terminates.

When DelTetAcutePLC terminates, no subsegment or subpolygon is encroached, hence $\{\sigma \in \mathrm{Del}\, S\,[\omega] : \sigma \subseteq |\mathcal{P}|\}$ is a Steiner triangulation of $\mathcal{P}$. Moreover, because DelTetAcutePLC terminates, no tetrahedron's orthoradius-edge ratio exceeds $\bar{\rho}$. □

The merit of the eunuch structure $\mathcal{E}$ is that it gives us intuition about how small angles affect the edge lengths in the mesh. Theorem 9.4 shows that DelTetAcutePLC returns a

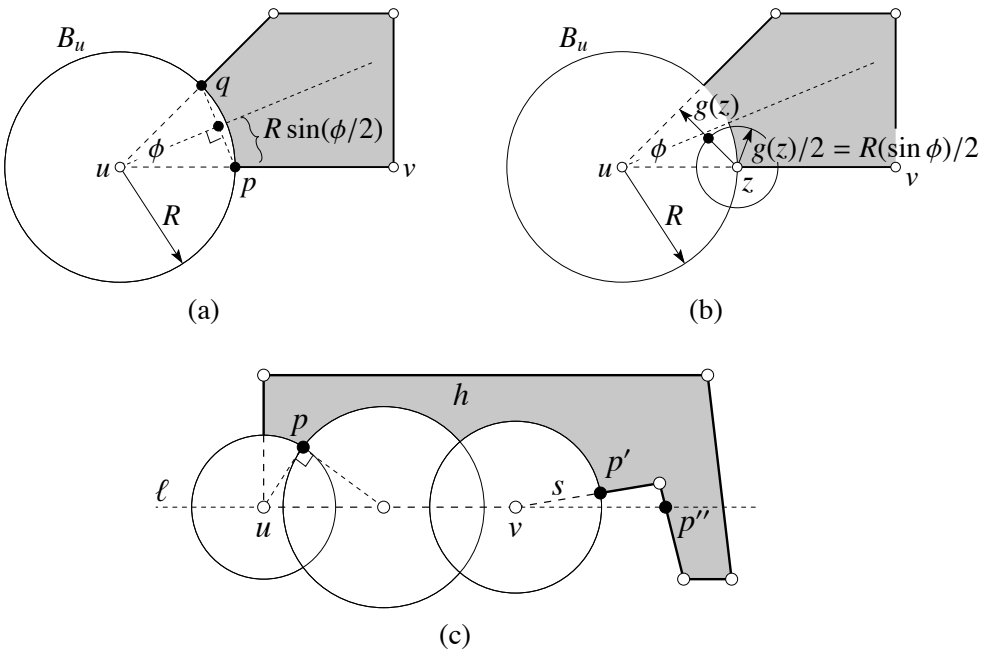


Figure 9.3: (a) The complex $\mathcal{E}$, shown as bold edges, has its minimum local feature size of $R\sin(\phi/2)$ at the midpoint of pq. (b) A protecting ball centered at z can have radius as great as half the gap size, as shown, or as small as nearly seven times smaller. (c) The polygon h meets another polygon h' (not shown) at a small dihedral angle along an edge uv. After the protecting balls are removed, the distance between the polygons depends partly on how close they are to the meeting line ℓ. A point p whose projection onto ℓ lies on $h \cap h'$ is kept at a distance by the orthogonal protecting balls. A point whose projection does not lie on $h \cap h'$, like p' and p'', can lie much closer to ℓ or even on ℓ.

mesh with edge lengths not much smaller than the size field λ or the local feature size $f_{\mathcal{E}}$ of the eunuch structure, whichever is smaller. There are two reasons why $f_{\mathcal{E}}(x)$ could be smaller than $f(x)$ at a point x: there are two adjoining linear cells in $\mathcal{P}$ whose counterparts in $\mathcal{E}$ are disjoint and whose distance from x is less than $f(x)$; or a new vertex was added to $\mathcal{E}$ at a protecting ball center near x.

Let us examine how small $f_{\mathcal{E}}$ can be in terms of the angles in $\mathcal{P}$. Recall from Chapter 8 that there are several notions of the angle at which two linear cells adjoin each other. For simplicity, we use the linear cells' affine hulls as proxies: if two segments or polygons (possibly one of each) h and h' adjoin each other and fail the projection condition, we take the smallest angle at which aff h meets aff h', and we say that h and h' meet at that angle.

Every acute vertex u is protected by a ball of radius $R = \min\{\lambda(u), f(u)/(2\sqrt{2})\}$. Consider a segment $uv \in \mathcal{P}$ and another segment or polygon $h \in \mathcal{P}$ that meet at u at an angle of ϕ. Their counterparts in $\mathcal{E}$, namely, $uv \setminus \operatorname{Int} B(u, R)$ and $h \setminus \operatorname{Int} B(u, R)$, can jointly generate a local feature size $f_{\mathcal{E}}$ as small as $R\sin(\phi/2)$, but not smaller (ignoring contributions from other linear cells). As Figure 9.3(a) shows, this value is achieved at the midpoint of a line segment pq where $p = uv \cap \operatorname{Bd} B(u, R)$ and $q \in (\operatorname{aff} h) \cap \operatorname{Bd} B(u, R)$ is chosen so that $\angle qup = \phi$.

The success of Protect depends on the local gap size having a positive lower bound wherever protecting balls can be centered. If ϕ is the smallest angle at which an edge uv meets any other edge or polygon, then the local gap size $g(z)$ for a point z in the interior of uv is at least $\min\{f(z), d(u,z)\sin\phi, d(v,z)\sin\phi\}$; see Figure 9.3(b). Recall from Proposition 9.2 that every protecting ball centered on uv has radius greater than $\inf_{x\in(uv\setminus(B_u\cup B_v))\cup\{u,v\}} \min\{\lambda(x)/7, g(x)/14\} \geq \inf_{x\in(uv\setminus(B_u\cup B_v))\cup\{u,v\}} \min\{\lambda(x)/7, f(x)/14, d(u,x)\sin\phi/14, d(v,x)\sin\phi/14\}$. As the protecting balls at u and v do not contain the centers of the other protecting ball centered on uv, both $d(u,x)$ and $d(v,x)$ are greater than the radius R of the smaller of the protecting balls centered at u and v. Thus, the other balls centered on uv have radii at least $\inf_{x\in uv} \min\{\lambda(x)/7, f(x)/14, R(\sin\phi)/14\}$. A very small angle ϕ can yield very small protecting balls and a proportionally small local feature size $f_{\mathcal{E}}$ near the protecting balls.

Consider two polygons h and h' that meet at a small dihedral angle θ along a shared edge uv (or perhaps at a single vertex). How close can their counterparts in $\mathcal{E}$ be to each other? Let U be the union of the interiors of the protecting balls placed on $h \cap h'$. Let p be a point in $h \setminus U$ and q be a point in $h' \setminus U$ such that $d(p,q)$ is globally minimized. There are two cases. If the projection of p onto the line $\ell = \operatorname{aff} h \cap \operatorname{aff} h'$ lies on $h \cap h'$, as at left in Figure 9.3(c), then p cannot be closer to the line than $R'/\sqrt{2}$, where R' is the radius of the smallest protecting ball on uv. This follows because consecutive balls are orthogonal and the distance is minimized where the boundaries of two radius-R' balls meet. In this case, $d(p,q)$ can be as small as $\sqrt{2}R'\sin(\theta/2)$ if p and q are on the boundaries of the same two protecting balls, forcing the local feature size $f_{\mathcal{E}}$ to be as small as $R'\sin(\theta/2)/\sqrt{2}$ at the midpoint of pq. (Recall Figure 9.3(a), but imagine a dihedral angle.)

If p's projection does not lie on $h \cap h'$, as at the points p' and p'' in Figure 9.3(c), it is possible for $d(p,q)$ to be much smaller. Because $d(p,q)$ is minimized, at least one of p or q lies on an edge of its polygon; suppose p lies on an edge s of h. If s adjoins h' (see p' in the figure), let $\phi \leq \theta$ be the an angle at which they meet, let R be the radius of the protecting ball at the vertex where s and h' meet, and recall that their counterparts in $\mathcal{E}$ can be as close as $2R\sin(\phi/2)$; this is the minimum for $h \setminus U$ and $h' \setminus U$ as well. Therefore, they can generate a local feature size $f_{\mathcal{E}}$ as small as $R\sin(\phi/2)$, but not smaller.

Conversely, if s and h' are disjoint (see p'' in the figure), it is possible that h and h' make $f_{\mathcal{E}}$ smaller than f at some points, but h and h' cannot make $\min_{x\in\mathbb{R}^3} f_{\mathcal{E}}(x)$ smaller than $\min_{x\in\mathbb{R}^3} f(x)$ because s and h' are disjoint, and the distance between h and h' is minimized by $p \in s$ and $q \in h'$.

The worst circumstance is when an edge uv participates in both a very small plane angle ϕ that induces a tiny local gap size and forces Protect to place tiny protecting balls on uv, and a very small dihedral angle θ that forces Refine to produce tinier edges just outside the protecting balls. The lengths of these edges are $\Theta(\phi\theta)$ as ϕ and θ approach zero.

9.5 Notes and exercises

If a PLC in $\mathbb{R}^3$ has segments or polygons that adjoin each other at small angles, it can be quite difficult to triangulate it with Delaunay simplices, even if there is no constraint on the quality of the tetrahedra. Most algorithms for generating Steiner Delaunay triangulations

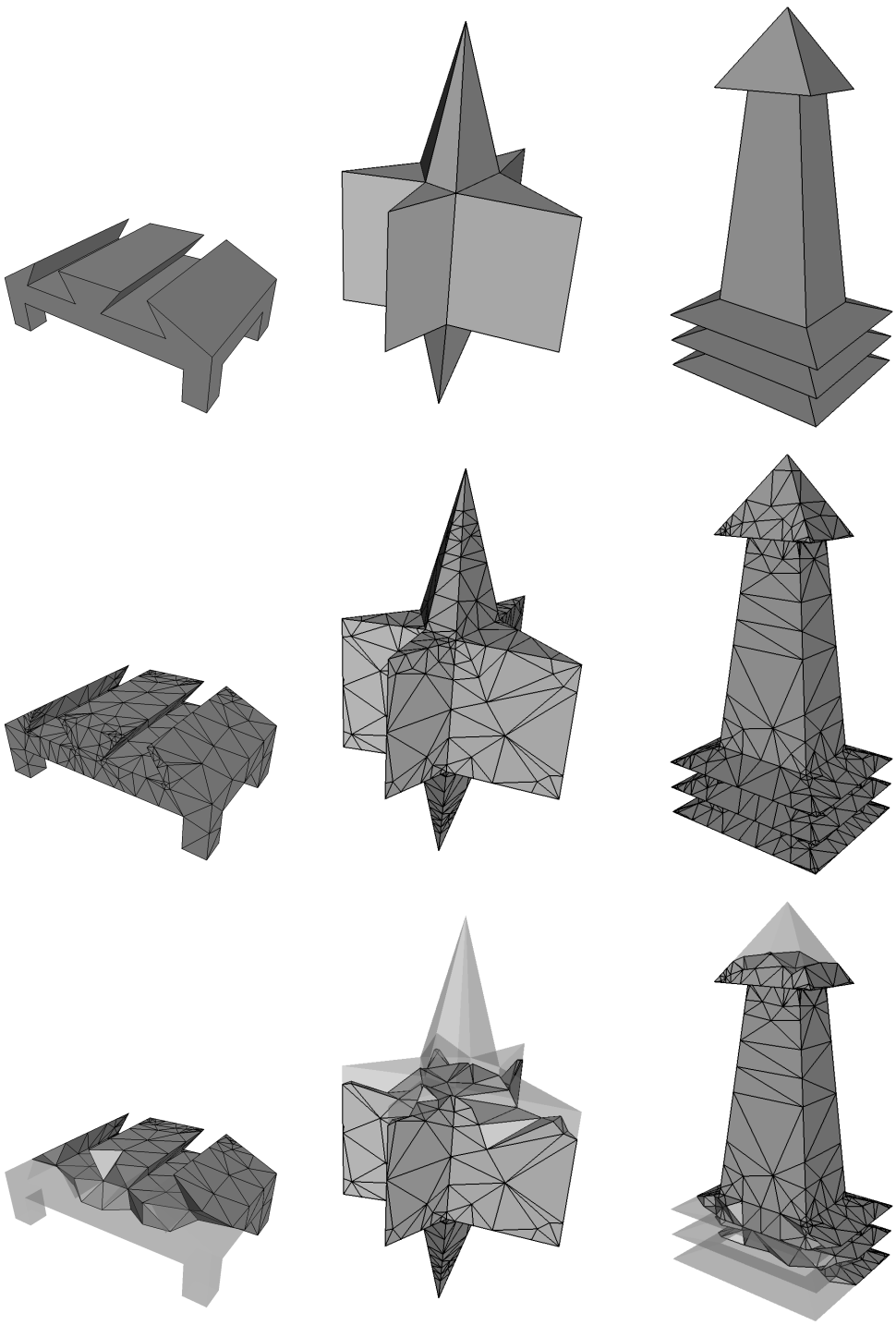


Figure 9.4: Three polyhedra that have small dihedral angles, and their tetrahedral meshes generated by a precursor of the algorithm described here.

of three-dimensional PLCs use the idea to protect PLC vertices and segments with balls inside which vertices may not be inserted. These algorithms do not use weighted vertices; rather, they prevent the insertions of vertices into the protecting balls by placing vertices on the boundary of the union of the protecting balls. Different protection methods have been proposed by Murphy, Mount, and Gable [156], Cohen-Steiner, Colin de Verdière, and Yvinec [65], Cheng and Poon [57], Cheng, Dey, Ramos, and Ray [52], and Pav and Walkington [165].

Murphy et al. [156] protect PLC vertices with balls whose radii are uniform—a fraction of the minimum local feature size among the vertices—and protect the segments with cylinders whose radii are uniform—a fraction of the minimum local gap size among the segments. The balls and cylinders are protected by vertices placed at points where PLC segments and polygons meet ball boundaries and cylinder boundaries. Finally, they triangulate the polygons by calling Chew's algorithm, described in Section 1.2. The triangles have uniform sizes, and their circumradii are a fraction of the minimum local gap size among the polygons. There are no restrictions on the shapes of the tetrahedra generated. This algorithm has the distinction of being the first proof that every three-dimensional PLC has a Steiner Delaunay triangulation, but it generates far more vertices than necessary.

Cohen-Steiner et al. [65] protect PLC vertices with balls whose radii are proportional to the local feature size and protect the segments with balls whose radii are proportional to the local gap size—much as we do in this chapter, except that two adjacent protecting balls may have a very small overlap, whereas the algorithm in this chapter places them so they are orthogonal. The balls are protected by vertices placed where ball boundaries intersect PLC polygons, as determined by special encroachment rules. These rules also recover the polygons. The algorithm produces graded triangulations in practice, but unduly short edges can appear.

The two algorithms above do not attempt to control the quality of the tetrahedra. The first Delaunay refinement algorithm to claim some theoretical guarantees on tetrahedron quality for domains with small angles is by Shewchuk [199]. It uses constrained Delaunay triangulations, the CDT Theorem (Theorem 4.9), and concentric spherical shell segment splitting to guarantee domain conformity. The final mesh is a graded Steiner CDT of the input PLC. The paper proves that for some PLCs, no algorithm can fix every skinny element, or even every skinny element that does not immediately adjoin a small domain angle: inherently, part of the problem of meshing domains with small angles is to decide when and where to leave skinny elements alone. The algorithm decides which skinny tetrahedra not to try to split by explicitly computing insertion radii and declining some of the vertex insertions that violate the consequences of Proposition 8.4, even if it means leaving a skinny tetrahedron intact.

Cheng and Poon [57] describe an algorithm that produces well-graded, high-quality Steiner Delaunay triangulations without the need for CDTs. They place protecting balls of graded sizes on vertices and segments, using a precursor of the scheme described in this chapter; then they run Delaunay refinement outside the protecting balls while placing vertices on the protecting ball boundaries according to encroachment rules. The union of the protecting balls is filled with Delaunay tetrahedra that are compatible with the triangulation outside the union. The algorithm guarantees an upper bound on the aspect ratios of the tetrahedra inside the protecting balls, ruling out the worst slivers. The bound depends on

the domain angles, and degrades as the domain angles do. All the other tetrahedra in the domain have bounded radius-edge ratios.

The Delaunay meshing algorithm of Cheng, Dey, Ramos, and Ray [52] also protects vertices and segments with balls of graded sizes. The radii of the protecting balls centered at input vertices are chosen by computing local feature sizes, but the protecting balls on acute segments are chosen and refined adaptively by a specialized Delaunay refinement method. Once the algorithm has computed a Steiner Delaunay triangulation, the protecting balls are frozen. Then skinny tetrahedra are refined as described in Chapter 8, but tetrahedron circumcenters that fall in protecting balls are discarded. The algorithm is guaranteed to work only for polyhedra, i.e. domains in which there are no internal boundaries and no segment is a face of more than two polygons. This restriction makes it easier to produce a conforming Delaunay triangulation and obviates the need to compute ball-polygon intersections. The meshes in Figure 9.4 are generated by this algorithm.

Very shortly thereafter, Pav and Walkington [165] proposed a similar meshing algorithm that generates graded Steiner Delaunay triangulations of general PLCs. An advantage of the algorithm is related to local feature size computations: the algorithms of Cohen-Steiner et al. [65] and Cheng and Poon [57] require expensive explicit computations of local feature sizes at protecting ball centers; Cheng et al. [52] require them only at PLC vertices; Pav and Walkington [165] (and Shewchuk [199]) infer the local feature sizes inexpensively from the process of meshing itself, as do the algorithms DelTriPLC and DelTetPLC.

The constrained Delaunay refinement algorithm of Si [206] copes with small domain angles by maintaining a CDT and declining to insert a new vertex v if its insertion will create an edge shorter than $b \cdot \lambda(v)$, where b is a user-specified constant and λ is the user-specified size field. The algorithm takes advantage of the CDT Theorem (Theorem 4.9) to guarantee that a CDT exists; hence, some vertices may be inserted on segments despite creating short edges.

The new algorithm presented in this chapter adapts the segment protection scheme of Cheng and Poon [57], but it uses a Steiner weighted Delaunay triangulation both inside and outside the protecting balls. The ideas to turn protecting balls into weighted points and to refine by inserting orthocenters come from Cheng, Dey, and Ramos [51], whose algorithm for meshing piecewise smooth complexes appears in Chapter 15.

Exercises

1. Let s be a linear cell in a PLC, and let $\mathring{s}$ be the union of its proper faces in the PLC. Prove that the local gap size function g is 1-Lipschitz on the domain $s \setminus \mathring{s}$; that is, for any two points x and y in $s \setminus \mathring{s}$, $g(x) \leq g(y) + d(x, y)$.

2. (a) Derive the expressions in Step 1 of Cover for the position and radius of a ball orthogonal to two other balls with collinear centers.

   (b) Step 3 of Cover can be made more aggressive. Suppose that we modify it to compute three new balls having equal radii that fill the gap between B_x and B_y tightly, so that in the sequence of five balls from B_x to B_y, each consecutive pair of balls is orthogonal. Then the center ball is shrunk if necessary to accommo-

date the size field and the local gap size, and the corresponding weighted point is inserted into $S[\omega]$.

Derive expressions for the radius and positions of the three balls. They need not be closed-form expressions, as the radius is the root of a quartic polynomial.

3. Let pqr be a triangle in $\mathbb{R}^2$. Suppose that we protect the segments of pqr as described in this chapter. Prove that for every protecting ball center x, $g(x) = \phi \cdot \Theta(f(x))$, where ϕ is the smallest angle of the triangle. Can you extend your proof to a polygon with polygonal holes?

4. Rather than have separate protection and refinement stages, we could change the algorithm so that it protects segments lazily during refinement, deciding whether to place a protecting ball on a segment only when one of its subsegments is encroached.

   (a) Write pseudocode for this algorithm.

   (b) When your algorithm decides to place a new protecting ball, can this ball contain an existing vertex? Why or why not?

   (c) Describe an advantage of the modified algorithm.

   (d) Prove that your algorithm produces a Steiner triangulation of the input PLC.

5. Derive formulae for computing the coordinates of the orthocenter of a triangle, given the coordinates of its vertices.

6. DelTetAcutePLC does not protect nonacute segments. The advantage is that every mesh tetrahedron that does not adjoin an acute segment or vertex is guaranteed to have a good radius-edge ratio. However, if a Delaunay refinement algorithm protects every segment, acute and nonacute, then $\bar{\rho}$ can be reduced to less than 2, and Refine can still be guaranteed to terminate and work correctly. For this modified algorithm, what is the best upper bound $\bar{\rho}$ on the orthoradius-edge ratio that can (without extraordinary efforts) be guaranteed with the same proof techniques employed in this chapter? Explain your answer.