Vertex-rounding a three-dimensional polyhedral subdivision

Steven Fortune
Bell Laboratories
Murray Hill, New Jersey 07974
sjf@research.bell-labs.com

Abstract

Let P be a polyhedral subdivision in $\mathbb{R}^3$ with a total of n faces. We show that there is an embedding σ of the vertices, edges, and facets of P into a subdivision Q, where every vertex coordinate of Q is an integral multiple of $2^{-\lceil \log_2 n+2 \rceil}$. For each face f of P, the Hausdorff distance in the L_∞ metric between f and $\sigma(f)$ is at most 3/2. The embedding σ preserves or collapses vertical order on faces of P. The subdivision Q has $O(n^4)$ vertices in the worst case, and can be computed in the same time.

1 Introduction

Geometric algorithms are usually described in the "real-number RAM" model of computation, where arithmetic operations on real numbers have unit cost. A programmer implementing a geometric algorithm must find some substitution for real arithmetic. The substitution of exact arithmetic on a subset of the reals, say the integers or the rationals, avoids the difficulties that can arise from naive substitution of floating-point arithmetic [4, 12, 14, 15]. The substitution is not trivial, since the required arithmetic bit-length usually exceeds the native arithmetic bit-length of most computer hardware, and some form of software arithmetic is required.

Recent research has made the use of software exact arithmetic for geometric algorithms much more attractive. A predicate on geometric data is determined by the sign of an arithmetic expression in the coordinates of the data. A promising strategy for sign-evaluation is adaptive-precision arithmetic [6, 13, 20], where the expression is evaluated to higher and higher precision until its sign is known, i.e. until the magnitude of the expression exceeds an error bound. Low precision, even floating-point, suffices most of the time, since most instances of geometric predicates are easy. In addition, for some basic predicates like the sign of a determinant, there are alternative evaluation strategies that require arithmetic with relatively low precision [1, 2, 3].

Exact arithmetic would be more useful if high-level geometric rounding algorithms were available. Virtually any geometric construction that produces new geometric data increases the bit-length of geometric coordinates. For example, suppose points are represented with homogeneous integer coordinates. The plane through three such points has coefficients whose bit-lengths are about three times the point coordinate bit-lengths; the point of intersection of three such planes has coordinate bit-length about nine times that of the original points. Thus a solid modeler, which implements boolean operations and rigid motions on polyhedra, might produce a polyhedron with high coordinate bit-length even if the original polyhedra had short coordinate bit-length. Typically an application requires only a low-precision approximation, not the exact answer. Hence there is a need for high-level rounding, which replaces a geometric structure with high bit-length coordinates with an approximating structure with short bit-length coordinates. It does not suffice to round each coordinate independently, since such rounding is a geometric perturbation, and may introduce inconsistencies between geometric and combinatorial information. Furthermore, some change in combinatorial structure is inevitable; indeed, in certain cases it is NP-hard to determine if it is possible to round to low-precision without changing combinatorial structure [19].

There are satisfactory high-level rounding algorithms for polygonal subdivisions in two dimensions. One such algorithm is snap-rounding [10]. Fix a polygonal subdivision, with arbitrary-precision coordinates. A *pixel* is a unit square in the plane centered at a point with integer coordinates; a pixel is *hot* if it contains a vertex of the subdivision. Snap-rounding replaces each vertex by the center of the pixel containing the vertex, and each edge by the polygonal chain through the centers of the hot pixels met by the edge, in the same order as met

SCG 98 Minneapolis Minnesota USA

by the edge. The snap-rounded subdivision approximates the original subdivision in the sense that each vertex and edge of the original subdivision has an image in the snap-rounded arrangement whose Hausdorff distance is at most 1/2 in the L_∞ metric. Snap-rounding may change the combinatorial structure of the subdivision, for example, vertices and edges may collapse together, but some combinatorial ordering information is preserved [10].

This paper generalizes snap-rounding to polyhedral subdivisions in three dimensions. Fix a polyhedral subdivision P with a total of n vertices, edges, and facets. We show that there is a polyhedral subdivision Q so that each vertex coordinate is an integer multiple of $1/2^{\lceil \log_2 n \rceil + 2}$. Each face f of P has an image $\sigma(f)$ in Q so that the Hausdorff distance between f and $\sigma(f)$ is at most 3/2. As with snap-rounding in two dimensions, f and $\sigma(f)$ may have different combinatorial structure: an edge may be replaced with a polygonal chain, and a facet with a triangulation. Two vertices may collapse together; the polygonal chains for two edges or the triangulations for two facets may collapse together or overlap partially, perhaps in several places. However, vertical order is preserved (or collapsed): if face f is vertically above face f' (i.e. there is a line parallel to the z-axis meeting both faces, and the intersection with f has higher z-coordinate), then $\sigma(f)$ is above (or overlaps) $\sigma(f')$. In the worst case Q has $O(n^4)$ vertices and can be computed in time $O(n^4)$.

Other work. Greene and Yao were the first to suggest a rounding scheme for polygonal subdivisions in two dimensions [8]. Hobby [11] and Greene [9] give algorithms to compute the snap-rounding of the arrangement formed by a set of intersecting edges. Guibas and Marimount [10] show how to maintain the snap-rounded arrangement of edges under insertion and deletion of edges; they also give elementary proofs of basic topological properties of snap rounding. Goodrich *et al* [7] give improved algorithms to snap-round a set of intersecting edges, in the case when there are many intersections within a pixel. Milenkovic [18] suggests a "shortest-path" geometric rounding scheme that sometimes introduces fewer bends than snap rounding.

Goodrich *et al* [7] propose a scheme for snap-rounding a set of edges in three dimensions after first adding as vertices the points of "closest encounter" between nearby edges. Milenkovic [16] sketches a scheme for rounding a polyhedral subdivision in three dimensions (in fact, any dimension). Unfortunately, both schemes have the property that rounded edges can cross (see below), which violates any notion of topological consistency.

Fortune [5] suggests a high-level rounding algorithm for polyhedra in three dimensions. His algorithm assumes that a polyhedron is presented by the equations of its face planes (and the combinatorial incidence structure of faces), not the coordinates of vertices as assumed by snap-rounding. His algorithm does not appear to extend from polyhedra to polyhedral subdivisions.

1.1 Overview

We give a brief overview of the rounding algorithm. We start by mentioning some difficulties with the three-dimensional extension of snap-rounding.

The obvious way to snap-round a vertex in three dimensions is to replace it with the center of the voxel containing it. (A *voxel* is a unit cube centered at an integer point.) However, it is less clear how to snap-round edges and facets.

Snap-rounding a set of edges in three-dimensions requires the addition of new vertices, unlike the situation in two dimensions. Consider two transverse nearby edges. Rounding the endpoints to voxel centers perturbs the edges, and hence the edges may change orientation or cross. We can attempt to prevent this by adding a vertex in the interior of each edge near the other edge; then either the two new vertices are in the same voxel and snap-round together, or they are in different voxels and the snap-rounded edges will not cross. Clearly, it might be necessary to add quadratically many vertices, if the edges form a "cross-hatch" pattern.

Snap-rounding with facets as well is more problematic. If a vertex v and a facet f are nearby, we can add a new vertex v' to f to ensure that v and f are properly separated or collapsed. However, this requires that f be triangulated, which introduces new edges. Potentially these edges are close to old edges, which could require new vertices, and it is not immediate that the process is finite. We can attempt to ensure termination by projecting nearby edges onto a facet, and then triangulating the facet compatibly with the projection. The actual rounding algorithm is a formalization of this idea.

The rounding algorithm. Orthogonally project all the edges of the subdivision P onto the xy-plane, form the arrangement, snap-round, and compute a triangulation T. Each face of P has an image within the triangulation: the image of an edge is a polygonal chain, and the image of a facet is a subtriangulation of T. The rounding of each facet f is obtained by lifting the image of f in T to three dimensions in such a way as to approximate f. By considering each cylinder over a vertex, edge, or triangle of T separately, we can ensure that the lifting preserves (or collapses) the vertical order on faces of P.

There are several technical difficulties with this algorithm. We must first ensure that there are no crossings among the polygonal chains that result from rounding the edges of P. Figure 1 indicates one way such a crossing could occur. To prevent crossings, we subdivide the edges of P by all $xy-$, $xz-$, and yz-intersection points. (If the orthogonal projections of e and e' into the xz-plane cross at a point p, and l is the line parallel to the y-axis through p, then $e \cap l$ and $e' \cap l$ are ***xz*-intersection points**.) Unfortunately, this subdivision is not quite suf-

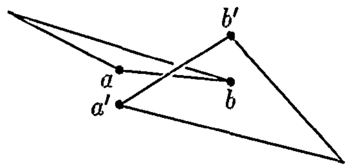

Figure 1: Vertices a and a' project to the same pixel in the xy-plane, as do b and b'. Hence in three dimensions, the snap-rounding of ab crosses the snap-rounding of $a'b'$.

ficient to prevent crossings among snap-rounded edges. In figure 2, d^* and e^* have endpoints on column facets. The xy-, xz-, and yz-projections of d^* and e^* are all disjoint, but their snap-roundings cross. Fortunately, the configuration of figure 2 is almost the only way this can happen, and we can show that there is a slight modification of snap-rounding that avoids crossings. In figure 2, the modified snap-rounding of d^* is a two-edge polygonal chain, connecting a snap-rounded endpoint of d^* to the snap-rounded endpoint of e^* on the same vertical line, and then to the other snap-rounded endpoint of d^*. We show that the distance between an edge and its modified snap-rounding increases slightly, to at most 3/2. (The configuration in figure 2 can be modified to show that the "close encounter" subdivision of Goodrich *et al* [7] does not prevent edge crossings.) Section 4 describes the modification of snap-rounding.

Let T_f be the image of facet f within triangulation T. We lift T_f to three dimensions by first lifting each edge and then the interior of each triangle. The lifting of an edge must satisfy three conditions: it must be close to f; it must not cross any other edge; and it must preserve vertical order. This last condition is a bit tricky. Consider the situation given in cross-section in figure 3, with e, e', and f orthogonal to the figure. Assume that edges e and e' project and snap-round to the same edge d in the triangulation T. Edge e is above facet f, so the lifting of edge d for facet f must be on

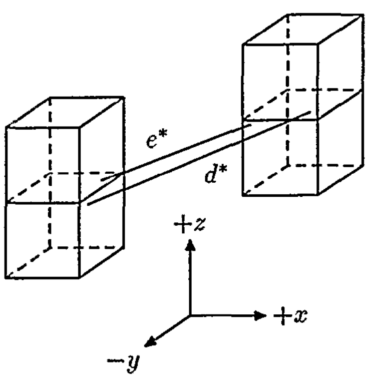

Figure 2: $\rho(d^*)$ and $\rho(e^*)$ cross, although the xy-, xz-, and yz-projections of d^* and e^* do not.

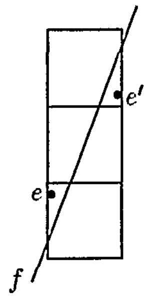

Figure 3: Side view. Edge e is above facet f and e' is below. Hence the rounding of facet f must contain the vertical interval from the rounding of e to the rounding of e'.

or below the rounding of e. Similarly, edge e' is below facet f, so the lifting of edge d for facet f must on or above the rounding of e. Hence the lifting of d for facet f must at least span the vertical interval between the rounding of e and the rounding of e'. Section 5 below describes how to lift triangulation edges.

Consider a triangle $\Delta = \Delta abc$ of T_f. If the lifted edges ab, ac, and bc for facet f are all pairwise incident, then the lifting of Δ for facet f, $l_f(\Delta)$, is simply the triangle with those edges. Unfortunately, incidence cannot be guaranteed (though, of course, both lifted edges ab and ac meet the vertical line through a, and similarly for b and c). Hence $l_f(\Delta)$ must be a triangulation of the polygon formed by the lifted edges ab, ac, and bc, and perhaps edges along the vertical lines through a, b, and c. See figure 4. It is easy to triangulate the polygon using a central vertex whose xy-projection is within triangle Δ. However, a vertical boundary edge may be shared among several different liftings. To ensure that there are no crossings among edges, each central vertex must have distinct coordinates. Since there may be n central vertices, coordinates that are integer multiples of roughly $1/n$ are necessary. More details of the lifting appear in section 6.

Naively the rounded subdivision Q has at most $O(n^3)$ faces: the triangulation T has $O(n^2)$ triangles, so for each facet f the rounding $\sigma(f)$ consists of $O(n^2)$ lifted triangles $\{l_f(\Delta)\}$. However, in the worst case each lifted triangle $l_f(\Delta)$ may consist of $O(n)$ faces, since there could be linearly many vertices on the vertical edges of its boundary. Hence Q has $O(n^4)$ faces.

2 The main theorem

For points $a, b \in \mathbb{R}^3$ and sets $A, B \subset \mathbb{R}^3$, $d(a,b)$ is the L_∞ distance between a and b (the L_∞ distance is used exclusively in this paper); $d(a,B)$ is $\inf_{b \in B} d(a,b)$; and $d(A,B)$ is $\sup_{a \in A} d(a,B)$. Note that d is symmetric for points, but not in general for sets. *Hausdorff distance* $d_H(A,B)$ is $\max(d(A,B), d(B,A))$.

The direction parallel to the z-axis is the *vertical*

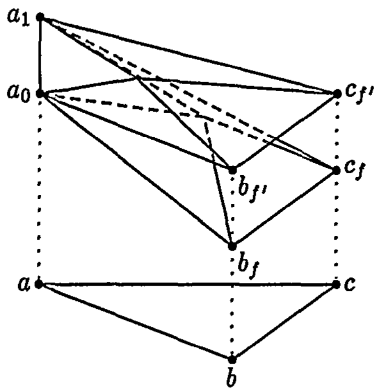

Figure 4: The liftings of triangle Δabc for facets f and f' have boundary $a_1 a_0 b_f c_f$ and $a_1 a_0 b_{f'} c_{f'}$, respectively.

direction. Two sets $A, B \subset \mathbb{R}^3$ are *vertically ordered* $A \prec B$ (read "A is below B") if there is a vertical line meeting both A and B, and for every vertical line l meeting A and B, $A \cap l$ is below $B \cap l$, i.e. the z-coordinate of every point of $A \cap l$ is less than the z-coordinate of every point in $B \cap l$. Sets A and B satisfy $A \preceq B$ if there is a vertical line meeting both, and for every vertical line meeting both, $A \cap l$ is below or intersects $B \cap l$. As is well-known, $\prec$ is not transitive in general; it is transitive among a family of sets that have the same xy-projection. If in addition, every set in the family is a *surface*, i.e. every vertical line misses the set or meets it at one point, then $\preceq$ is also transitive.

A *subdivision* P in $\mathbb{R}^3$ is a set of compact convex polyhedral *cells* so that every face of every cell is in the subdivision and so that the intersection of two cells is a face of both. Cells of dimension 0, 1, and 2 are *vertices*, *edges*, and *facets*, respectively. $|P|$ is the union of the cells of P. An *embedding* of a subdivision P into a subdivision Q is a mapping σ that maps each cell of P into a subdivision contained in Q so that if f is a face of f', then $\sigma(f) \subseteq \sigma(f')$.

To simplify notation somewhat, we extend d and $\prec$ to subdivisions. Thus for subdivisions P and Q, $P \prec Q$ means $|P| \prec |Q|$ and $d(P,Q)$ means $d(|P|,|Q|)$.

Throughout this paper we assume that subdivisions in $\mathbb{R}^3$ do not include cells of dimension 3. Furthermore, we assume that every subdivision is in general position, specifically, that no edge or facet is parallel to a coordinate axis and that no vertex has a coordinate that is an integer multiple of 1/2. The general position assumption simplifies presentation; it is not hard to remove (either explicitly or for example by an infinitesimal symbolic rigid motion).

Theorem 2.1 *Let P be a subdivision in $\mathbb{R}^3$ with a total of n cells; set $\kappa = 3/2$. There is a subdivision Q and an embedding σ of P into Q so that:*

1. *For each cell f of P, $d_H(f, \sigma(f)) < \kappa$.*
2. *Each vertex coordinate of Q is an integral multiple of $1/2^{\lceil 2+\log_2 n \rceil}$.*
3. *If cells f, f' of P satisfy $f \preceq f'$, then $\sigma(f) \preceq \sigma(f')$.*
4. *Q can be computed in time $O(n^4)$ and has $O(n^4)$ cells.*

This theorem follows from the discussion below. At a high level, the algorithm required for step (4) has three steps.

1. Subdivide the vertices and edges of P, forming a set of vertices and edges P^* (Section 4).
2. Orthogonally project P^* onto the xy-plane, snap-round, and triangulate the convex hull of the resulting subdivision. Let T be the resulting triangulation.
3. For each cell f in P, lift T_f (the image of f in T) to a subdivision $Q_f \subset \mathbb{R}^3$ (Sections 5 and 6).

3 Definitions

A *pixel* is an open unit square in the xy-plane centered at an integer point; pixel(q) is the pixel containing point q. A *voxel* is an open unit cube in $\mathbb{R}^3$ centered at an integer point; voxel(q) is the voxel containing point q. A *column* (of voxels) is all voxels whose centers have the same x- and y-coordinates; column(q) is the column containing q.

Let A be a subdivision in the xy-plane. A pixel is *hot* (with respect to A) if it contains a vertex. The *snap-rounding* (with respect to A) of an edge e of A is the polygonal chain connecting the centers of the hot pixels met by e in the same order as met by e; similarly the *snap-rounding* of a vertex of A is the center of the hot pixel containing it. A basic fact[10] is that two polygonal chains that result from snap-rounding intersect only at vertices and edges of both chains. The *snap-rounding* of A is obtained by replacing each edge and vertex of A with its snap-rounding with respect to A; it is a polygonal subdivision whose vertices are hot pixel centers, i.e. integer points, and whose edges connect integer points.

Let π_{xy} be orthogonal projection onto the xy-plane, and similarly for π_{xz} and π_{yz}. A set $A \subset \mathbb{R}^3$ is *over* a set P in the xy-plane if $\pi_{xy}(A) = P$. If A is a surface with $p \in \pi_{xy}(A)$, then A_p is the point of A over p (i.e. $\pi_{xy}(A_p) = p$). If A and B are surfaces over the same set, then $\max(A, B)$ is the pointwise maximum (viewed as functions of the xy-plane), and $\min(A, B)$ is the pointwise minimum. If A, B, C are surfaces over the same set with $A \succeq B$, then $\text{snap}(C, [A, B])$ is $\min(A, \max(B, C))$. Clearly, $A \succeq \text{snap}(C, [A, B]) \succeq B$.

Suppose a set P in the xy-plane is fixed. We define symbolic sets $\top$ (top) and $\bot$ (bottom) satisfying $\bot \prec A \prec \top$ for any other set A over P. We have for example

$\min(A, \top) = A = \max(A, \bot)$; we define min and max of an empty collection to be $\top$ and $\bot$, respectively.

Two edges *cross* if they intersect at a point interior to at least one of the edges.

Proposition 3.1 *Let $T \subset \mathbb{R}^3$ be convex, $\{s_1, \dots, s_k\}$ be a finite set of points in $\mathbb{R}^3$ with convex hull S, and let $\kappa \geq 0$. If $d(s_i, T) \leq \kappa$ for $i = 1, \dots, k$, then $d(S, T) \leq \kappa$.*

Proof: Any point in S can be expressed as $\sum \alpha_i s_i$ with $0 \leq \alpha_i \leq 1$ and $\sum \alpha_i = 1$. For each s_i, there is a point $t_i \in T$ so that $d(s_i, t_i) \leq \kappa$. Clearly $\sum \alpha_i t_i \in T$ and $d(\sum \alpha_i s_i, \sum \alpha_i t_i)$ is the maximum absolute value of any coordinate of $\sum \alpha_i (s_i - t_i)$, which is bounded by κ since $\sum \alpha_i = 1$, $\alpha_i \geq 0$, and the absolute value of each coordinate of $s_i - t_i$ is bounded by κ. □

4 Snap-rounding edges

Define $\rho(q)$ to be the center of the voxel containing q, and extend to ρ to edges: $\rho(qq')$ is the edge $\rho(q)\rho(q')$. The mapping ρ is the obvious extension of snap-rounding to three dimensions (ignoring snapping to hot voxels, which is unimportant here). Unfortunately, ρ may cause two edges to cross. We now define a refinement P^* of the vertices and edges of P and a modification τ of ρ so that no two edges in $\tau(P^*)$ cross.

4.1 The subdivision P^*

Let e and e' be two edges of P whose xy-projections cross at a point p. An *xy-intersection point* (of P) is either point on e or e' that meets the line through p parallel to the z-axis. The definition of an xz– or yz-intersection point is similar.

Subdivision P^* results from subdividing the edges of P. At any point in the process, $\hat{e}$ denotes the subdivision of edge e of P; any voxel containing a vertex is a *hot* voxel; and any column of voxels containing a hot voxel is a *hot* column. There are two steps in the subdivision:

1. Subdivide the edges of P at all xy-, xz-, and yz-intersection points of P.

2. For each edge e of P, split $\hat{e}$ by each hot column C it meets: $\hat{e}$ must meet C in a consecutive set of voxels; $\hat{e}$ is *split* by C by further subdividing $\hat{e}$ at any point in the first voxel (if $\hat{e}$ does not yet have a vertex in the first voxel) and similarly by subdividing $\hat{e}$ in the last voxel.

Splitting by hot columns has an easy consequence: for any edge e of P, the snap-rounding of $\pi_{xy}(\hat{e})$ with respect to $\pi_{xy}(P^*)$ is identical to the snap rounding of $\pi_{xy}(\hat{e})$ with respect to $\hat{e}$. Henceforth we use a superscript '*' for edges and vertices of P^*. For e^* an edge of P^*, we write $s(e^*)$ for the snap-rounding of $\pi_{xy}(e^*)$. It is immediate that if d^*, e^* are edges of P^*, then $\rho(d^*)$ crosses $\rho(e^*)$ only if $s(d^*) = s(e^*)$.

Lemma 4.1 *P^* has $O(n^3)$ vertices; there are $O(n^2)$ hot columns and $O(n^3)$ hot voxels.*

Proof: Clearly there are at most $O(n^2)$ xy-, xz-, and yz-intersection points, and $O(n)$ vertices of P. Splitting edges by hot columns adds no new hot columns, hence there are $O(n^2)$ hot columns. For each edge e of P and for each hot column, there are at most two vertices added when $\hat{e}$ is split by the column. Hence there are $O(n^3)$ vertices altogether. □

As mentioned earlier, T is a triangulation of the convex hull of $s(P^*)$. Consider the edges E^* in P^* bounding a facet f of P. The projection $\pi_{xy}(E^*)$ forms a simple cycle, but the snap-rounding $s(E^*)$ need not. However, it is not hard to see that $s(E^*)$ consists of some number of simple cycles connected by polygonal chains. Let T_f be the subtriangulation of T consisting of the vertices and edges of $s(E^*)$ plus any vertices, edges and triangles of T interior to the simple cycles in $s(E^*)$.

For v a vertex of T, e an edge of T, and Δ a triangle of T, define

$$\begin{aligned} P_e^* &= \{e^* \in P^* : s(e^*) = e\} \\ P_v^* &= \{v^* \in P^* : s(v^*) = v\} \\ F_e &= \{f \in P^* : e \in T_f\} \\ F_\Delta &= \{f \in P^* : \Delta \in T_f\} \end{aligned}$$

where v^* and e^* are vertices and edges of P^*, respectively, and f is a facet of P.

4.2 The mapping τ

Lemma 4.2 *Let e be an edge of T. If $d^*, e^* \in P_e^*$ and $\rho(d^*), \rho(e^*)$ cross, then either there is an endpoint w of $\rho(d^*)$ with $d(w, e^*) < \kappa$ or an endpoint w' of $\rho(e^*)$ with $d(w', d^*) < \kappa$.*

The intricate proof of this lemma is omitted due to lack of space.

Lemma 4.3 *Let e be an edge of T. There is a mapping τ defined on P_e^* so that*

1. *For all edges $e^* \in P_e^*$, $\tau(e^*)$ is an edge over e with endpoints among the endpoints of $\rho(P_e^*)$.*

2. *For all edges e^*, $d(\tau(e^*), e^*) < \kappa$.*

3. *$\tau(P_e^*)$ is noncrossing.*

4. *τ can be computed in time quadratic in the size of P_e^*.*

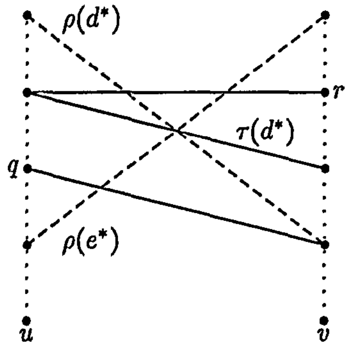

Figure 5: Definition of τ on new edge e^*.

Proof: We define τ inductively, adding edges of P_e^* one by one in arbitrary order. The addition of an edge may change the definition of τ on other edges as well; however, properties (1) through (3) of the lemma statement are maintained.

So suppose τ has been defined on a subset S of P_e^* and e^* is the next edge. If no edge of $\tau(S)$ crosses $\rho(e^*)$, then simply define $\tau(e^*) = \rho(e^*)$. Otherwise, since $\tau(S)$ is noncrossing, we can assume up to a symmetric argument that every edge $\tau(d^*)$ crossing $\rho(e^*)$ has $\tau(d^*)_u \succ \rho(e^*)_u$ and $\tau(d^*)_v \prec \rho(e^*)_v$. Let q be the highest (in $\prec$) vertex in $\tau(S)_u$ so that $d(q, e^*) < \kappa$ and let r be the lowest vertex in $\tau(S)_v$ so that $d(r, e^*) < \kappa$. If there is an edge $q'r'$ not crossing any edge in $\tau(S)$ with $q' \in \tau(S)_u$, q' between $\rho(e^*)_u$ and q, $r' \in \tau(S)_v$, and r' between $\rho(e^*)_v$ and r, define $\tau(e^*) = q'r'$; condition (2) of the lemma is easily verified. Otherwise, let S' be the subset of S crossing qr; S' must not be empty. Clearly for any $d^* \in S'$, $\tau(d^*)_u \succ q$ and $\tau(d^*)_v \prec r$. See figure 5.

We claim that for any edge $\tau(d^*) \in S'$, either $d(q, d^*) < \kappa$ or $d(r, d^*) < \kappa$. If $\rho(d^*)_u \preceq q$, then certainly $d(q, d^*) < d(\tau(d^*)_u, d^*) < \kappa$. Similarly if $\rho(d^*)_v \succeq r$, then $d(r, d^*) < \kappa$. Otherwise $\rho(d^*)_u \succ q \succeq \rho(e^*)_u$ and $\rho(d^*)_v \prec r \preceq \rho(e^*)_v$, so $\rho(d^*)$ crosses $\rho(e^*)$. See figure 5. By lemma 4.2 and the definition of q and r, either $d(\rho(e^*)_v, d^*) < \kappa$ or $d(\rho(e^*)_u, d^*) < \kappa$, so either $d(q, d^*) < \kappa$ or $d(r, d^*) < \kappa$.

Let Q be the set of edges $d^* \in S'$ so that $d(q, d^*) < \kappa$, and $R = S' \setminus Q$. Define $\tau(e^*) = qr$; for $d^* \in Q$, redefine $\tau(d^*)_u = q$; and for $d^* \in R$, redefine $\tau(d^*)_v = r$. It is easy to check that τ satisfies conditions (1) through (3). The running time is immediate. □

Henceforth we let τ be defined on all edges of P^*, by choosing a definition on P_e^* separately for each edge e of T, using lemma 4.3. Since there can be $O(n^2)$ edges e in T, and $O(n)$ edges in P_e^*, computation of τ takes time $O(n^4)$.

There is no guarantee that $d(e^*, \tau(e^*)) \leq \kappa$ nor that $\tau(e^*)$ and $\rho(e^*)$ have the same endpoints. In section 6, we guarantee both properties by in effect augmenting $\tau(e^*)$ to a polygonal chain using vertical edges connecting its endpoints to the endpoints of $\rho(e^*)$.

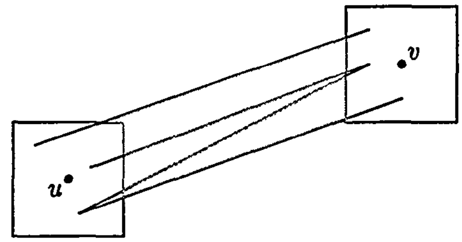

Figure 6: R_e is the shaded region plus the portion of the edges inside pixel(u) and pixel(v).

5 Lifting triangle edges

The desired embedding $\sigma(f)$ of facet f of P will eventually be obtained by lifting each vertex, edge, and triangle of T_f to three dimensions. This section handles a technically difficult case, the lifting of an edge e of a triangle Δ of T_f to the "lifted edge" $l_{f\Delta}(e)$. The lifted edge will satisfy three properties: $f \preceq f'$ implies $l_{f\Delta}(e) \preceq l_{f'\Delta}$, i.e. vertical order is preserved or collapsed; $d(l_{f\Delta}(e), f) \leq \kappa$; and no pair of lifted edges cross.

5.1 The order $\lhd$ and the snapping lemma

Let edge e of T have endpoints u and v. Define R_e to be the convex hull of $\pi_{xy}(P_e^*)$, less the interior of pixel(u) and pixel(v), unioned with $\pi_{xy}(P_e^*)$. See figure 6. Notice that there are no intersections among the boundaries of $\{\pi_{xy}(f) : f \in F_\Delta\}$ within R_e except possibly at the endpoints of edges of $\pi_{xy}(P_e^*)$. Facet $f \in F_e$ *covers* e if no edge in P_e^* bounds f; it is easy to check that $R_e \subseteq \pi_{xy}(f)$. A facet f *covers facet* f' *at* e if $\pi_{xy}(f') \cap R_e \subseteq \pi_{xy}(f) \cap R_e$. For any two facets $f, f' \in F_e$, either f covers f' at e, or f' covers f at e.

Suppose that e is an edge of triangle Δ of T. The *covering order* $\lhd$ on the facets in F_Δ is any total order so that $f \lhd f'$ implies f' covers f at e. (The order depends on both e and Δ, but to keep the notation simple we do not make this dependence explicit.) The order $\lhd$ can be described as follows. Assume that Δ lies to the left of the e, directed from endpoint u to endpoint v; direct all edges in P_e^* from pixel(u) to pixel(v). If facets $f_0, f_1 \in F_\Delta$ have bounding edges $e_0^*, e_1^* \in P_e^*$, then $f_0 \lhd f_1$ if e_0^* is to the left of e_1^*; all facets covering e appear at the end of the order $\lhd$, and are ordered arbitrarily among themselves.

For a set $S \subset \mathbb{R}^3$, let $V(S)$ be all points on all vertical lines through S. Let f_A be a facet of P, e an edge of T, and A an edge over e with endpoints u and v. Edge A *approximates* f_A *at* e if $d(A_u, f_A \cap V(R_e)) < \kappa$ and $d(A_v, f_A \cap V(R_e)) < \kappa$. Clearly, if A approximates f_A at e, then $d(A, f_A) < \kappa$. Also, if $e^* \in P_e^*$ is a boundary

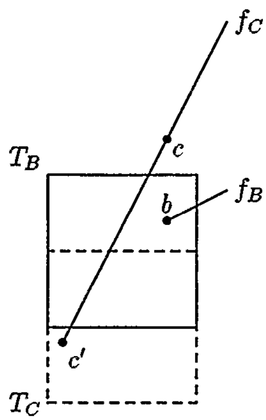

Figure 7: Proof of lemma 5.1, side view. T_B is solid square, T_C dashed.

edge of face f, then $\tau(e^*)$ approximates f at e.

Lemma 5.1 *Let edge e bound triangle Δ of T. Suppose $f_A, f_B, f_C \in F_\Delta$ with $f_A \succeq f_C \succeq f_B$, A, B, C are edges over e approximating f_A, f_B, f_C, respectively, $A \succeq B$, and f_C covers f_A and f_B. Then* snap$(C, [A, B])$ *also approximates f_C at e.*

Proof: We claim $\max(B, C)$ approximates f_C at e; a similar result holds for min, from which the lemma follows. Let u be an endpoint of e. We show

$$d(\max(B_u, C_u), f_C \cap V(R_e)) < \kappa.$$

If $C_u \succeq B_u$, there is nothing to prove, so suppose $C_u \prec B_u$.

Let T_B and T_C be the cubes of sidelength 2κ centered at B_u and C_u, respectively, and $T = V(T_B)$ (clearly also $T = V(T_C)$). See figure 7.

Since B approximates f_B, there is a point $b \in f_B \cap V(R_e) \cap T_B$. Since $f_C \succeq f_B$ and f_C covers f_B at e, there is a point $c \in f_C$ with $c \succeq b$; clearly $c \in T$. Since C approximates f_C, there is a point $c' \in f_C \cap V(R_e) \cap T_C$. Since $f_C \cap V(R_e) \cap T$ is path-connected, there is a path in $f_C \cap V(R_e) \cap T$ from c to c'. Since c is above the bottom facet of T_B, c' is below the top facet of T_C, and $T_B \succeq T_C$, some point of the path meets T_B. Hence $d(B_u, f_C) < \kappa$. □

5.2 Default edges.

Let e be an edge of T_f with endpoints u and v and with some triangle incident. We define the *default lifting of edge e for facet f, $c_f(e)$*, which is to be used in the absence of other constraints. If e is a boundary edge of T_f, then there is a unique edge $e^* \in P_e^*$ bounding facet f, and we define $c_f(e) = \tau(e^*)$. Otherwise suppose e is an interior edge of T_f; clearly f covers e and no edge in P_e^* meets f. The definition is slightly complicated because of the requirement that e not cross any edge in $\tau(P_e^*)$.

Define low$_f(e)$ to be the edge $\hat{u}\hat{v}$, where $\hat{u}$ is the center of the lowest voxel X in column(u) so that $X \cap f \cap V(R_e)$ is not empty, and similarly for $\hat{v}$. A pair of distinct edges (a^*, b^*) in P_e^* is a *bracketing pair* if $a^* \succ f$ and $f \succ b^*$, $\tau(a^*) \succeq \tau(b^*)$ and no edge $\tau(d^*)$, $d^* \in P_e^*$, lies between $\tau(a^*)$ and $\tau(b^*)$ (possibly $\tau(a^*) = \tau(b^*)$). The existence of a bracketing pair can be seen by indexing the edges of $P_e^* = \{e_0^*, \ldots, e_k^*\}$ so that $\tau(e_0^*) \succeq \tau(e_1^*) \succeq \ldots \succeq \tau(e_k^*)$. Either $f \succ e_0^*$, and the pair $(\top, e_0^*)$ suffices (with the definition $\tau(\top) = \top$); or $e_k^* \succ f$, and the pair $(e_k^*, \bot)$ suffices; or there is i so that an $e_i \succ f$ and $f \succ e_{i+1}$, and the pair (e_{i+1}, e_i) suffices. Define

$$c_f(e) = \text{snap}(\text{low}_f(e), [\tau(a^*), \tau(b^*)]),$$

choosing bracketing pair (a^*, b^*) so that $(\tau(a^*), \tau(b^*))$ is minimal in $\prec$ among bracketing pairs. Set $C(e) = \{c_f(e) : f \in F_\Delta\}$.

Lemma 5.2 *Let f be a facet of P and e an edge of T_f.*

1. *$c_f(e)$ approximates f at e.*
2. *If f, f' cover e and $f \preceq f'$, then $c_f(e) \preceq c_{f'}(e)$.*
3. *$\tau(P_e^*) \cup C(e)$ is noncrossing.*

Proof:

1. If e is a bounding edge of T_f, then the claim is immediate. Otherwise

$$c_f(e) = \text{snap}(\text{low}_f(e), [\tau(a^*), \tau(b^*)]),$$

for some bracketing pair (a^*, b^*). Let a^* and b^* be incident to faces f_a and f_b, respectively. It is easy to check that low$_f(e)$ approximates f. Clearly $\tau(a^*)$ approximates f_a, $\tau(b^*)$ approximates f_b, and f covers f_a and f_b (since e is not a bounding edge of T_f). Part (1) is thus immediate from lemma 5.1.

2. We have $R_e \subseteq \pi_{xy}(f)$, $R_e \subseteq \pi_{xy}(f')$, and $f \prec f'$, so low$_f(e) \preceq$ low$_{f'}(e)$. Let a_f^*, b_f^* and $a_{f'}^*, b_{f'}^*$ be bracketing pairs for f and f', respectively. We claim $\tau(a_f^*) \preceq \tau(a_{f'}^*)$ and $\tau(b_f^*) \preceq \tau(b_{f'}^*)$, from which $c_f(e) \preceq c_{f'}(e)$ follows easily. Clearly $\tau(a_f^*) \succeq \tau(b_f^*)$. It cannot be that $\tau(b_f^*) \succ \tau(a_{f'})$, for $a_{f'} \succ f' \succeq f$ and f would have a bracketing pair below (a_f^*, b_f^*), contradicting minimality. No edge of $\tau(P_e^*)$ lies between $\tau(a_f^*)$ and $\tau(b_f^*)$, so it must be that $\tau(a_{f'}^*) \succeq \tau(a_f^*)$. Similarly $\tau(b_{f'}^*) \succeq \tau(b_f^*)$.

3. Clearly no edge $c_f(e)$ crosses an edge of $\tau(P_e^*)$. Also clearly, if f and f' have distinct bracketing pairs, then $c_f(e)$ and $c_{f'}(e)$ do not cross. If f and f' have the same bracketing pair, then $c_f(e)$ and $c_{f'}(e)$ do not cross because low$_f(e)$ and low$_{f'}(e)$ do not cross. □

5.3 Lifting triangle edges.

Let e be an edge of triangle Δ of T. For facets $f \in F_\Delta$ in the order $\lhd$, simultaneously and inductively define

$a_{f\Delta}(e)$ and $b_{f\Delta}(e)$ (the *constraints from above and below, respectively*) and $l_{f\Delta}(e)$ (the *lifting of edge e of* Δ *in* f), as follows:

$$\begin{aligned} a_{f\Delta}(e) &= \min\{l_{f'\Delta}(e) : f' \lhd f \text{ and } f' \succ f\} \\ b_{f\Delta}(e) &= \max\{l_{f'\Delta}(e) : f' \lhd f \text{ and } f' \prec f\} \\ l_{f\Delta}(e) &= \text{snap}(c_f(e), [a_{f\Delta}(e), b_{f\Delta}(e)]). \end{aligned}$$

We have $a_{f\Delta}(e) \succeq b_{f\Delta}(e)$ by lemma 5.3(1) below.

Lemma 5.3 *Let e be an edge of triangle* Δ *of* T *with edge e, and let* $f, f' \in F_\Delta$.

1. $a_{f\Delta}(e) \succeq b_{f\Delta}(e)$.
2. *If* $f \preceq f'$, *then* $l_{f\Delta}(e) \preceq l_{f'\Delta}(e)$.
3. $l_{f\Delta}(e)$ *approximates* f *at* e.
4. $l_{f\Delta}(e) \in \tau(P_e^*) \cup C(e)$.

Proof: 1. and 2. We prove both simultaneously by induction on $\lhd$. If $a_{f\Delta}(e) = \top$ or $b_{f\Delta}(e) = \bot$, $a_{f\Delta}(e) \succeq b_{f\Delta}(e)$ is immediate. Otherwise $a_{f\Delta}(e) = l_{f_0\Delta}(e)$ and $b_{f\Delta}(e) = l_{f_1\Delta}(e)$ for some facets $f_0 \succeq f \succeq f_1$, so by induction hypothesis $a_{f\Delta}(e) \succeq b_{f\Delta}(e)$. For (2), suppose $f \preceq f'$; without loss of generality assume $f' \lhd f$. Then by definition $l_{f'\Delta}(e) \succeq a_{f\Delta}(e) \succeq l_{f\Delta}(e)$.

3. Since $l_{f\Delta}(e)$ is defined in the order $\lhd$, the claim follows from an easy induction using lemma 5.1.

4. By lemma 5.2, $\tau(P_e^*) \cup C(e)$ is noncrossing, so the "snap" in the definition of $l_{f\Delta}(e)$ results in an element of $\tau(P_e^*) \cup C(e)$. □

6 The subdivision Q

In this section we define the subdivision Q and the embedding σ of P into Q required by theorem 2.1. For technical reasons σ is defined on P^* as well.

We first define a "vertical carrier" over each vertex and edge of T. For v a vertex of T, let $VC(v)$ be the vertical chain of edges through $\rho(P_v^*)$, i.e. all edges connecting two vertices of $\rho(P_v^*)$ that are adjacent in vertical order. Let e be an edge of T with endpoints u and v. Consider the edges $\tau(P_e^*) \cup \{l_{f\Delta}(e) : \Delta \text{ incident to } e, f \in F_\Delta\}$; they are noncrossing, by lemmas 5.1(3) and 5.3(4). Split each edge at its midpoint. These edges together with $VC(u)$ and $VC(v)$ form a planar graph (in the plane through $VC(u)$ and $VC(u)$); let $VC(e)$ be any triangulation of this graph.

For v^* a vertex of P^*, define $\sigma(v^*) = \rho(v^*)$. Let $e^* \in P_e^*$, where edge e in T has endpoints u and v. Define $\sigma(e^*)$ to be the subdivision consisting of $\tau(e^*)$, the subchain of $VC(u)$ connecting $\tau(e^*)_u$ to $\rho(e^*)_u$ and the subchain of $VC(v)$ connecting $\tau(e^*)_v$ to $\rho(e^*)_v$. Extend σ to edges e of P:

$$\sigma(e) = \bigcup_{e^* \in P^*,\, e^* \subseteq e} \sigma(e^*)$$

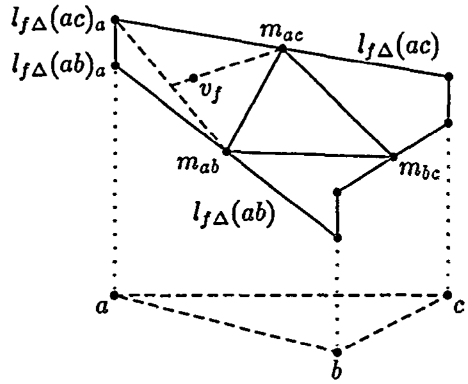

Figure 8: Construction of $l_f(\Delta)$ for triangle Δ of T_f with vertices a, b, c.

Clearly $\sigma(e)$ is a subdivision. We extend σ to facets in section 6.2.

Lemma 6.1 *If* w, w' *are vertices or edges of* P *and* $w \preceq w'$, *then* $\sigma(w) \preceq \sigma(w')$.

Proof: The claim is immediate for two vertices. Suppose w is a vertex and w' an edge; the symmetric case is similar. Then $\sigma(w')$ contains a vertical chain of edges from the center of the first voxel in column(w) met by w' to the center of the last such voxel. Since $w \prec w'$, $\rho(w)$ is below or on the chain, and $\sigma(w) \preceq \sigma(w')$. The case of two edges is similar. □

6.1 Lifting triangles

Let Δ be a triangle of T_f with vertices a, b, c. Consider the edges $l_{f\Delta}(ab)$, $l_{f\Delta}(ac)$, $l_{f\Delta}(bc)$. There is no guarantee that these edges are pairwise incident (of course both $l_{f\Delta}(ab)$ and $l_{f\Delta}(ac)$ are incident to vertices over a, and similarly for the other pairs). We form a (three-dimensional) polygon from $l_{f\Delta}(ab)$, $l_{f\Delta}(ac)$, $l_{f\Delta}(bc)$ by adding the vertical subchain of $L(a)$ connecting $l_{f\Delta}(ab)_a$ to $l_{f\Delta}(ac)_a$ (if they are not equal) and similarly for the b and c endpoints. The *lifting of* Δ *for facet* f, $l_f(\Delta)$, is a triangulation of this polygon, described as follows.

Split edges $l_{f\Delta}(ab)$, $l_{f\Delta}(ac)$, $l_{f\Delta}(bc)$ at their respective midpoints m_{ab}, m_{ac}, m_{bc}, and add the three edges connecting midpoints. This forms a triangle $m_{ab}m_{ac}m_{bc}$ and three polygons, where e.g. the *a-polygon (of f)* consists of edge $m_{ab}m_{ac}$, the two subedges of $l_{f\Delta}(ab)$ and $l_{f\Delta}(ac)$ with endpoints over a, and possibly a vertical chain over a. See figure 8.

For points $p, q \in \mathbb{R}^3$ and $\alpha \in \mathbb{R}$, let $\alpha[p, q]$ be the point $(1-\alpha)p + \alpha q$, i.e. the point a fraction α of the way from p to q.

The *a-index* of f is the number of distinct pairs $(l_{f'\Delta}(ab), l_{f'\Delta}(ac))$, where $f' \succeq f$. Let $\alpha_f = i/2^{\lceil \log_2 n \rceil}$, where i is the a-index of f; clearly $0 < \alpha_f < 1$. First assume $l_{f\Delta}(ac)_a \succeq l_{f\Delta}(ab)_a$. Set

$$v_f = \alpha_f[\frac{1}{2}[l_{f\Delta}(ac)_a, m_{ab}], m_{ac}].$$

See figure 8. Triangulate the a-polygon of f with v_f, i.e. connect v_f to m_{ac}, m_{ab}, and any vertex on the chain from $l_{f\Delta}(ab)_a$ to $l_{f\Delta}(ac)_a$. If $l_{f\Delta}(ab)_a \succ l_{f\Delta}(ac)_a$, the construction is similar, with m_{ab} and m_{ac} interchanged and $l_{f\Delta}(ab)_a$ substituting for $l_{f\Delta}(ac)_a$. The other two polygons are triangulated in a similar fashion.

Lemma 6.2 *Let Δ be a triangle of T and $f, f' \in F_\Delta$.*

1. *$d(l_f(\Delta), f) < \kappa$.*
2. *If $f \preceq f'$, then $l_f(\Delta) \preceq l_{f'}(\Delta)$.*
3. *Every vertex coordinate of $l_f(\Delta)$ is an integral multiple of $1/2^{\lceil \log_2 n \rceil + 2}$.*
4. *$l_f(\Delta)$ has $O(n)$ cells.*

Proof:
1. Let Δ have vertices a, b, c. Every vertex of $l_f(\Delta)$ is within the convex hull of $\{l_{f\Delta}(ab), l_{f\Delta}(ac), l_{f\Delta}(bc)\}$. The claim follows using lemma 5.3(1) and proposition 3.1.

2. We can assume that $l_{f\Delta}(ac)_a \succeq l_{f\Delta}(ab)_a$. Using lemma 5.3(2), we must have

$$l_{f'\Delta}(ac)_a \succeq l_{f\Delta}(ac)_a \text{ and } l_{f'\Delta}(ab)_a \succeq l_{f\Delta}(ab)_a.$$

If $l_{f'\Delta}(ab)_a \succ l_{f\Delta}(ac)_a$, then the result is immediate, since the convex hull of $\{l_{f\Delta}(ab), l_{f\Delta}(ac)\}$ and the convex hull of $\{l_{f'\Delta}(ab), l_{f'\Delta}(ac)\}$ have disjoint interiors. Hence we can assume that

$$l_{f'\Delta}(ac)_a \succeq l_{f\Delta}(ac)_a \succeq l_{f'\Delta}(ab)_a \succeq l_{f\Delta}(ab)_a.$$

Let i_f and $i_{f'}$ be the a-indices of f and f' respectively. If $i_f = i_{f'}$, then the a-polygons for f and f' are identical. Otherwise, $i_f > i_{f'}$ since $f \preceq f'$. Let s be the edge connecting $v_{f'}$ to the midpoint of $l_{f'\Delta}(ac)$; clearly we have $\pi_{xy}(v_f) \in \pi_{xy}(s)$ since $\alpha_f > \alpha_{f'}$. Furthermore we have $v_f \prec s$, since $l_{f\Delta}(ab) \preceq l_{f'\Delta}(ab)$ and $l_{f\Delta}(ac) \preceq l_{f'\Delta}(ac)$ with inequality holding in at least one case. $l_f(\Delta) \preceq l_{f'}(\Delta)$ follows easily.

3., 4. Immediate. □

6.2 Vertical ordering

It is tempting to define $\sigma(f) = \bigcup_{\Delta \in T_f} l_f(\Delta)$. This definition would preserve or collapse vertical order (in the sense of theorem 2.1) over triangles of T, but not necessarily over edges and vertices. Hence we develop an alternate definition of $\sigma(f)$.

Let $E_f(v)$ be the set of all endpoints over v of all edges $l_{f\Delta}(e)$, where edge e of T is incident to v and triangle Δ of T is incident to e. Define

$$\begin{aligned} a_f(v) &= \min\{\sigma(v^*) : v^* \in P_v^* \text{ and } v^* \succeq f\} \cup E_f(v) \\ b_f(v) &= \max\{\sigma(v^*) : v^* \in P_v^* \text{ and } v^* \preceq f\} \cup E_f(v). \end{aligned}$$

The *lifting of vertex v for facet f*, $l_f(v)$, is the subchain of $VC(v)$ connecting $a_f(v)$ and $b_f(v)$.

For e an edge of T_f, let $E_f(e)$ be all edges $l_{f\Delta}(e)$, where Δ varies over triangles in T_f incident to e. Define

$$\begin{aligned} a_f(e) &= \min\{\sigma(e^*) : e^* \in P_e^* \text{ and } e^* \succeq f\} \cup E_f(e) \\ b_f(e) &= \max\{\sigma(e^*) : e^* \in P_e^* \text{ and } e^* \preceq f\} \cup E_f(e). \end{aligned}$$

The *lifting of edge e for facet f*, $l_f(e)$, is all edges and vertices w of $VC(e)$ satisfying $b_f(e) \succeq w$ and $w \succeq a_f(e)$.

Lemma 6.3 *Suppose w is a vertex or edge of T, $w^* \in P_w^*$, and f is a facet of P. Then $w^* \preceq f$ implies $\sigma(w^*) \preceq l_f(w)$ and $w^* \succeq f$ implies $\sigma(w^*) \succeq l_f(w)$.*

Proof: By construction. □

Lemma 6.4 *Let f be a facet of P and w an edge of T_f. Then $d(l_f(w), f) \leq \kappa$.*

Proof: Similar to the proof of lemma 5.1. □

For each facet f of P, define

$$\sigma(f) = \bigcup_{w \in T_f} l_f(w),$$

where w varies over vertices, edges, and triangles. It is easy to check that $\sigma(f)$ is a subdivision.

Lemma 6.5 *If f, f' are cells of P and $f \prec f'$, then $\sigma(f) \preceq \sigma(f')$.*

Proof: The lemma follows from lemmas 6.1 and 6.3 if one of f and f' is a vertex or edge. So suppose both are facets. For each triangle Δ in both T_f and $T_{f'}$, $l_f(\Delta) \preceq l_{f'}(\Delta)$ by lemma 6.2. Suppose e is an edge in both T_f and $T_{f'}$. If there is a triangle Δ in both T_f and $T_{f'}$ incident to e, lemma 6.2 again implies $l_f(e) \preceq l_{f'}(e)$. Otherwise, up to symmetry, there is an edge $e^* \in P_e^*$ bounding f with $e^* \preceq f'$, so by lemma 6.3, $\sigma(e^*) \preceq l_{f'}(e)$. Since $e^* \preceq f$, $\sigma(e^*) \subseteq l_f(e)$, and $l_f(e) \preceq l_{f'}(e)$. A similar argument shows that if v is a vertex in both T_f and $T_{f'}$, then $l_f(v) \preceq l_{f'}(v)$. Hence $\sigma(f) \preceq \sigma(f')$. □

6.3 The subdivision Q

Let $Q = \bigcup_f \sigma(f)$, where f varies over all facets of P. It is easy to check that Q is a subdivision and that σ is an embedding of P into Q.

Lemma 6.6 *Q has $O(n^4)$ cells and can be computed in time $O(n^4)$.*

Proof: For each facet f of P, T_f has $O(n^2)$ triangles Δ. By lemma 6.2, $l_f(\Delta)$ has $O(n)$ cells. Hence $\sigma(f)$ has $O(n^3)$ cells, for a total of $O(n^4)$ over all facets of f. Q can easily be computed in the same time. □

6.4 Hausdorff distance

It follows immediately from lemmas 6.2 and 6.4 that $d(\sigma(f), f) \leq \kappa$. The proof that $d_H(\sigma(f), f) \leq \kappa$ requires a proof that $d(f, \sigma(f)) \leq \kappa$. This proof has a topological flavor, using $d(e, \sigma(e)) \leq \kappa$ for each edge e bounding f. The proof is omitted due to lack of space.

7 Discussion

It may be possible to improve the worst-case bounds given in theorem 2.1. For example, the $O(n^4)$ bound on the size of Q could be an artifact of vertical projection; perhaps an $O(n^3)$ bound could be obtained by using different projection directions in different places, each tuned to the local configuration. Obtaining a worst-case bound below $O(n^3)$ seems very challenging. It would also be desirable to remove the extra $\lceil \log_2 n \rceil + 2$ bits needed for vertex coordinates; again, this may be an artifact of vertical projection.

The algorithm of theorem 2.1 adds many vertices, far more than are necessary unless the input subdivision has been chosen by an adversary. Another challenge is to devise a straightforward algorithm that adds vertices only to nearby features, just enough to avoid self-intersections and to maintain combinatorial ordering. Presumably, most subdivisions would need far fewer new vertices than the bounds given in theorem 2.1.

A programmer would probably prefer a simple rounding algorithm, even at the expense of degraded worst-case bounds, as long as the typical-case bounds are reasonable. One reason that the rounding algorithm is complicated is the need to avoid edge crossings. Milenkovic [17] suggests rounding existing vertices to integer coordinates. If two rounded edges cross, then a vertex of intersection is added, with coordinates computed exactly. This would require a constant-factor increase in the bit-length of some vertex coordinates, and hence of some predicate evaluations. However, the maximum required bit-length is still bounded, and perhaps the increased-length calculations are relatively infrequent. It may be that this approach can lead to a practical rounding algorithm.

References

[1] H. Brönnimann, M. Yvinec, Efficient exact evaluation of signs of determinants, *Proc. Thirteenth Ann. Symp. Comp. Geom*, pp. 166-173, 1997.

[2] H. Brönnimann, I. Emiris, V. Pan, S. Pion, Computing exact geometric predicates using modular arithmetic with single precision, *Proc. Thirteenth Ann. Symp. Comp. Geom*, pp. 174-182, 1997.

[3] K.L. Clarkson, Safe and effective determinant evaluation, *33th Symp. on Found. Comp. Sci. 387-395, 1992.*

[4] S. Fortune, Robustness issues in geometric algorithms, *Applied computational geometry: towards geometric engineering*, pp. 9-14, Lecture notes in computer science 1148, M. Lin, D. Manocha, eds., Springer-Verlag, 1996.

[5] S. Fortune, Polyhedral modelling with multiprecision integer arithmetic, *Computer-Aided Design*, 20:123-133, 1997.

[6] S. Fortune, C. Van Wyk, Static analysis yields efficient exact integer arithmetic for computational geometry, *ACM Trans. Graphics*, 15(3), pp. 223-248, July 1996.

[7] M. Goodrich, L. Guibas, J. Hershberger, P. Tanenbaum, Snap-rounding line segments efficiently in two and three dimensions, *Proc. Thirteenth Ann. Symp. Comp. Geom*, pp. 284-293, 1997.

[8] D. Greene, F. Yao, Finite-resolution computational geometry, *Proc. 27th IEEE Symp. Found. Comp. Sci.*, pp. 143-152, 1986.

[9] D. Greene, Integer line segment intersection, unpublished manuscript.

[10] L. Guibas, D. Marimount, Rounding arrangements dynamically, *Proc. Eleventh Ann. Symp. Comp. Geom*, pp. 190-199, 1995.

[11] J. Hobby, Practical segment intersection with finite precision output, *Computational geometry: theory and applications*, to appear.

[12] C. Hoffmann, The problems of accuracy and robustness in geometric computation, *Computer* 22:31-42, 1989.

[13] P. Jaillon, Proposition d'une arithmétique rationnelle paresseuse et d'un outil d'aide à la saise d'objets en synth'èse, Ecole Nationale Superieure des Mine de Saint-Etienee, 1993.

[14] M. Karasick, D. Lieber, L. Nackman, Efficient Delaunay triangulation using rational arithmetic, *ACM Trans. Graphics* 10(1):71-91, 1990.

[15] V. Milenkovic, Verifiable implementations of geometric algorithms using finite precision arithmetic, *Artificial intelligence*, 37:377-401, 1988.

[16] V. Milenkovic, Rounding face lattices in d dimensions, Abstract for *Second Canadian Computational Geometry Conference*, 1990.

[17] V. Milenkovic, private communication, 1996.

[18] V. Milenkovic, Shortest path geometric rounding, submitted.

[19] V. Milenkovic, L. Nackman, Finding compact coordinate representations for polygons and polyhedra, *IBM J. Res. Dev.* 34(5):752-768, 1990. A version also appeared in *Proc. Sixth Ann. Symp. Comp. Geom.* 244-252, 1990.

[20] J. Shewchuk, Robust adaptive floating-point geometric predicates, *Proc. Twelfth Ann. Symp. Comp. Geom*, pp. 141-150, 1997.