Bounded-Degree Polyhedronization of Point Sets

Gill Barequet[a,b,1], Nadia Benbernou[c], David Charlton[c], Erik D. Demaine[c], Martin L. Demaine[c], Mashhood Ishaque[a,2], Anna Lubiw[d], André Schulz[e,3], Diane L. Souvaine[a,2], Godfried T. Toussaint[f,g,a,4], Andrew Winslow[a,2,*]

[a] *Department of Computer Science, Tufts University*
[b] *Department of Computer Science, The Technion—Israel Institute of Technology*
[c] *Computer Science and Artificial Intelligence Laboratory, MIT*
[d] *David R. Cheriton School of Computer Science, University of Waterloo*
[e] *Institut für Mathematische Logik und Grundlagenforschung, Westfälische Wilhelms-Universität, Münster*
[f] *Department of Music, Harvard University*
[g] *School of Computer Science, McGill University*

Abstract

In 1994 Grünbaum showed that, given a point set S in $\mathbb{R}^3$, it is always possible to construct a polyhedron whose vertices are exactly S. Such a polyhedron is called a *polyhedronization* of S. Agarwal et al. extended this work in 2008 by showing that there always exists a polyhedronization that can be decomposed into a union of tetrahedra (*tetrahedralizable*). In the same work they introduced the notion of a *serpentine* polyhedronization for which the dual of its tetrahedralization is a chain. In this work we present a randomized algorithm running in $O(n \log^6 n)$ expected time which constructs a serpentine polyhedronization that has vertices with degree at most 7, answering an open question by Agarwal et al.

Keywords: serpentine, tetrahedralization, convex hull, gift wrapping

---

[*] Corresponding author

Email addresses: `barequet@cs.tufts.edu` (Gill Barequet), `nbenbern@mit.edu` (Nadia Benbernou), `dchar@mit.edu` (David Charlton), `edemaine@mit.edu` (Erik D. Demaine), `mdemaine@mit.edu` (Martin L. Demaine), `mishaq01@cs.tufts.edu` (Mashhood Ishaque), `alubiw@uwaterloo.ca` (Anna Lubiw), `andre.schulz@uni-muenster.de` (André Schulz), `dls@cs.tufts.edu` (Diane L. Souvaine), `godfried@cs.mcgill.ca` (Godfried T. Toussaint), `awinslow@cs.tufts.edu` (Andrew Winslow)

[1] Research supported in part by a grant from Intel Corporation.

[2] Research supported in part by NSF grants CCF-0830734 and CBET-0941538.

[3] Partially supported by the German Research Foundation (DFG) under grant SCHU 2458/1-1.

[4] Research supported in part by NSERC (Canada) and the Radcliffe Institute for Advanced Study at Harvard University.

1. Introduction

Any set S of points in the plane (not all of which are collinear) admits a *polygonization*, that is, there is a simple polygon whose vertex set is exactly S. Similarly, a point set $S \subset \mathbb{R}^3$ admits a *polyhedronization* if there exists a simple polyhedron that has exactly S as its vertices. In 1994, Grünbaum proved that every point set in $\mathbb{R}^3$ (not all of which are coplanar) admits a polyhedronization. Unfortunately, the polyhedronizations generated by Grünbaum's method can be impossible to tetrahedralize. This is because they may contain *Schönhardt polyhedra*, a class of nontetrahedralizable polyhedra [5].

In 2008, Agarwal, Hurtado, Toussaint, and Trias described a variety of methods for producing polyhedronizations with various properties [1]. One of these methods, called hinge polyhedronization, produces *serpentine polyhedronizations*, meaning that the polyhedron admits a tetrahedralization whose dual (a graph where each tetrahedron is a node and each edge connects a pair of nodes whose primal entities are tetrahedra sharing a face) is a chain. Serpentine polyhedronizations produced by the hinge polyhedronization method are guaranteed to have two vertices with edges to every other vertex in the set. As a result, two vertices in these constructions have degree $n - 1$, where n is the number of points in the set. It should be noted that some tetrahedra produced by this method may be degenerate if the point set is not in general position, that is, it contains four coplanar points. A natural question, and one posed by Agarwal et al., is whether it is always possible to create serpentine polyhedronizations with bounded degree.

In this work we describe a randomized algorithm for point sets in general position that constructs a serpentine polyhedronization with constant bounded degree. This algorithm runs in $O(n \log^6 n)$ expected time, and the expectation is independent of the input point set and output polyhedronization. The bound on the degree of the produced polyhedronizations is 7, which we show is nearly optimal for all sets of more than 12 points. Such bounded-degree serpentine polyhedronizations are useful in applications of modeling and graphics, where low local complexity is desirable for engineering and computational efficiency.

2. Setting

The convex hull of S, written $\mathcal{CH}(S)$, is the intersection of all half-spaces containing S. The boundary of each face of $\mathcal{CH}(S)$ is a polygon with coplanar vertices. In the next five sections we assume that S has no four coplanar points, and in the conclusion briefly discuss relaxing this assumption. so each of the faces of $\mathcal{CH}(S)$ is triangular. We call the three vertices composing a face of $\mathcal{CH}(S)$ a *face triplet*.

We will make reference to points and faces that *see* each other. We say that a pair of points p, q can see each other if the segment pq does not intersect a portion of any polyhedron present (either the convex hull of a set of points or a portion of the partially constructed polyhedronization). Similarly, two segments s_1 and s_2 can see each other if every pair of points $p \in s_1$ and $q \in s_2$ can see each other. A face f is the planar region bounded by a triangle formed by three points of S. A point p can see a face f if p can see every point in f (strong visibility). Similarly, a point p can see a segment s if p can see every point on s.

3. Algorithm

In this section we present a high-level description of the algorithm. Begin with a point set $S \subset \mathbb{R}^3$. Select a face triplet of $\mathcal{CH}(S)$ arbitrarily. Call this face triplet T_0. Let $S_0 = S \setminus T_0$. Assign the labels u_0, v_0, w_0 arbitrarily to the vertices of T_0 and connect the three vertices to form a triangle.

Next we search for a face triplet T_1 of $\mathcal{CH}(S_0)$ that we can attach to the triangle T_0 via a polyhedron *tunnel* (see Figure 1). The tunnel is tetrahedralizable and has the face triplet T_0 at one end, the face triplet T_1 at the other end, and it is disjoint from the interior of $\mathcal{CH}(S_0)$. The method for selecting T_1 is described in the next section. Once the tunnel is construction, we will require that vertex w_0 has degree 3 and vertices u_0 and v_0 have degrees 5 and 4 (not necessarily respectively). Moreover, the vertices of the face triplet T_1 which we will call u_1, v_1, w_1 should have degree 3, 4, and 5, respectively. Note that the vertex degree only counts edges in the tunnel and that the constructed tunnel must meet the degree requirements for the vertices of T_0 while it determines the labeling of the vertices u_1, v_1, w_1 in T_1.

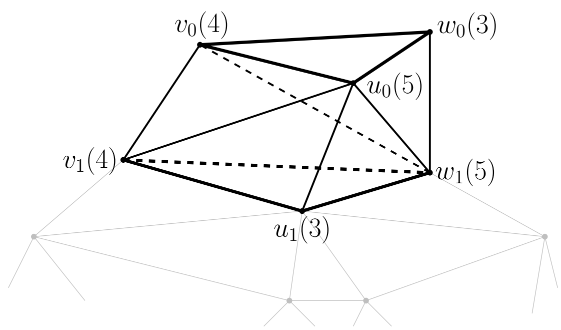

Figure 1: Constructing a tunnel between T_0, T_1. The vertices u_0 and v_0 have degree 5 and 4, while w_0 has degree 3. The other end of the tunnel, T_1, has three vertices that will be labeled u_1, v_1, w_1 with degree 3, 4, and 5 (shown in parentheses), respectively.

After finding a face triplet T_1 that meets these requirements, the process is repeated for T_1 and S_1, T_2 and S_2, where $S_i = S_{i-1} \setminus T_i$, until S_i contains fewer than three points. At each step T_i has three vertices: u_i connected to 3 vertices of S_i, and v_i, w_i that are connected to 1 and 2 vertices of S_i (not necessarily respectively). Once S_i contains fewer than three points, a degenerate tunnel is built out of the remaining points and the algorithm stops. In Sections 4 and 5 we prove that such a construction is always possible, producing a valid serpentine polyhedronization with vertex degrees bounded by 7. In Section 6 we provide details regarding the data structures used and provide an analysis of the algorithm's running time.

4. Tunnel Construction

Here we prove that given T_i, it is always possible to find a face triplet T_{i+1} such that a three-tetrahedron tunnel $(\Delta_1\Delta_2\Delta_3)$ can be constructed between them.

Refer to Figure 2. Let ℓ_1 denote the line through u_iv_i. Call H_1 the plane containing T_i (and thus ℓ_1). Note that the plane supporting T_i does not intersect $\mathcal{CH}(S_i)$ because T_i is a face of $\mathcal{CH}(S_{i-1})$. Rotate H_1 about ℓ_1 in the direction that maintains separation of w_i and $\mathcal{CH}(S_i)$ until $\mathcal{CH}(S_i)$ is intersected. By the general position assumption, this intersection will be at a vertex v_{cone}. Let H_2 be the plane through ℓ_1 and v_{cone}, and let R_1 be the wedge between H_1 and H_2 swept-out by the halfplane of H_1 containing T_i.

Now let ℓ_2 denote the line parallel to ℓ_1 through v_{cone}. Rotate H_2 about ℓ_2, starting at u_iv_i, in the direction that maintains the separation of u_iv_i and $\mathcal{CH}(S_i)$ until $\mathcal{CH}(S_i)$ is intersected. The intersection is either an edge or a face. If it is an edge, call this edge e. If it is a face, select an edge e of this face that has v_{cone} as an endpoint. Let H_3 be the plane containing ℓ_2 and e, and let R_2 be the wedge between H_2 and H_3 swept-out by the halfplane of H_2 containing ℓ_1.

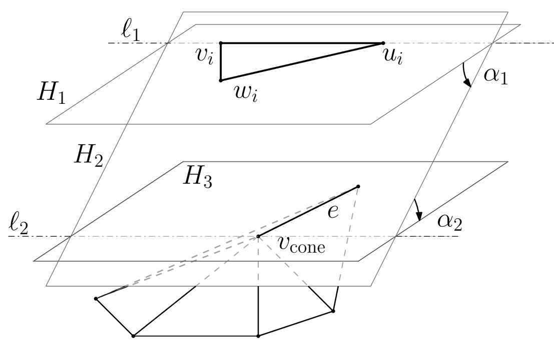

Figure 2: A visualization of the arrangement created by T_i. The angles α_1, α_2 denote the swept angular regions forming R_1 and R_2, respectively.

Lemma 1. *The segment u_iv_i can see edge e.*

Proof. Recall that the plane supporting T_i does not intersect $\mathcal{CH}(S_i)$, so w_i cannot interfere with visibility. Now consider a segment connecting a point on u_iv_i and a point on e. This segment is contained in R_2, which does not intersect $\mathcal{CH}(S_i)$. Thus, neither w_i nor $\mathcal{CH}(S_i)$ can block visibility between u_iv_i and e.

Connect the endpoints of e to u_i and v_i with four edges to form the middle tetrahedron Δ_2. The endpoint of e that is not v_{cone} is connected to two vertices of T_i and will not be connected to w_i. Thus, this vertex is v_{i+1}.

Lemma 2. *Vertex w_i can see face $u_iv_iv_{cone}$ of Δ_2.*

Proof. The swept-out region R_1 does not contain any portion of $\mathcal{CH}(S_i)$ or Δ_2. Furthermore, every segment connecting w_i to a point on the face $u_i v_i v_{\text{cone}}$ is contained in R_1. Thus, w_i can see the face $u_i v_i v_{\text{cone}}$.

Connect w_i to v_{cone} (it is already connected to u_i and v_i) to form a tetrahedron Δ_1. The vertex v_{cone} is connected to all three vertices of T_i and is assigned to be w_{i+1}.

Lemma 3. *One of the faces f of $\mathcal{CH}(S_i)$ incident to e is seen by u_i or v_i.*

Proof. First consider Δ_1. The plane H_2 separates Δ_1 from $\mathcal{CH}(S_i)$ and Δ_2. So Δ_1 cannot obscure visibility between a vertex of Δ_2 and either face of $\mathcal{CH}(S_i)$ incident to e. Now refer to Figure 3. Consider rotating each face f of $\mathcal{CH}(S_i)$ incident to e away from $\mathcal{CH}(S_i)$ until a face of Δ_2 is intersected. These rotations are disjoint and both occur around the line containing e. So both cannot be greater than 180°. It is also not possible that both rotations are 180° due to the general-position assumption. Let f be a face that rotates less than 180°. The face f is seen by the vertices of the face of Δ_2 it intersects, including either u_i or v_i. Call the vertex u_i or v_i intersected q.

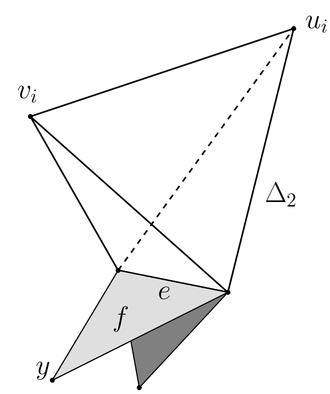

Figure 3: The scenario described in Lemma 3. Either u_i or v_i must see a face of $\mathcal{CH}(S_i)$ incident to e. In this case, v_i sees f. So $q = v_i$.

Connect q to y, the third vertex of this face (q is already connected to the other two vertices of f, the endpoints of e) to form tetrahedron Δ_3. The vertex y is connected to only a single vertex of T_i, and so is u_{i+1}.

Lemma 4. *The tetrahedra $\Delta_1, \Delta_2, \Delta_3$ form a three-tetrahedron tunnel in which u_i, v_i have degree 5 and 4 (not necessarily respectively), and w_i has degree 3.*

Proof. See Figure 4. The vertices u_i, v_i, w_i each have two edges connecting them to the other two vertices of T_i. Vertex w_i is also connected to v_{cone}, so it has degree 3. Vertices u_i and v_i are also connected to the endpoints of e. Vertex

q, which is either u_i or v_i, is also connected to y. Thus, one vertex from $\{u_i, v_i\}$ has degree 5, while the other has degree 4.

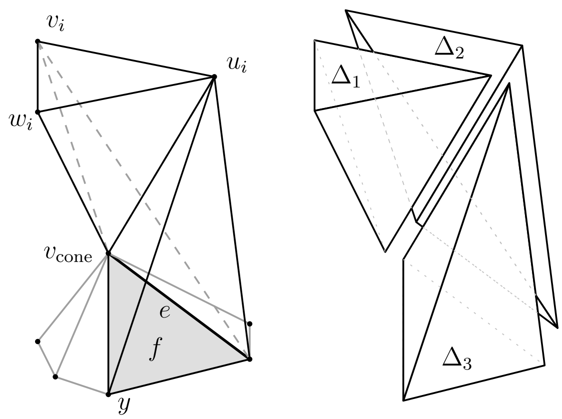

Figure 4: A complete tunnel and the three tetrahedra $\Delta_1, \Delta_2, \Delta_3$ composing it.

Once the tunnel between T_0 and T_1 is constructed, repeat the process to build a tunnel from T_1 to T_2, etc. When T_i is reached, such that S_i contains fewer than three points, construct a four- or five-vertex polyhedron.

5. Polyhedronization Properties

In this section we prove that the union of the constructed tunnels is a serpentine polyhedronization with vertex degrees bounded by 7, and that this bound is nearly optimal.

Lemma 5. *Tunnel interiors are disjoint.*

Proof. Consider the two tunnels between T_i, T_{i+1} and T_j, T_{j+1} for $j \neq i$. Without loss of generality, let $j > i$. All of the vertices of the tunnel between T_i, T_{i+1} are on the boundary or exterior of $\mathcal{CH}(S_i)$. Additionally, all of the vertices of the tunnel between T_j and T_{j+1} are on the boundary or interior of $\mathcal{CH}(S_i)$. Therefore, the two tunnels may only intersect on the boundary of $\mathcal{CH}(S_i)$. Hence, their interiors are disjoint.

Lemma 6. *The resulting polyhedronization of S is a serpentine polyhedron.*

Proof. Each tunnel is constructed of three tetrahedra that form a chain from T_i to T_{i+1} in the order $\Delta_1, \Delta_2, \Delta_3$. The tunnel between face triplets T_i and T_{i+1} shares T_i (resp., T_{i+1}) with the previous (resp., next) tunnels for $i > 0$. For $i = 0$ there is no previous tunnel and the tetrahedron with face T_0 is the first element of the dual chain. For the last tunnel, T_k, either a degenerate tunnel is formed with the remaining one, or two points or the last tetrahedron of T_k is the end of the chain. In the degenerate case, a face of T_k shares a face with the final degenerate tunnel. The final degenerate tunnel must be tetrahedralizable and have a dual chain since it is a polyhedron with four or five vertices. Therefore, in both cases the dual of the polyhedronization is a chain.

Theorem 7. *Every vertex in the polyhedronization of S has degree at most 7.*

PROOF. First consider the face triplets except the first and last. Each vertex is part of some triangle T_i and has two edges connecting it to the other vertices of T_i.

For a vertex u_i, one additional edge is connected to u_i in the tunnel between T_{i-1} and T_i, and at most three additional edges are connected to u_i in the tunnel between T_i and T_{i+1} (this occurs when $u_i = q$). So u_i has degree at most $1 + 2 + 3 = 6$. For a vertex v_i, two additional edges are connected to v_i in the tunnel between T_{i-1} and T_i, and at most three additional edges are connected to v_i in the tunnel between T_i and T_{i+1} (this occurs when $v_i = q$). So v_i has degree at most $2 + 2 + 3 = 7$. For a vertex w_i, three additional edges are connected to w_i in the tunnel between T_{i-1} and T_i, and one additional edge is connected to w_i in the tunnel between T_i and T_{i+1}. So w_i has degree at most $3 + 2 + 1 = 6$.

The face triplet T_1 is only attached to the triplet T_2, so every vertex of T_1 has degree at most 5. Let T_k be the last face triplet. Thus, $|S_k| \in \{0, 1, 2\}$. By construction, the vertices of T_k only share edges with vertices in $T_{k-1} \cup T_k \cup S_k$. This set has size at most $3+3+2 = 8$. So any vertex in T_k has degree at most 7. Similarly, the vertices of S_k only share edges with vertices in $T_k \cup S_k$, and so have degree at most 4.

Lemma 8. *Any polyhedronization of a set of at least 12 points in general position has a vertex of degree at least 6.*

PROOF. By Euler's formula, every polyhedron in general position with $|S|$ vertices has $3|S| - 6$ edges. Hence, the average degree of a vertex is $\frac{2(3|S|-6)}{|S|} = 6 - \frac{12}{|S|}$. Therefore, for $|S| > 12$, some vertex must have degree at least 6.

The algorithm described produces polyhedronizations with nearly optimal degree bounds. Indeed, Theorem 7 proved that the construction produces a polyhedronization with bounded degree 7, while by Lemma 8, every polyhedronization of an arbitrary number of points must have some vertex with degree at least 6.

6. Running Time

The efficiency of the algorithm described in Section 3 depends upon the data structure used to compute S_i and T_i at each step. The vertices of T_i can be found using four *gift-wrapping* queries: given a plane in space and a line contained in the plane, find the first point of the set intersected by the plane when it is rotated around the line in some direction. The first of these queries is used to find w_i, the second to find the other vertex of e, and the third and fourth to find the third vertices of the two faces adjacent to e on $\mathcal{CH}(S_{i-1})$. Computing $S_i = S_{i-1} \setminus T_i$ can be done by simply deleting the three points in T_i from S_{i-1}. Thus, we need a data structure that supports efficient gift-wrapping queries and deletions for point sets in 3D.

The randomized data structure described by Chan [3] supports gift-wrapping queries in $O(\log^2 n)$ worst-case time, deletions in $O(\log^6 n)$ expected amortized time, and initialization in $O(n \log^2 n)$ expected time, where n is the number of input points. Since we perform $O(n)$ gift-wrapping queries and deletions, the total time spent on these operations is $O(n \log^6 n)$ expected time. Note that the expectation is based exclusively on the random sampling of selected subsets of the input points which are used to construct the data structure suggested by Ramos [6], which in turn is used by Chan's data structure. Thus, the expected time is not dependent upon the input point set or the resulting polyhedronization. The space used by Chan's data structure is $O(n)$. So using this data structure in our algorithm yields a randomized algorithm with $O(n \log^6 n)$ expected time and $O(n)$ space.

It should be noted that the basic structure described by Chan supports extreme-point queries rather than gift-wrapping queries. Chan discusses the adaptation of the structure to support the latter type of queries, noting that the query algorithm can easily be modified "with no change at all in the description and correctness proof" by replacing the vertical ray-shooting data structure used with the non-vertical ray-shooting structure suggested by Dobkin and Kirkpatrick [4].

A deterministic algorithm can be achieved by replacing Chan's data structure with the hyperplane data structure described by Agarwal and Matoušek [2]. This algorithm requires $O(n^{1+\varepsilon})$ time and space, for any fixed $\varepsilon > 0$.

7. Conclusion

In this paper we have shown that in $\mathbb{R}^3$, fast construction of simple polyhedronizations with bounded constant degree is always possible for points in general position. If the requirement of general position is removed, then the same algorithm still produces a bounded-degree serpentine polyhedronization, but with the possibility that some of the tetrahedra are degenerate, i.e., have zero volume. It still remains open whether a polyhedronization with degree 6 always exists, or whether some point sets only admit polyhedronizations with degree 7 or higher. It is also unknown whether a similar algorithm could work in higher dimensions, and how the problem changes if polyhedronizations with positive genus (i.e. with holes) are permitted.

Acknowledgements

We wish to thank Joseph O'Rourke for a key insight related to segment-segment visibility. We also thank anonymous reviewers for helpful comments

and suggestions.

References

[1] P.K. Agarwal, F. Hurtado, G.T. Toussaint, and J. Trias, On polyhedra induced by point sets in space, *Discrete Applied Mathematics*, 156 (2008), 42–54.

[2] P.K. Agarwal and J. Matoušek, Dynamic half-space range reporting and its applications, *Algorithmica*, 13 (1995), 325–345.

[3] T.M. Chan, A dynamic data structure for 3-d convex hulls and 2-d nearest neighbor queries, *J. of the ACM*, 57 (2010), 1–16.

[4] D.P. Dobkin and D.G Kirkpatrick, Fast detection of polyhedral intersection, *Theoretical Computer Science*, 27 (1983), 241–253.

[5] N.J. Lennes, Theorems on the simple finite polygon and polyhedron, *American J. of Mathematics*, 33 (1911), 37–62.

[6] E.A. Ramos, On range reporting, ray shooting, and k-level construction, *Proc. 15th Ann. ACM Symp. on Computational Geometry*, Miami Beach, FL, 1999, 390–399.