Largest Subsets of Triangles in a Triangulation*

Boris Aronov† Marc van Kreveld‡ Maarten Löffler‡ Rodrigo I. Silveira‡

Abstract

Given a triangulation of n points, with some triangles marked "good", this paper discusses the problems of computing the largest-area connected set of good triangles that (i) is convex, (ii) is monotone, (iii) has a bounded total angular change, or (iv) has a bounded negative turning angle. The first, second, and fourth problems are solved in polynomial time, the third problem is NP-hard.

1 Introduction

One of the uses of triangulations is to subdivide a region into smaller elements that are easier to handle or to evaluate. If the triangulation is a terrain, we can evaluate the slope of each triangle, for instance. We may want to find a large subregion of the terrain where the slope does not exceed a given value. This eliminates some triangles, since slope is constant over a triangle. Similarly, we could specify that any north-facing triangle is bad. We consider computing a simple polygon of maximum area, formed as a union of good triangles and satisfying an additional shape constraint.

More formally, we address the following problems. Given a triangulation of a point set with any subset of triangles marked *bad* (the rest are *good*), find the largest-area simply-connected region comprised of good triangles, and such that:

(i) it is convex (see Figure 1(a)),
(ii) it is monotone in some direction,
(iii) the total (absolute) angular change is at most β,
(iv) or the maximum negative turning angle (defined below) is at least $-\gamma$ for some $\gamma < \pi$.

To define the maximum negative turning angle of a simple polygon P, let P be defined by the edges $\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_m$ in counterclockwise order, see Figure 1(b). We define the *turning angle* from edge $\vec{e}_i$ to edge $\vec{e}_j$ for $i \neq j$ as $\alpha(\vec{e}_i, \vec{e}_j) = \sum_{k=i}^{j-1} \measuredangle(\vec{e}_k, \vec{e}_{k+1})$ (indices wrap around, the sum is taken over at most $n-1$ angles), where $\measuredangle(\vec{e}_k, \vec{e}_{k+1})$ is the counterclockwise turn from $\vec{e}_k$ to $\vec{e}_{k+1}$ (a clockwise turn yields a negative $\measuredangle(\vec{e}_k, \vec{e}_{k+1})$). When $i = j$, we set $\alpha(\vec{e}_i, \vec{e}_i) = 0$. The *maximum negative turning angle* Φ of a simple polygon P is defined as $\min_{i,j} \alpha(\vec{e}_i, \vec{e}_j)$. Note that $\Phi \leq 0$, and that a simple polygon is convex if and only if $\Phi = 0$.

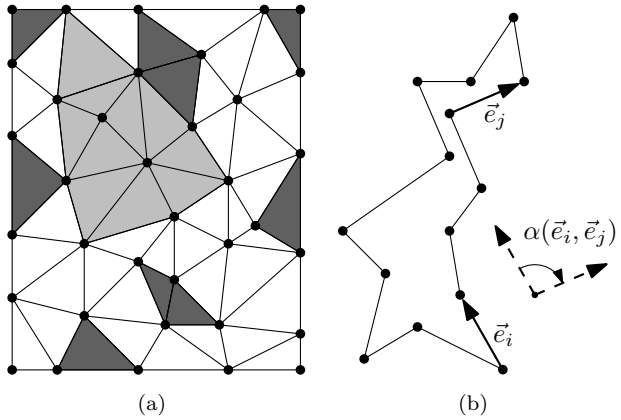

Figure 1: (a) The largest convex polygon (light gray) that is the union of triangles in a triangulation, while excluding bad (dark gray) triangles. (b) The maximum negative turning angle of the polygon is slightly less than $-\pi/2$.

We show that problem (i) can be solved in $O(n^2)$ time, problem (ii) in $O(n^2)$ time if the direction is given and in $O(n^3)$ time otherwise, problem (iv) in $O(n^6)$ time, and problem (iii) is NP-hard.

If the largest-area simple polygon we aim to compute were not constrained to be a union of good triangles, but just to be contained in that union, problem (i) would turn into the well-known *potato-peeling problem* that can be solved [5] in time $O(n^7)$ if the union of good triangles is simple, and in time $O(n^8)$ otherwise. Other papers present algorithms for the largest axis-parallel rectangle inside a polygon with or without holes [4, 7], the largest similar copy of a polygon inside a polygon [6, 10], or the largest axially symmetric polygon inside a convex polygon [3]. Turning angles are used for shape matching of polygons [2] and for certain optimal geometric tours [1, 8].

Henceforth, $\mathcal{T}$ is a triangulation of a set of n points.

*Work by B.A. has been supported by a grant from the U.S.-Israel Binational Science Foundation and by NSA MSP Grant H98230-06-1-0016. Partially funded by the Netherlands Organization for Scientific Research (NWO) under the project GOGO. Part of the work was performed while B.A. visited UU in January 2007.

†Polytechnic University, Brooklyn, NY 11201-3840, USA. http://cis.poly.edu/~aronov

‡Department of Computer Science, Utrecht University, NL. {marc,loffler,rodrigo}@cs.uu.nl.

2 Largest convex polygon

In this section we compute the largest-area convex polygon formed as a union of good triangles of $\mathcal{T}$. Without loss of generality, we assume that no two vertices of $\mathcal{T}$ have the same x-coordinate.

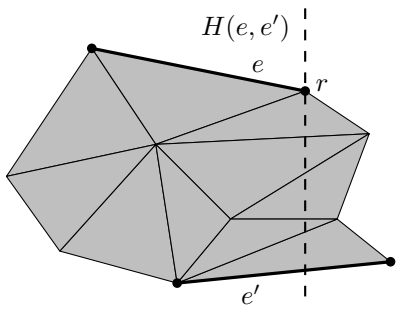

Figure 2: An admissible collection C of good triangles (gray) for the edge pair (e, e').

Given two edges e, e' of $\mathcal{T}$, we call a collection C of good triangles *admissible* for (e, e') if the following conditions hold (see Figure 2):

- $U := U(C) \cap H(e, e')$ is a convex polygon, where
  - $r = (r_x, r_y)$ is the leftmost of the right endpoints of e and e',
  - $H(e, e')$ is the half-plane $x \leq r_x$, and
  - $U(C)$ is the union of the triangles in C.
- The rightmost top edge of U lies on e.
- The rightmost bottom edge of U lies on e'.
- Every triangle of C intersects $H(e, e')$.

Now set $Q(e, e') := -\infty$ if (e, e') has no admissible collection of good triangles, otherwise $Q(e, e')$ is defined to be the largest area$(U(C) \cap H(e, e'))$ achieved by (the union of) such a collection C, restricted to $H(e, e')$. We will compute $Q(e, e')$ by dynamic programming. Notice that, if e and e' share their right endpoints, $Q(e, e')$ is precisely the area of the largest convex polygon consisting of good triangles and having e and e' as its rightmost top and bottom edges, respectively. Hence, examining $Q(e, e')$ over all pairs of edges with the same right endpoint, one can find the area of the desired largest-area convex polygon. The polygon itself can be extracted by standard dynamic programming methods. Henceforth we focus on computing the quantity $Q(e, e')$ for all pairs (e, e') of edges.

We call a pair of edges (e, e') *promising* if e lies above e' and the trapezoid $T(e, e')$, defined as the locus of points vertically below e and above e', does not contain an interior point of a bad triangle. Clearly, any pair that is not *promising* has Q value of $-\infty$ (notice that the converse does not hold!). We will first give the sketch of an algorithm to identify all the promising pairs, before explaining how to compute Q values for all pairs of edges.

For a given pair of edges (e, e'), with e above e', let $X(e, e')$ be the common interval of their x values. It must be non-empty for promising pairs.

We sweep $\mathcal{T}$ by a vertical sweep-line ℓ moving from left to right and stopping at each vertex. At a given point of the sweep, when we are at a sweep-line $\ell : x = A$, all the promising pairs with $X(e, e')$ to the left of A have already been discovered. All the ones with $A \in X(e, e')$ and with no interior point of a bad triangle in $T(e, e')$ to the left of ℓ are currently being maintained.

Along the sweep, we maintain the following information. For each edge e meeting the sweep-line, we keep two sorted lists $E_a(e)$ and $E_b(e)$. $E_b(e)$ contains all the edges e' meeting ℓ below e, which have their leftmost endpoint to the left of that of e and such that the intersection of $T(e, e')$ with the halfplane $x < A$ is free of interior points of bad triangles. $E_b(e)$ is stored sorted, in downward direction along ℓ. $E_a(e)$ is analogous but for the edges lying above e. In addition to these two lists, we also keep a list $LL(e)$ that contains all the edges e' such that $E_a(e')$ or $E_b(e')$ contains e. The need for these cross-references will become clear later.

We discern three different types of events. When a new edge e begins, we walk along ℓ upwards collecting all the edges crossing ℓ above e, and adding them to $E_a(e)$ (and adding the corresponding cross-references to the LL lists). This continues until a bad triangle is reached. The same is done downwards to fill in $E_b(e)$. We spend constant time per added pair. When an edge e ends, we use $E_a(e)$, $E_b(e)$ and $LL(e)$ to generate all the promising pairs that involve e and update the lists of remaining edges. This takes constant time per pair as well. The last type of event occurs when a bad triangle B begins, at some vertex v. Intuitively we want to remove all pairs (e, e') for which $T(e, e')$ contains v. Each such no-longer-promising pair has a top edge e above v and a bottom edge e' below v. We can assume that no other bad triangle B' interferes with (e, e'). We walk up ℓ, from v, looking for candidates e, until we hit a bad triangle. For each e found, we scan its list $E_b(e)$, from the tail backwards. If the last element e' is above v, the entire list is safe. If it is below v, the pair (e, e') is not promising, and we drop e' from $E_b(e)$ (and e from $LL(e')$), continuing upwards until the first e' above v. This is repeated until all non-promising pairs have been dropped. Similarly, we walk down ℓ from v and scan $E_a(e)$. Each pair dropped is processed only once, so we do at most $O(n^2)$ work overall. It follows that all the promising pairs can be computed in total $O(n^2)$ time.

Having established how to identify promising pairs (e, e'), we explain how to compute the remaining values $Q(e, e')$. Conceptually, we order pairs (e, e') by their r_x value (the x-coordinate of the leftmost of their right

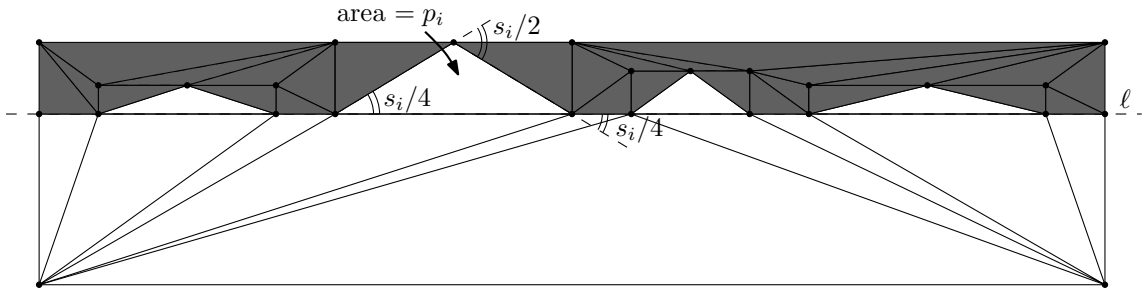

Figure 3: Reduction of knapsack to maximum area subtriangulation with bounded total angular change.

endpoints) and compute the values for later pairs from the values of earlier ones. Practically, we proceed as follows: For a promising pair (e, e') sharing their left endpoint, let $Q(e, e')$ be the area of the triangle delimited by e, e' and the vertical line through the leftmost of their right endpoints. For every vertex v, with e_1 ending at v, e_2 starting at v, and such that the slope of e_2 is smaller than that of e_1, we set $Q(e_2, e') = Q(e_1, e') + A$, for all pairs (e_1, e'), (e_2, e') of promising pairs, where A is the area of $T(e_2, e')$; since (e_2, e') is promising the whole area added is good. Similar actions are taken for promising pairs where the bottom edge changes at v.

It is easy to organize the computation so that all pairs sharing leftmost or rightmost endpoints are processed in total time proportional to the square of the degree of their common vertex and the remaining updates can be made in time proportional to n times the degree of the vertex. This leads to quadratic time.

Theorem 1 *Given a triangulation of n points, with any subset of triangles marked as good, a maximum-area convex polygon that is the union of good triangles can be computed in $O(n^2)$ time.*

Turning to problem (ii), we note that the algorithm given above can easily be adapted to compute the largest-area polygon that is monotone in a given direction. We omit the details due to space limitations.

Theorem 2 *Given a triangulation of n points, with any subset of triangles marked as good, a maximum-area monotone polygon that is the union of good triangles can be computed in $O(n^2)$ time if the direction is given, or in $O(n^3)$ time otherwise.*

3 Largest-area bounded-angular-change region

We now consider the problem of finding the largest-area simple polygon comprised of good triangles, so that the total angular change of its boundary is at most some constant. We show that this problem is NP-hard for a total angular change of, say, 3π. The reduction is from the NP-hard problem KNAPSACK [9]: Given a set of n pairs $(s_1, p_1), \ldots, (s_n, p_n)$, where s_i is the size of the i^{th} item and p_i is its profit, and a maximum allowed size S, select the subset of total size at most S maximizing total profit . Without loss of generality, we assume that the sizes $s_1, \ldots, s_n$ and S have been scaled so that $S = \pi$. We construct a triangulation as follows, see Figure 3.

Take a pair (s_i, p_i). If $s_i > S = \pi$ then we discard it. Otherwise we construct a rectangle R_i of height 1 and width $2/\tan(s_i/4)$, and put a fifth vertex m_i in the middle of its top side. Triangulate the rectangle in a star fashion from m_i. Scale the construction so that the area of R_i is $2p_i$. The total angular change of a path that arrives horizontally from the left at the lower left corner of R_i, goes diagonally up to m_i, and then diagonally down to the lower right corner of R_i, and then goes horizontally to the right, is s_i, and the area in R_i and below the path is p_i. This corresponds to choosing the i^{th} item in the knapsack: the profit (area) is p_i and the cost (extra angular change) is s_i. Now we combine the rectangles so that all lower sides lie on a common horizontal line ℓ and they are sufficiently spaced; their order is irrelevant. All triangles above ℓ, except the largest one in each rectangle, are marked bad. Below the line ℓ we make large good triangles whose union is a large rectangle $\bar{R}$ with the following properties:

- The top side of $\bar{R}$ lies on ℓ.
- The upper left corner of $\bar{R}$ is strictly to the left of all the rectangles R_i above ℓ.
- The upper right corner of $\bar{R}$ is strictly to the right of all the rectangles R_i above ℓ.
- Every triangle in a triangulation of $\bar{R}$ has area greater than $\sum_{i=1}^{n} p_i$ (note that $\bar{R}$ has all bottom vertices of rectangles R_i on its top side).

All four conditions are easily satisfied. By construction, a maximum area solution will contain at least all triangles inside $\bar{R}$, because leaving one out cannot be compensated by all remaining good triangles, namely those that lie in R_i. The total angular change of $\bar{R}$ is exactly 2π. We can choose each of the remaining good triangles independently. A good triangle lying in R_i has cost of s_i and area p_i. Hence, the subset of items with maximum profit in the knapsack with total size at most π corresponds precisely to the subset of extra triangles

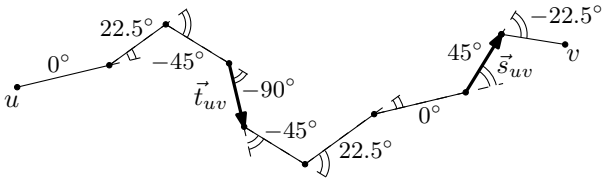

Figure 4: Path from u to v showing the turning angle of each edge, with respect to the first edge of the path. The extreme edges are the ones with the maximum and minimum turning angle.

that together with triangles of $\bar{R}$ form the largest-area simple polygon with angular change bounded by 3π.

Theorem 3 *Given a triangulation of n points, with a subset of the triangles marked as good, computing the maximum-area polygon, comprised of good triangles, with total angular change at most 3π is NP-hard.*

4 Bounded maximum negative turning angle

We want to find the largest-area simple polygon P that is a union of good triangles, with the constraint that the maximum negative turning angle Φ of its the boundary is bounded by some constant $\gamma \geq 0$; specifically we require that $\Phi := \min_{i,j} \alpha(\vec{e}_i, \vec{e}_j) > -\gamma$.

For a given path from vertex u to vertex v, we define the *extreme edges* $\vec{s}_{uv}$ and $\vec{t}_{uv}$ as follows: $\vec{s}_{uv}$ is the edge of the path with the largest turning angle with respect to the first edge of the path, whereas $\vec{t}_{uv}$ is the edge with the smallest one. See Figure 4 for an example.

It will be useful to redefine our angle constraint in terms of extreme edges. Given a polygon $P = \{\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_m\}$, we can split it into a path at any vertex v on its boundary. Such a path, counterclockwise from v to v around the polygon, will have a pair of extreme edges $\vec{s}_{vv}$ and $\vec{t}_{vv}$. It can be shown that $\min_{i,j} \alpha(\vec{e}_i, \vec{e}_j) > -\gamma$ if and only if, for every vertex v of P $\alpha(\vec{t}_{vv}, \vec{s}_{vv}) > -\gamma$. The subregion we are looking for is a simple polygon P. Our dynamic programming algorithm defines a value function Q which assigns to each subproblem the value of an optimal solution of the subproblem. A subproblem is defined on a pair of vertices, a pair of edges that are incident to those vertices, and a pair of extreme directions. Let $(v, u, \vec{e}_{v,\text{out}}, \vec{e}_{u,\text{in}}, \sigma, \tau)$ be such an instance. The subproblem is to find a path from v to u, that uses $\vec{e}_{v,\text{out}}$ as the first edge and $\vec{e}_{u,\text{in}}$ as the last edge, such that the directions of the extreme edges $\vec{s}_{vu}$ and $\vec{t}_{vu}$ of the path are bounded by σ and τ. The polygon consisting of this path and edge $\vec{uv}$ should be a simple polygon containing no (parts of) bad triangles and have optimal area. Note that $\vec{uv}$ is the only edge of this polygon that is not necessarily an edge of the triangulation.

The dynamic program is based on the fact that the optimal polygon P^* has a triangulation (not to be confused with the given triangulation $\mathcal{T}$ of the whole region), and we can use it to recursively define the solution to an instance in terms of the solutions to smaller instances. In particular, the path from v to u in the instance $S = (v, u, \vec{e}_{v,\text{out}}, \vec{e}_{u,\text{in}}, \sigma, \tau)$ is composed of a path from v to some vertex w and a path from w to the vertex u, and such that uw and vw are diagonals of a triangulation of the solution to the instance S. We can maximize over all vertices w, the edges incident to w that are used on the paths, and the way the bounds of the extreme directions may occur in the subproblems, as long as the solutions do not contain bad triangles. The details are rather involved and deferred to the full paper.

As for a time analysis, we have a table with $O(n^4)$ entries: there are a linear number of possibilities for the pair $(v, \vec{e}_{v,\text{out}})$, a linear number for the pair $(u, \vec{e}_{u,\text{in}})$, a linear number for σ, and a linear number for τ. We can fill any entry in quadratic time, so we need $O(n^6)$ time in total.

References

[1] A. Aggarwal, D. Coppersmith, S. Khanna, R. Motwani, and B. Schieber. The angular-metric traveling salesman problem. *SIAM J. Comput.*, 29:697–711, 1999.

[2] E. M. Arkin, L. P. Chew, D. P. Huttenlocher, K. Kedem, and J. S. B. Mitchell. An efficiently computable metric for comparing polygonal shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(3):209–216, 1991.

[3] G. Barequet and V. Rogol. Maximizing the area of an axis-symmetric polygon inscribed by a convex polygon. In *Proc. 16th Canadian Conference on Computational Geometry (CCCG'04)*, pages 128–131, 2004.

[4] R. Boland and J. Urrutia. Finding the largest axis aligned rectangle in a polygon in $o(n \log n)$ time. In *Proc. 13th Can. Conf. Comp. Geom.*, pages 41–44, 2001.

[5] J. S. Chang and C. K. Yap. A polynomial solution for the potato-peeling problem. *Discrete Comput. Geom.*, 1:155–182, 1986.

[6] B. Chazelle. The polygon containment problem. In F. P. Preparata, editor, *Computational Geometry*, volume 1 of *Adv. Comput. Res.*, pages 1–33. JAI Press, Greenwich, Conn., 1983.

[7] K. Daniels, V. Milenkovic, and D. Roth. Finding the maximum area axis-parallel rectangle in a polygon. *Comput. Geom. Theory Appl.*, 7:125–148, 1997.

[8] S. P. Fekete and G. J. Woeginger. Angle-restricted tours in the plane. *Comp. Geom. Th. Appl.*, 8:195–218, 1997.

[9] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, NY, 1979.

[10] M. Sharir and S. Toledo. Extremal polygon containment problems. *Comp. Geom. Th. Appl.*, 4:99–118, 1994.