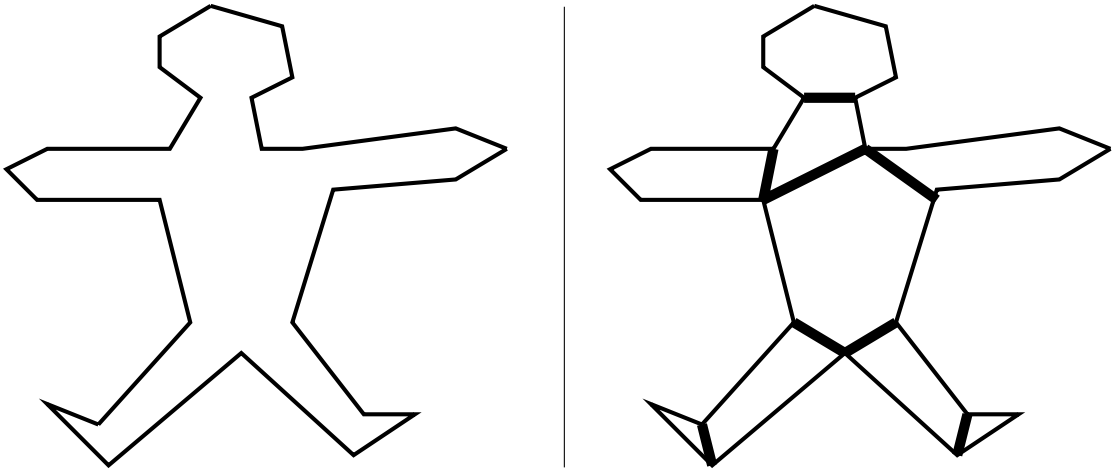

INPUT OUTPUT

17.11 Polygon Partitioning

Input description: A polygon or polyhedron P.

Problem description: Partition P into a small number of simple (typically convex) pieces.

Discussion: Polygon partitioning is an important preprocessing step for many geometric algorithms, because geometric problems tend to be simpler on convex objects than on nonconvex ones. It is often easier to work with a small number of convex pieces than with a single nonconvex polygon.

Several flavors of polygon partitioning arise, depending upon the particular application:

- *Should all the pieces be triangles?* – Triangulation is the mother of all polygon partitioning problems, where we partition the interior of the polygon completely into triangles. Triangles are convex and have only three sides, making them the most elementary possible polygon.

  All triangulations of an n-vertex polygon contain exactly $n - 2$ triangles. Therefore, triangulation cannot be the answer if we seek a small number of convex pieces. A "nice" triangulation is judged by the shape of the triangles, not the count. See Section 17.3 (page 572) for a thorough discussion of triangulation.

- *Do I want to cover or partition my polygon?* – *Partitioning* a polygon means completely dividing the interior into nonoverlapping pieces. *Covering* a polygon means that our decomposition is permitted to contain mutually overlapping pieces. Both can be useful in different situations. In decomposing a complicated query polygon in preparation for a range search (Section 17.6 (page 584)), we seek a partitioning, so that each point we locate occurs in exactly one piece. In decomposing a polygon for painting purposes, a covering suffices, since there is no difficulty with filling in a region twice. We will concentrate here on partitioning, since it is simpler to do right, and any application needing a covering will accept a partitioning. The only drawback is that partitions can be larger than coverings.

- *Am I allowed to add extra vertices?* – A final issue is whether we are allowed to add Steiner vertices to the polygon, either by splitting edges or adding interior points. Otherwise, we are restricted to adding chords between two existing vertices. The former may result in a smaller number of pieces, at the cost of more complicated algorithms and perhaps messier results.

The Hertel-Mehlhorn heuristic for convex decomposition using diagonals is simple and efficient. It starts with an arbitrary triangulation of the polygon and then deletes any chord that leaves only convex pieces. A chord deletion creates a nonconvex piece only if it creates an internal angle that is greater than 180 degrees. The decision of whether such an angle will be created can be made locally from the chords and edges surrounding the chord, in constant time. The result always contains no more than four times the minimum number of convex pieces.

I recommend using this heuristic unless it is critical for you to minimize the number of pieces. By experimenting with different triangulations and various deletion orders, you may be able to obtain somewhat better decompositions.

Dynamic programming may be used to find the absolute minimum number of diagonals used in the decomposition. The simplest implementation, which maintains the number of pieces for all $O(n^2)$ subpolygons split by a chord, runs in $O(n^4)$. Faster algorithms use fancier data structures, running in $O(n+r^2 \min(r^2, n))$ time, where r is the number of reflex vertices. An $O(n^3)$ algorithm that further reduces the number of pieces by adding interior vertices is cited below, although it is complex and presumably difficult to implement.

An alternate decomposition problem partitions polygons into *monotone* pieces. The vertices of a y-monotone polygon can be divided into two chains such that any horizontal line intersects either chain at most once.

Implementations: Many triangulation codes start by finding a trapezoidal or monotone decomposition of polygons. Further, a triangulation is a simple form of convex decomposition. Check out the codes in Section 17.3 (page 572) as a starting point.

CGAL (*www.cgal.org*) contains a polygon-partitioning library that includes (1) the Hertel-Mehlhorn heuristic for partitioning a polygon into convex pieces,

(2) finding a optimal convex partitioning using the $O(n^4)$ dynamic programming algorithm, and (3) an $O(n \log n)$ sweepline heuristic for partitioning into monotone polygons.

A triangulation code of particular relevance here is GEOMPACK—a suite of Fortran 77 codes by Barry Joe for 2- and 3-dimensional triangulation and convex decomposition problems. In particular, it does both Delaunay triangulation and convex decompositions of polygonal and polyhedral regions, as well as arbitrary-dimensional Delaunay triangulations.

Notes: Recent survey articles on polygon partitioning include [Kei00, OS04]. Keil and Sack [KS85] give an excellent survey on what is known about partitioning and covering polygons. Expositions on the Hertel-Mehlhorn heuristic [HM83] include [O'R01]. The $O(n+r^2 \min(r^2, n))$ dynamic programming algorithm for minimum convex decomposition using diagonals is due to Keil and Snoeyink [KS02]. The $O(r^3 + n)$ algorithm minimizing the number of convex pieces with Steiner points appears in [CD85]. Lien and Amato [LA06] provide an efficient heuristic for decomposing polygons with holes into "almost convex" polygons in $O(nr)$ time, with later work generalizing this to polyhedra.

Art gallery problems are an interesting topic related to polygon covering, where we seek to position the minimum number of guards in a given polygon such that every point in the interior of the polygon is watched by at least one guard. This corresponds to covering the polygon with a minimum number of star-shaped polygons. O'Rourke [O'R87] is a beautiful (although sadly out of print) book that presents the art gallery problem and its many variations.

Related Problems: Triangulation (see page 572), set cover (see page 621).