23: Computational Geometry - Polygon Triangulation

CS 473u - Algorithms - Spring 2005

April 15, 2005

1 The Art Gallery Problem

> 04-Feb-2003
> Stephane Breitwieser, a 31-year-old waiter, admitted the theft of 69 works of art in Switzerland and faces up to seven and a half years in prison.
> A total of 240 items were taken during visits to museums in Austria, Belgium, Denmark, France, Germany, the Netherlands and Switzerland during a six-year crime spree which stunned the art world.

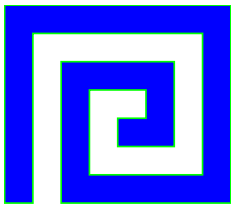
Given an art gallery, you would like to place minimum number of guards so that they see the whole art gallery.

Q: Compute the minimum number of guards needed, and where to place them.

1.1 Polygons

Definition 1.1 A *segment* (or line segment) is a closed subset of a line contained between two points a and b, which are its *endpoints*.

Definition 1.2 A ***polygon*** is a region of the plane bounded by a finite collection of line segments forming a simple close curve.



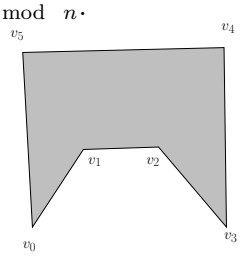
Formally, we are given n vertices (i.e., points) $v_0, v_1, \ldots, v_{n-1}$, the chain formed by $v_0v_1v_2 \ldots v_{n-1}$ is a polygon iff

1. The segments $s_0 = v_0v_1$,$\ldots$,$s_{n-2} = v_{n-2}v_{n-1}$, $s_{n-1} = v_{n-1}v_0$ are disjoint in their interior.

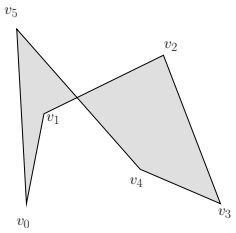
2. Consecutive segments intersect only in their endpoints. Namely $s_i \cap s_{i+1} = v_{i+1}$ for $i = 0, \ldots, n-2$ and $s_{n-1} \cap s_0 = v_0$

3. Non adjacent segments do not intersect $s_i \cap s_j = \phi$ for $j > i+1$.

We work $\mod n$. Namely $v_i = v_{i \mod n}$.

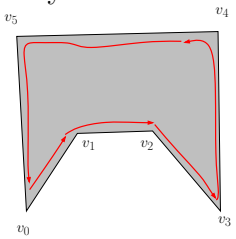


This polygon is simple because it does not self intersect. If it does self intersect, it is not simple. Example:



Theorem 1.3 *(Jordan Curve Theorem) Every simple closed plane curve divides the plane into two components.*

P - denote a polygon
∂P - boundary of a polygon.
$\partial P \subseteq P$ - polygon is closed and contains its boundary.
Vertices will usually be specified in a counterclockwise direction. So interior of polygon is to your left, as you go on the boundary...



1.1.1 Visibility

Definition 1.4 A point x sees points y inside P if $xy \subseteq P$.

Q: How many points do we need to place inside P so that they see all the points of P?

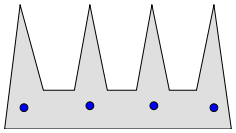
Definition 1.5 A ***guard*** is just a point.
A set of guards G ***covers*** a polygon P if for any point $x \in P$ there is a guard $y \in G$ such that x sees y.
$g(P)$ - minimum number of guards guarding P.
= cardinality of smallest set that covers P.

IP_n - set of all simple polygons with n vertices.

$G(n) = \max_{P \in IP_n} g(P)$ - maximum number of guards needed to guard a simple polygon with n vertices.

Lemma 1.6 $G(n) \geq n/3$.

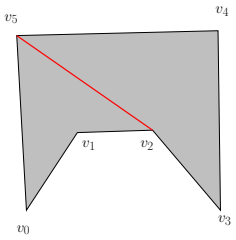
Proof: Consider the example:



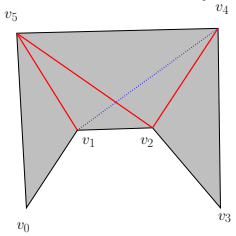
Clearly, we need four guards to guard this polygon that has 12 vertices. Furthermore, we can introduce as many teeth as want to this polygon. Every tooth would require one additional guard, and would require another guard. So, number of guards is $n/3$. ■

1.1.2 Diagonals and Triangulations.

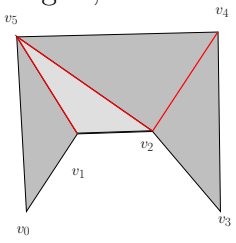
Definition 1.7 A ***diagonal*** of a polygon P is a segment connecting two vertices of P that see each other (in fact, we are strict, and demand that no vertex of P would lie in the interior of the segment).



Let us insert diagonals into P repeatedly, such that the diagonals do not cross (i.e., intersect in their interior), until we can not do it any more.

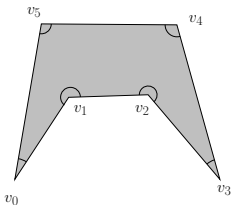


This form a partition of P into triangles, known as a ***triangulation***.



Lemma 1.8 *In every triangulation of a polygon with n vertices, there are $n-2$ triangles.*

Definition 1.9 A vertex is ***strictly convex*** if the interior angle in the vertex is strictly smaller than π.

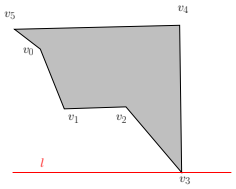


v_0, v_3, v_4, v_5 are strictly convex.

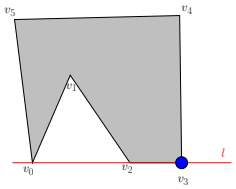
Lemma 1.10 *Every polygon must have at least one strictly convex vertex.*

Proof: Let l be the lowest horizontal line with non-empty intersection with P. Let v be the rightmost vertex of P on l. Clearly, v is strictly convex.

Easy case:



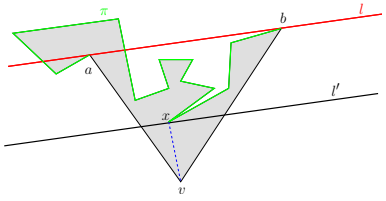
Harder case:



■

Lemma 1.11 *Every polygon with more than* 3 *vertices, has a diagonal.*

Proof: Let P be a polygon, and let v be a strictly convex vertex of P. Let a and b be the two vertices of P adjacent to v. Let l be the line passing through ab. If ab is a diagonal, then we are done.

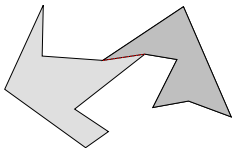


Otherwise, consider the triangle $T = \triangle avb$, and observe that the boundary of P must intersect the interior of T. Let π denote the boundary of P inside T, and drag l toward v in parallel, as long as it intersects π. Let l' be the resulting line. Clearly, $\pi \cap l'$ is a vertex x of P that sees v. Thus we found our diagonal. ■

Theorem 1.12 *(Triangulation) Every polygon P of n vertices may be partitioned into triangles by the addition of (zero or more) diagonals.*

Proof: By induction. For $n = 3$ we are done.

Otherwise, for $n > 3$, find a diagonal $d = ab$ in P. The diagonal d partition P into two polygons that have less than $n - 1$ vertices. As such, by induction, they can be partitioned into triangles. Resulting in a partition of the original polygon into triangles.



■

1.1.3 Properties of triangulations

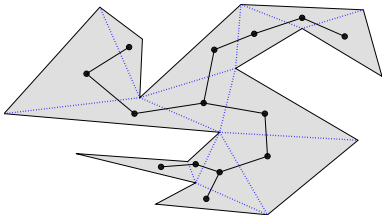
Lemma 1.13 *(Number of Diagonals) Every triangulation of a polygon P of n vertices uses n-3 diagonals and consists of $n-2$ triangles.*

Proof: By induction (exercise). ■

Lemma 1.14 *(Sum of angles) The sum of the internal angles of a polygon of n vertices is $(n-2)\pi$.*

Proof: Exercise. ■

Definition 1.15 The dual T of a triangulation of a polygon is a graph with a node associated with each triangle and an arc between two nodes iff their triangles share an edge.

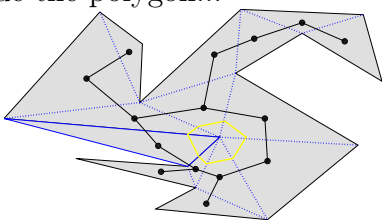


Lemma 1.16 *The dual T of a triangulation of a polygon is a tree with a vertex degree at most three.*

Proof: The degree three is immediate from the fact that every triangle have three sides.

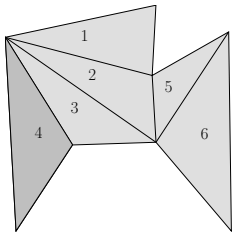
If there is a cycle C in T it is easy to verify that...

There must be a vertex inside the polygon...



■

Definition 1.17 Three consecutive vertices of a polygon a, b, c, form an ear of the polygon if ac is a diagonal; b is the ear tip. Two ears are non-overlapping if their triangle interiors are disjoint.



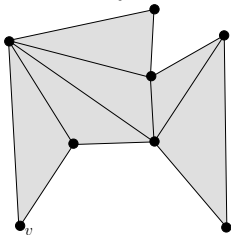
Theorem 1.18 (Meister's Two-Ears theorem) *Every polygon of n vertices has at least two non-overlapping ears.*

Proof: A leaf node in a dual (of the triangulation) corresponds to an ear. A tree of two or more nodes must have at least two leaves. ■

We can interpret a triangulation of a polygon, as a graph. In particular, let $D(P)$ denote this graph.

Theorem 1.19 *The graph $D(P)$ is three colorable.*

Proof: By the two-ears theorem, there is always a ear in $D(P)$. Let v be the trip of the ear.



By induction, $D(P) \setminus \{v\}$ is 3-colorable, and since v has degree 2, we can always find a color for it out of the three that keep the coloring legal. ■

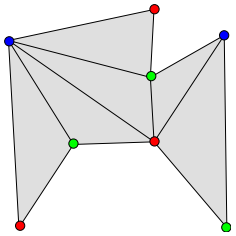
Well, why is this interesting?

If we want to guard P, effectively, we can place guards in the vertices. A guard would see all the triangles adjacent to this vertex.

Theorem 1.20 *Every simple polygon P with n vertices, can be guarded using $\leq n/3$ guards. Formally, $G(n) \leq n/3$.*

Proof: Compute the triangulation of P, and compute a 3 coloring for P. Clearly, every color appears in any triangle, and as such we can use all the vertices that have the same color as a guarding set. Now, one of the tree colors, appears in at most $n/3$ vertices. Use this set to guard P.

Example:



■

2 Computing a triangulation

How to compute a triangulation efficiently?

2.1 First Solution - insert diagonals

Given a segment, we can check if it is a diagonal in linear time.

How?

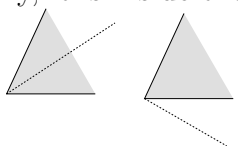
Well, we assume that we can check if two segments intersect in constant time.

Which is true. Why?

Constant size problem.

So, to check if a segment is a diagonal of a polygon, just check:

1. Does it intersects the boundary of the polygon. Linear time by scanning all the edges of the polygon.
2. Its two endpoints belong to the polygon (easy).
3. It lies inside the polygon. Locally, it is inside the polygon near the endpoints.



(Again, this can be decided in constant time)

Thus, we can decide in linear time if a segment is a diagonal.

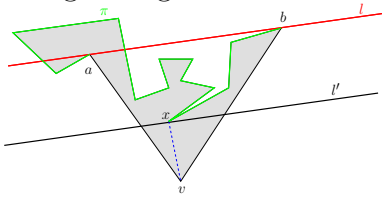
There are $O(n^2)$ candidates to be diagonals, so finding a diagonal would take us $O(n^3)$ time. We need to find $n-3$ diagonals to triangulate a polygon. Overall, we get

Lemma 2.1 *Given a polygon P with n vertices, it can be triangulated in $O(n^4)$.*

Can we do better?

2.2 Solution 2

Remember, that the proof for finding a diagonal was constructive:



1. Find lowest vertex v in a polygon (linear time).
2. Find adjacent two vertices a, b (constant time).
3. Scan polygon to find l-closest vertex x that is inside the $\triangle avb$. (linear time)

4. vx is the required diagonal.

Lemma 2.2 *One can compute a diagonal of a polygon in linear time.*

Lemma 2.3 *One can triangulate a polygon with n vertices in $O(n^2)$ time.*

It turns out that one can triangulate a polygon in $O(n \log n)$ time. We will describe this algorithm in the next lecture.

3 Implementation Details

3.1 Representation

It is very natural to use OOP for geometry. Nevertheless, we still need to understand how to represent objects.

So,

1. Point - a pair (x, y) of two real numbers defining the coordinates of the point.
2. Segment - a pair of points ab.
3. Polygon - a sequence n of points.

The problem is that to implement the algorithms described above, we need to be able to answer the following questions quickly:

1. Are two points identical?
2. Are three points collinear?
3. isIntersect(s_1, s_2) - do two segments intersect?
4. intersection(s_1, s_2) - compute intersection of two segments.
5. etc...

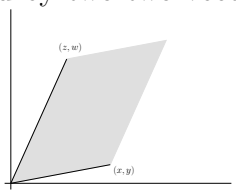
The question is how to implement those **geometric primitives**?

Consider two points $v = (x, y)$ and $u = (z, w)$, what is the interpetation of the quantity:

$$L(v, u) = Det\begin{pmatrix} x & y \\ z & w \end{pmatrix} = x * w - y * z$$

?

Area of the parallelogram spanned by two two vectors:

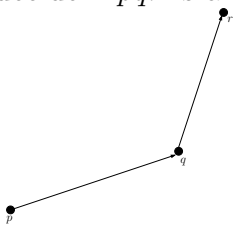


But, in fact:

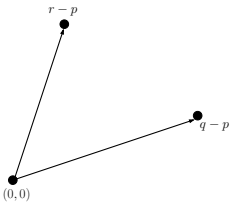
$$L(u, v) = Det\begin{pmatrix} z & w \\ x & y \end{pmatrix} = ?$$

Lemma 3.1 *If $L(v,u) > 0$ then if you stand in the origin and look in the direction of v and then turn to u, then this is a left turn. If $L(u,v) = 0$ then u and v are coolinear with the origin. If $L(v,u) < 0$ then this is a right turn.*

Given three points p, q, r how to decide if pqr is a left turn or a right turn?

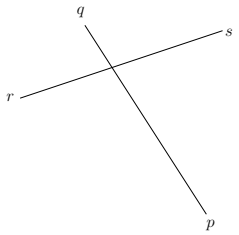


Idea: subtract p from q and r and see if the resulting two vectors are a left turn:



$\text{isLeftTurn}(p,q,r) = \text{is}\circlearrowleft(p,q,r) \leftarrow (L(q-p, r-p) > 0)$

Q: Given two segments (p,q) and (r,s) how to decide if they intersect?
Hint: Use isLeftTurn.



Define

$$\begin{aligned}\text{DiffSides}(pq,r,s) &= \Big[\text{is}\circlearrowleft(p,q,r) \wedge \text{is}\circlearrowright(p,q,s)\Big] \\ &\vee \Big[\text{is}\circlearrowright(p,q,r) \wedge \text{is}\circlearrowleft(p,q,s)\Big]\end{aligned}$$

to decide if the points r and s are on different sides of the line defined by pq.

Clearly, pq intersects rs iff $\text{DiffSides}(pq,r,s) \wedge \text{DiffSides}(rs,p,q)$. Not that everything boils down to computing several determinants.