Triangulating a Polygon

Recall: PSLG = Planar Straightline graph.

Def (Simple) Polygonal chain is a PSLG consisting of a simple cycle P.

Claim A Polygonal chain has a unique interior.

Def Polygon is Polygonal chain + interior

Triangulation: Addition of seg so that
1) Still PSLG
2) Interior is decomposed into triangles
Then every simple polygon can be triangulated.

Proof: Induct on # of edges or points \( n \)

Base case: \( n = 3 \) \( \triangle \) done

Assume no \( 180^\circ \) angles \( \quad \)

\( n > 3 \) true for \( m < n \)

Let \( v \) be left most point with \( m \geq 2 \) \& \( u \)

1) Case 1 \( \text{seg} = [w, u] \) is interior

\( w \) gets two Poly \( P_1 \left| = 3 \right. \quad P_2 \left| = n - 1 \right. 

2) Case 2 3 point \( v \) interior to \( \text{Tri} = [v, v', w] \)

Let \( v' \) be left most such point

\( \text{seg} = [v, v'] \) is interior

\( P_1 \leq n \) \& \( P_2 ) \leq n \)
Thm Not Every simple polygonal surface in 3D can be decomposed in Tetrahedra

By Contra!

Consider Tet with faces B

Missing vertex in X or Y not Z

Not X since deg [X, Y] undercut

In general polyh in NP-Hard
Guarding A Polygon

Input: Polygon $P$

Output: locations $p_1, \ldots, p_k \in P$ (guards)

1) Guards cover $P$

2) $k$ small.

Thm A polygon $P$ with $n$ vertices

$\frac{2n}{3}$ guards suffice and maybe necessary.

Necessary:

$P = \begin{array}{c}
\text{n/3 prongs} \\
|P| = n \quad \text{Needs a guard per prong}
\end{array}$
$\ell/3$-guards Alg $(P)$

1) Two $P \bar{P}$
2) 3-color $\bar{P}$
   a) Construct geometric dual $T$ (a tree)
   b) 3-color $\bar{P}$ by traversing trees in an inorder fashion.
3) Pick least used color.

Only non-linear time step is 1)
2D-Algorithm

Proof \( \Rightarrow O(n^2) \)

Known: \( O(n) \) Chazelle

Today: \( O(n \log n) \) (sweep line)

This Class: \( O(n \log^* n) \) Seidel (incremental, randomized)

Def \( \log^* n = \min_k \log \log \ldots \log n \) \( n \leq 1 \)

\( \text{Prob}^5 \) Give a \( O(n) \) time alg to determine which side of
an edge in interior/exterior of a simple poly.

\( \text{Prob}^5 \) test \( P \subseteq \text{Int}(P) \) in \( O(n) \) time.

3.14 \( O(n \log n) \) OK

\( O(n) \)?

\( \text{Trap} \Rightarrow T \)
Step 1: Partition into Monotone Polygons

Def: A y-monotone if
- Every horizontal line l
  - l ∩ P is connected or empty

Alg: Type: line sweep $O(n \log n)$ time

Def:
- $\square$ = start vertex
- $\blacksquare$ = end
- $\bigcirc$ = edge
- $\triangle$ = split
- $\triangledown$ = merge
Claim 3.4  Pro v-motion iff no split or merge vertices

(⇒) easy

(⇐)

not move ⇒ 3 split or merge

⇒ 2 with at least 1 internal

⇒ 2" with one False

⇒ 2" sum lemma 2 to 1

Alg Line Sweep

1) Events are endpoints

2) Define any

   1) Even # of segments in pairs (intervals)

   2) Each IS has a helper vertex

helping (e, e') = lowest vertex above l and between e & e'

3) Degeneracy: no horizontal seg.
Make Monotone (if event)

Case (Start Vertex)
1) Add new interval
2) set helper <= e

Case (End Vertex)
1) if helper is a, merge vertex then add (g, helper)
2) remove interval & helper.

Case (Regular)
1) add (g, helper)
2) replace e with e''
3) helper <= g

Case (Split)
1) add (g, helper)
2) "split" interval
3) helps <= g
Case (Merge)

1) add(helpl, g) add(helpr, g)
2) "Merge" intervals
3) help < g
Another View

1) Make Trapegoidd Decom (swEEP line)
2) For each trap add a diagonal if possible
   a) types of traps

\[ \text{Diagram showing types of traps} \]
Triangulating a Monotone Poly

Y-monotone

Q = 2-sided queue
1) push
2) pop
3) dequeue

Process points in decreasing Y-value.
Maintain this on Reflection

WLOG
Event 8

Case (8 in right chain)
1) Push 8
2) Pop until chain is concave

Case (8 in left chain)
1) Dequeue chain

\(O(n)\)
Each vertex push at most once.