Polygon Triangulation

Piotr Indyk
(and Daniel Vlasic )
Triangulation: Definition

• Triangulation of a simple polygon $P$: decomposition of $P$ into triangles by a maximal set of non-intersecting diagonals
• Diagonal: an open line segment that connects two vertices of $P$ and lies in the interior of $P$
• Triangulations are usually not unique
Example motivation: Guarding an Art Gallery

- An art gallery has several rooms
- Each room guarded by cameras that see in all directions
- Want to have few cameras that cover the whole gallery
Triangulation: Existence

• Theorem:
  – Every simple polygon admits a triangulation
  – Any triangulation of a simple polygon with $n$ vertices consists of exactly $n-2$ triangles

• Proof:
  – Base case: $n=3$
    • 1 triangle ($=n-2$)
    • trivially correct
  – Inductive step: assume theorem holds for all $m<n$
Inductive step

• First, prove existence of a diagonal:
  – Let \( v \) be the leftmost vertex of \( P \)
  – Let \( u \) and \( w \) be the two neighboring vertices of \( v \)
  – If open segment \( uw \) lies inside \( P \), then \( uw \) is a diagonal
Inductive step ctd.

• If open segment $uw$ does not lie inside $P$
  – there are one or more vertices inside triangle $uvw$
  – of those vertices, let $v'$ be the farthest one from $uw$
  – segment $vv'$ cannot intersect any edge of $P$, so $vv'$ is a diagonal

• Thus, a diagonal exists
• Can recurse on both sides
• Math works out:
  $$(n1-2) + (n2-2) = (n1+n2-2)-2$$
Back to cameras

• Where should we put the cameras?
• Idea: cover every triangle
  – 3-color the nodes (for each edge, endpoints have different colors)
  – Each triangle has vertices with all 3 colors
  – Can choose the least frequent color class → $\left\lfloor \frac{n}{3} \right\rfloor$ cameras suffice
  – There are polygons that require $\left\lceil \frac{n}{3} \right\rceil$ cameras
3-coloring Always Possible

• Simple inductive argument (Ilya)
  – Find the diagonal and split the polygon
  – Independently 3 color both sub-polygons
  – Adjust the colorings so that the colors of the vertices on the diagonal match

• More complex, but linear-time algorithm:
  – Take the dual graph $G$
  – This graph has no cycles
  – Find 3-coloring by DFS traversal of $G$:
    • Start from any triangle and 3-color its vertices
    • When reaching new triangle we cross an already colored diagonal
    • Choose the third color to finish the triangle
How to triangulate fast

• Partition the polygon into y-monotone parts, i.e., into polygons $P$ such that an intersection of any horizontal line $L$ with $P$ is connected

• Triangulate the monotone parts

• Assume all x,y-coordinates distinct, to avoid headache
Monotone partitioning

- Line sweep (top down)
- Vertices where the direction changes downward<>upward are called *turn vertices*
- To have $y$-monotone pieces, we need to get rid of turn vertices:
  - when we encounter a turn vertex, it might be necessary to introduce a diagonal and split the polygon into pieces
Vertex Ontology

February 22, 2007
Removing split

• To partition $P$ into $y$-monotone pieces, get rid of split and merge vertices
  – add a diagonal going upward from each split vertex

• Where do the edges go?
Helpers

- Let $\text{helper}(e_j)$ be the lowest vertex above the sweep-line such that the horizontal segment connecting the vertex to $e_j$ lies inside $P$. 
Removing Split Vertices

• For a split vertex $v_i$, let $e_j$ be the edge immediately to the left of it
• Add a diagonal from $v_i$ to $helper(e_j)$
• Question: does the new edge intersect any existing one?
• Question II: does the new edge introduce any merge vertices?
  – No
  – The edge goes “down” from the helper
  – Merge vertex is incident to only “up” edges
→ Can get rid of merge vertices by a second scan (in reverse direction)
Proof

• Recall:
  – $v_i$ is the split vertex
  – $e_j$ is the edge to the left of $v_i$
  – $h = \text{helper}(e_j)$

• The segment $h-v_i$ does not intersect any other boundary segment because:
  – The dotted segments do not intersect any polygon boundaries, by def. Same holds for $e_j$.
  – If there was any polygon segment intersecting $h-v_i$, one of its endpoints would have to be inside the yellow region, and the other outside of the yellow region
  – Lemma (next slide): given a collection of such segments, at least one inside endpoint (say, $p$) must be reachable from $e_j$ horizontally, without intersecting any other segments

• Note: the edges added earlier are, for the above purposes, considered to be the boundary edges
Proof of the Lemma

• Pick \( p \) that is the furthest away from the line \( L \) passing through \( h \) and \( vi \)
• Assume that the horizontal ray from \( p \) towards \( ej \) hits another segment \( s \), splitting it into two segments, say \( s' \) and \( s'' \)
• Then if we continue along one of \( s' \) or \( s'' \), we get further away from \( L \). Thus, when we hit an endpoint, it is even further away from \( L \) than \( p \)
• A contradiction!
Digression

• We could try to make that argument for an endpoint $p$ that is closest to $e_j$ (rather than furthest from $L$)

• This does not work! See example
The algorithm

• Use plane sweep method
  – move sweep line downward over the plane (need to sort first)
  – halt the line on every vertex
  – handle the event depending on the vertex type
    • Update the helper
    • Add the diagonal if a split vertex

• Time: $O(n \log n)$
Triangulating monotone polygon
Triangulating monotone polygons

• Single pass from top to bottom
• Keeps removing triangles
• Invariants:
  – Top vertex convex
  – Other vertices form a concave chain
Altogether

• Can triangulate a polygon in $O(n \log n)$ time
• Fairly simple $O(n \log^* n)$ time algorithms
• Very complex $O(n)$ time algorithm
Removing Merge Vertices

- For a merge vertex $v_i$, let $e_j$ be the edge immediately to the left of it.
- $v_i$ becomes $helper(e_j)$ once we reach it.
- Whenever the $helper(e_j)$ is replaced by some vertex $v_m$, add a diagonal from $v_m$ to $v_i$.
- If $v_i$ is never replaced as $helper(e_j)$, we can connect it to the lower endpoint of $e_j$. 

Fanaoa 22, 2007