Freie Universität Berlin

Computational Geometry:
Polygon triangulation

Panos Giannopoulos, Wolfgang Mulzer, Lena Schlipf
AG TI

SS 2013

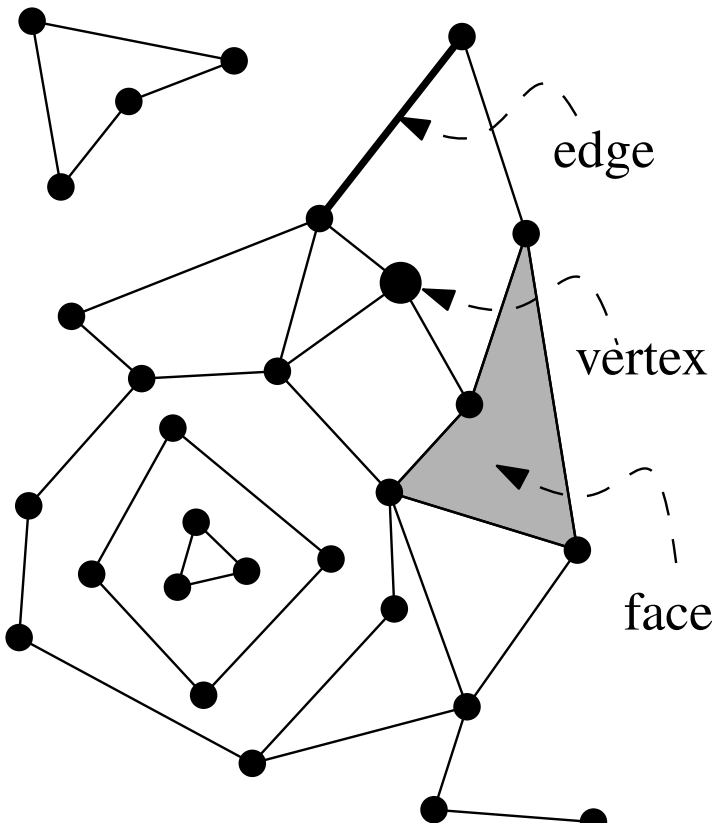
Outline

Doubly-connected edge list

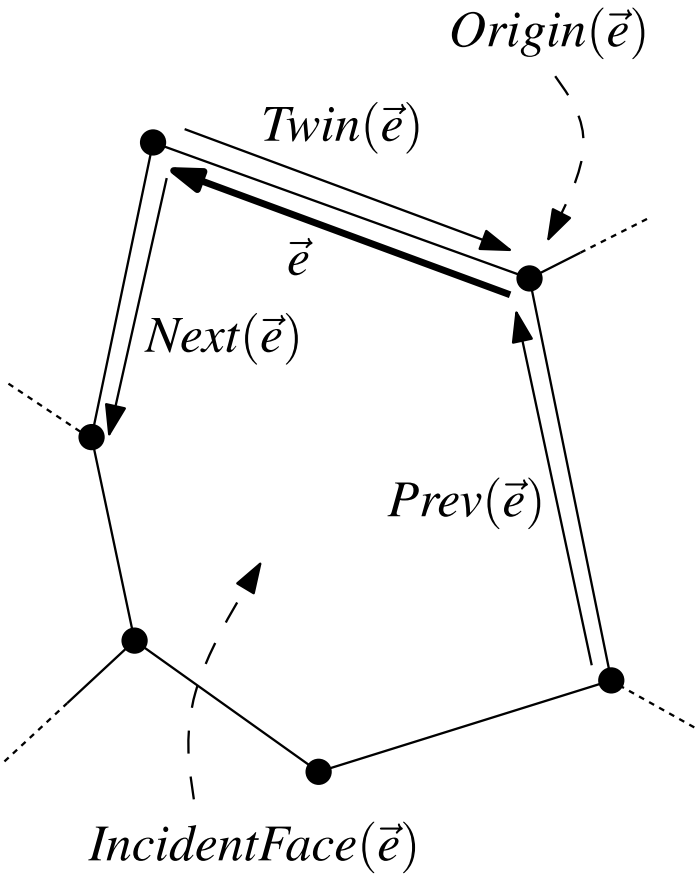
Triangulating a simple polygon I

Doubly-connected edge list (DCEL)

Representing a planar subdivision

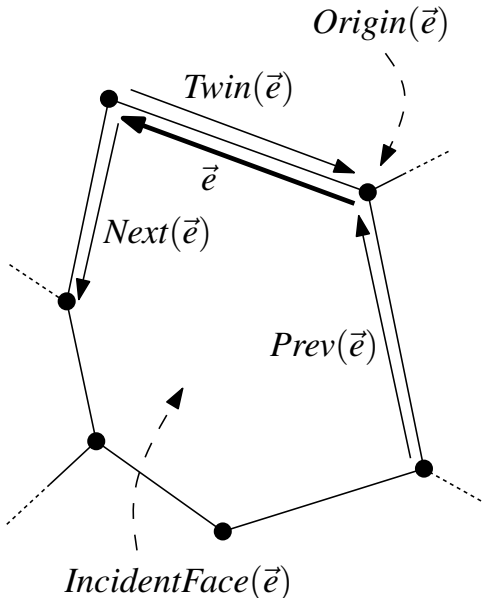


Representing a planar subdivision



- geometric and topological information stored in records (vertex, face, half-edge)
- two oriented half-edges for an edge (origin, distination)
- incident face of a half-edge lies to its left
- counterclockwise traversal of a face

Doubly-connected edge list (DCEL)



The doubly-connected edge list is a subdivision representation structure with an object for every vertex, every half-edge, and every face

A vertex object stores:

- Coordinates
- **IncidentEdge** (some half-edge leaving it)

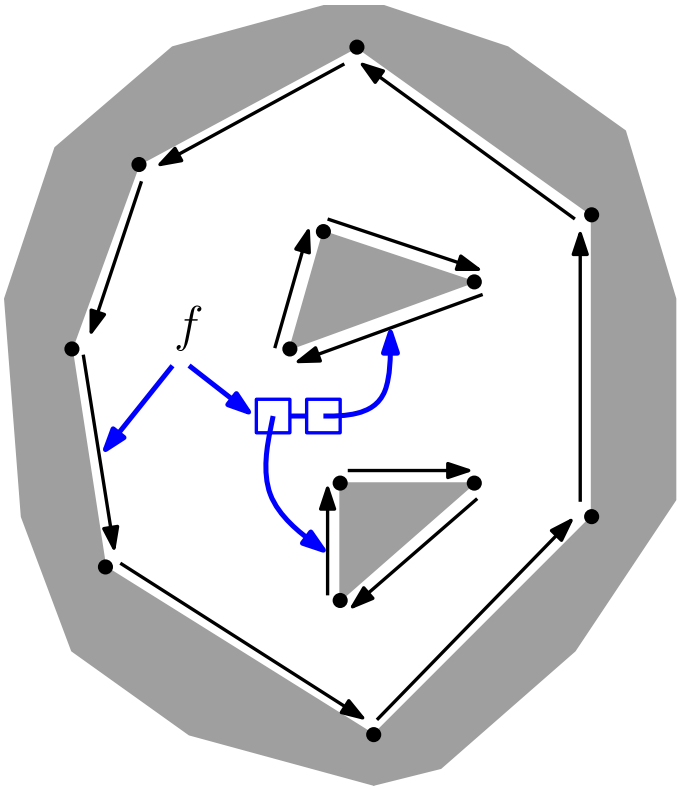
A half-edge object stores:

- **Origin** (vertex)
- **Twin** (half-edge)
- **IncidentFace** (face)
- **Next** (half-edge in cycle of the incident face)
- **Prev** (half-edge in cycle of the incident face)

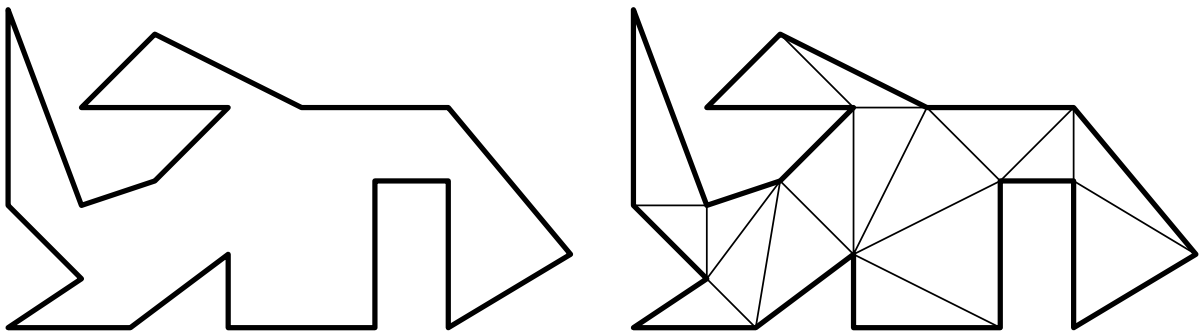
Doubly-connected edge list (DCEL)

A face object stores:

- **OuterComponent** (half-edge of outer cycle)
- **InnerComponents** (list of half-edges for the inner cycles bounding the face)



Polygon triangulation



A decomposition of a polygon into triangles
by a maximal set of non-intersecting diagonals

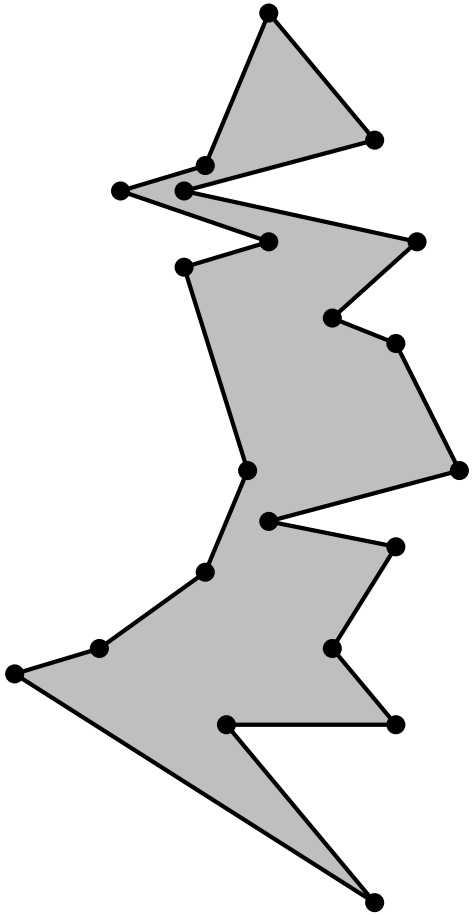
Triangulating a simple polygon

Approach:

A simple polygon is y-monotone iff any horizontal line intersects it in a connected set (or not at all)

Use plane sweep to partition the polygon into y-monotone polygons

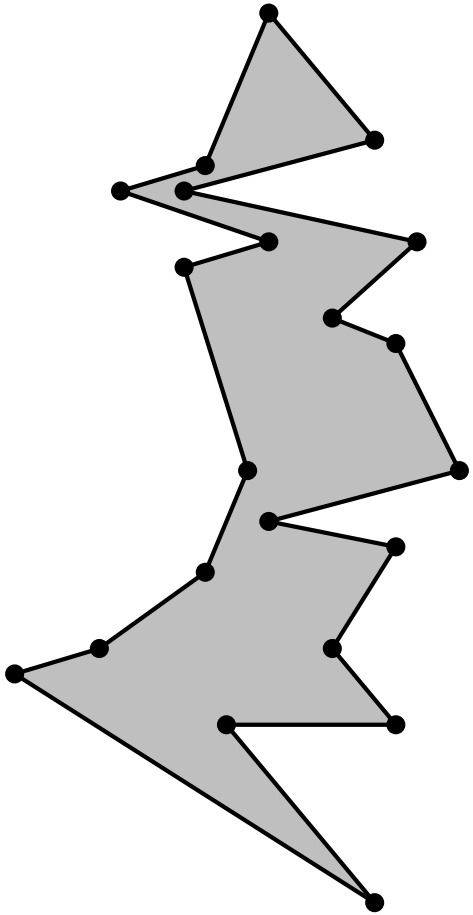
Then triangulate each y-monotone polygon



Monotone polygons

A y-monotone polygon has a top vertex, a bottom vertex, and two y-monotone chains between top and bottom as its boundary

Any simple polygon with one top vertex and one bottom vertex is y-monotone



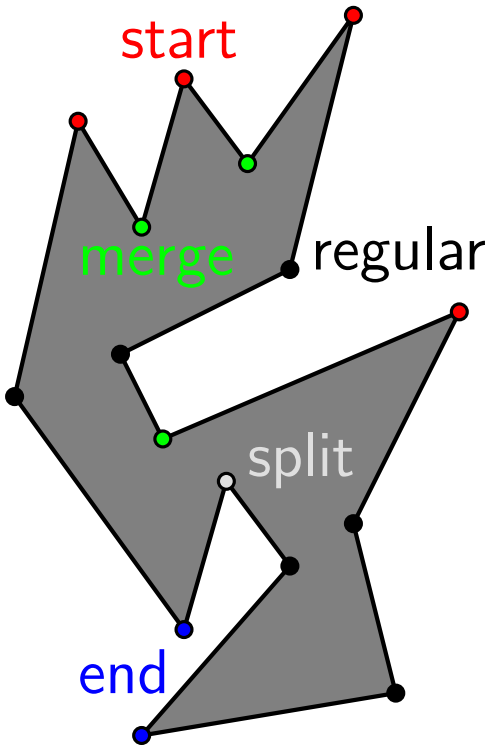
Formal proof ?

Vertex types

What types of vertices does a simple polygon have?

- start
- stop
- split
- merge
- regular

. . . imagining a sweep line going top to bottom

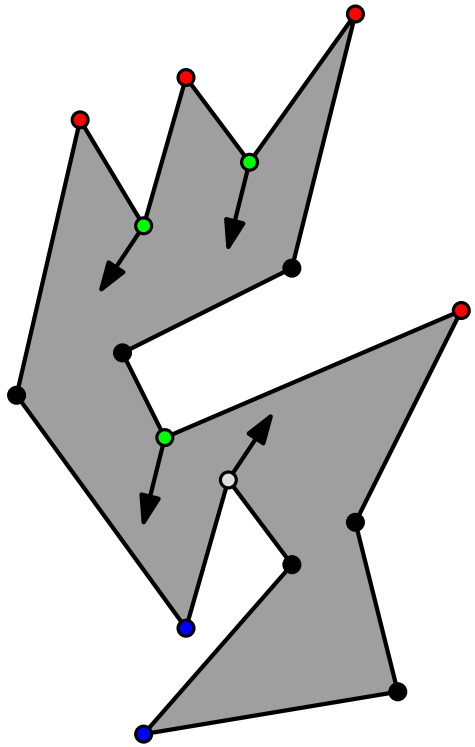


Define type properties!

Sweep ideas

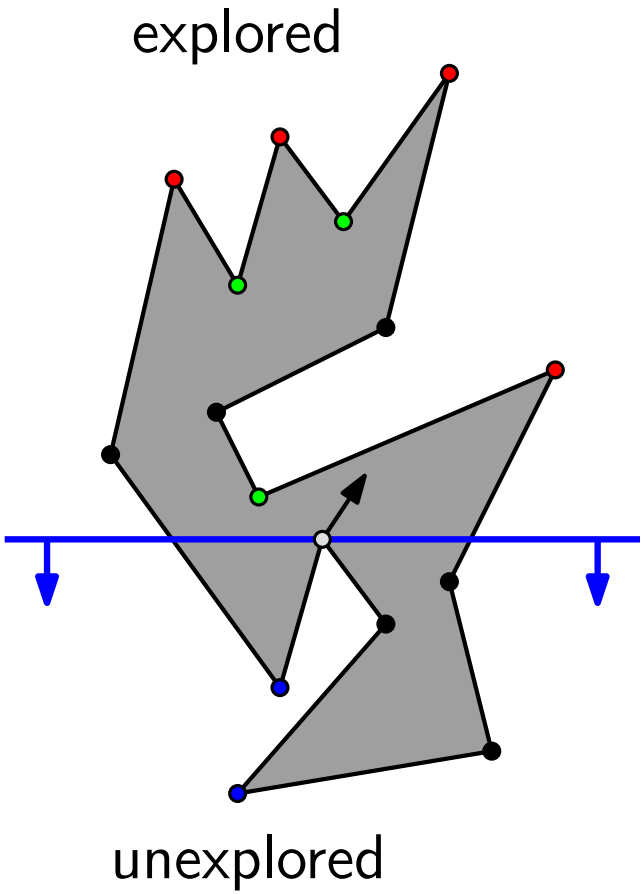
Find diagonals from each merge vertex down, and from each split vertex up

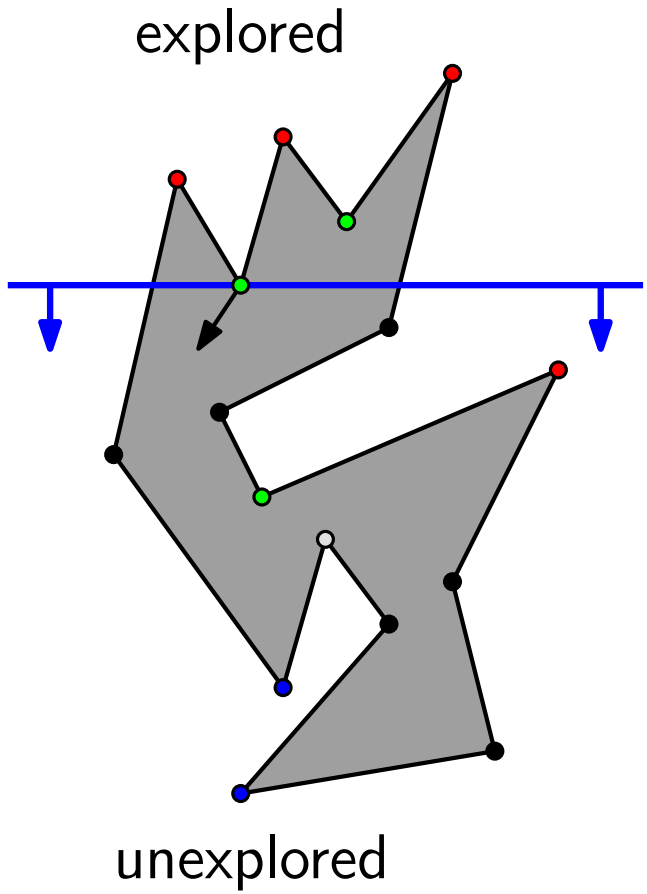
A simple polygon with no split or merge vertices can have at most one start and one end vertex, so it is y-monotone



Proof!

Sweep ideas

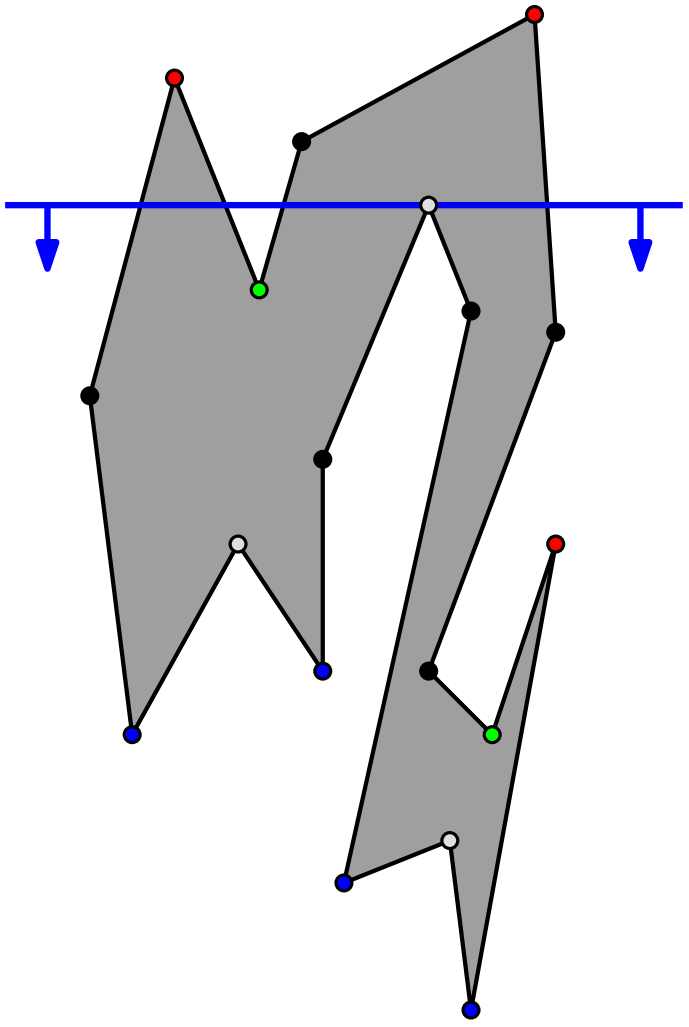




Sweep ideas

Where can a diagonal from a split vertex go?

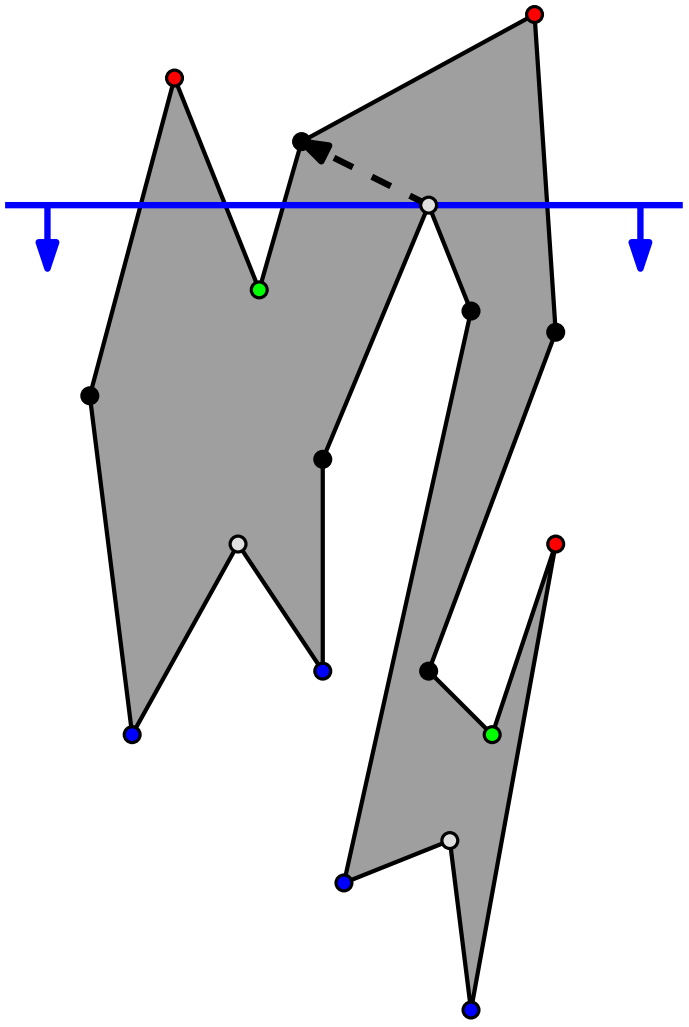
Perhaps the upper endpoint of the edge immediately left of the merge vertex?



Sweep ideas

Where can a diagonal from a split vertex go?

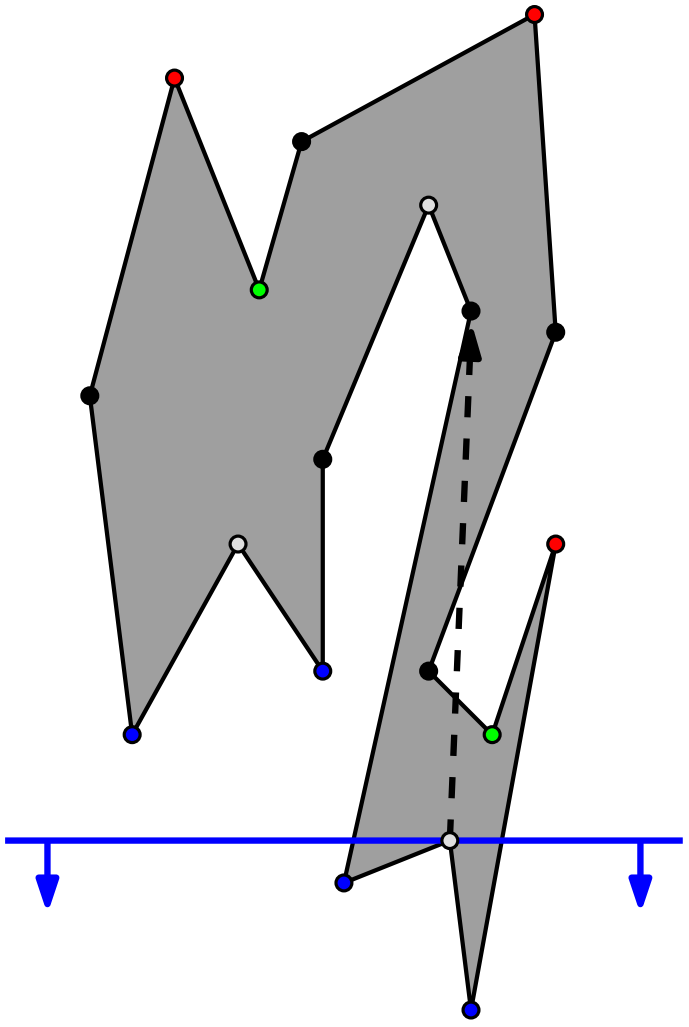
Perhaps the upper endpoint of the edge immediately left of the merge vertex?



Sweep ideas

Where can a diagonal from a split vertex go?

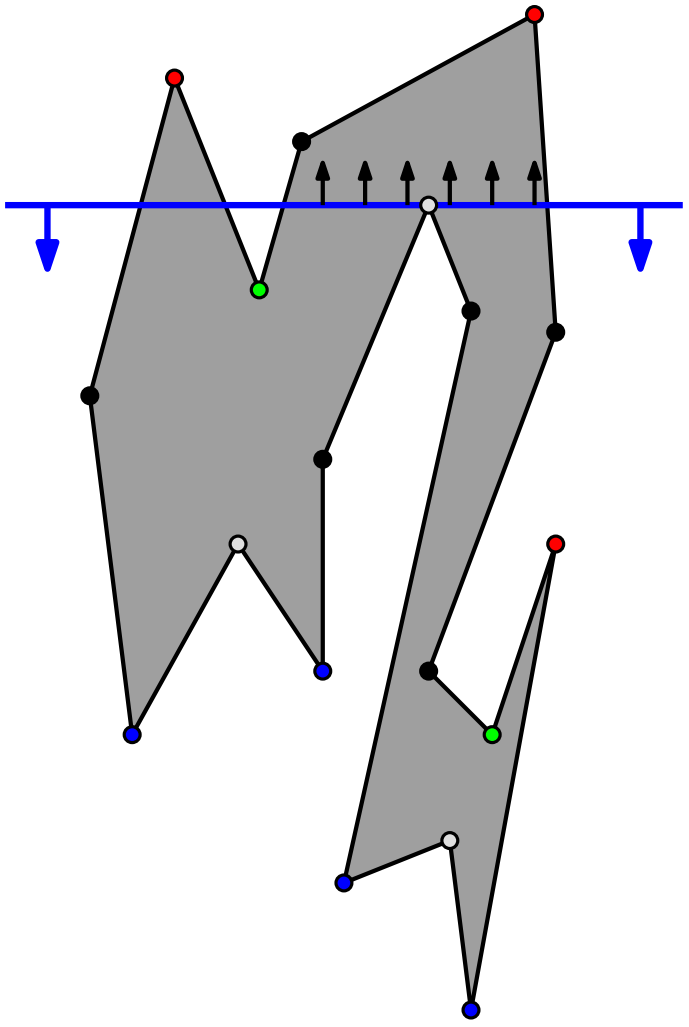
Perhaps the upper endpoint of the edge immediately left of the merge vertex?



Sweep ideas

Where can a diagonal from a split vertex go?

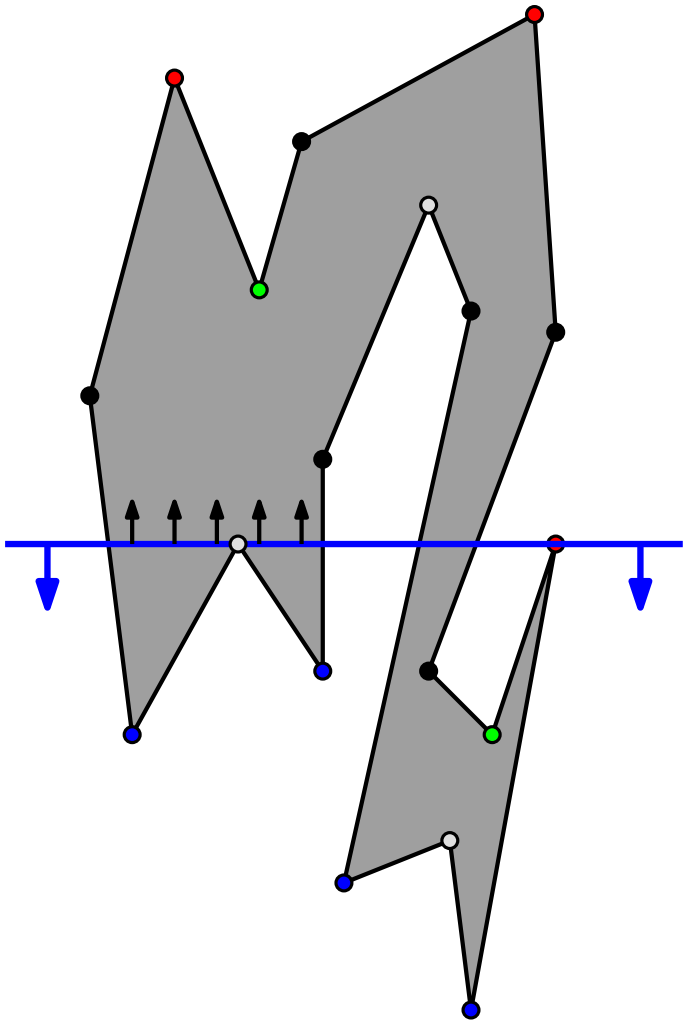
Perhaps the last vertex passed in the same “component”?



Sweep ideas

Where can a diagonal from a split vertex go?

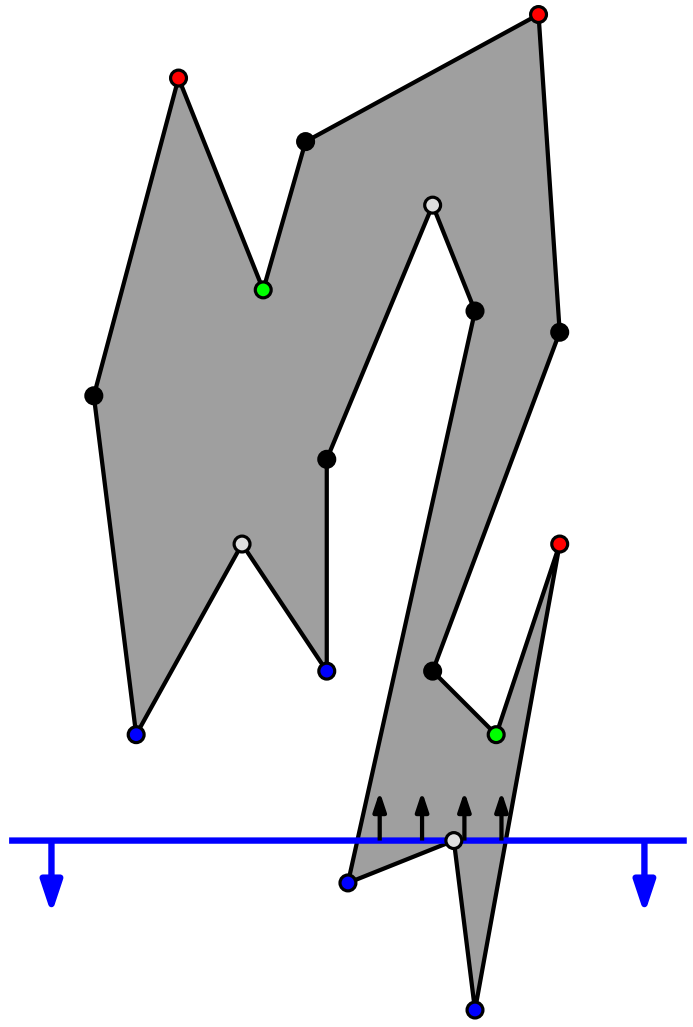
Perhaps the last vertex passed in the same "component"?



Sweep ideas

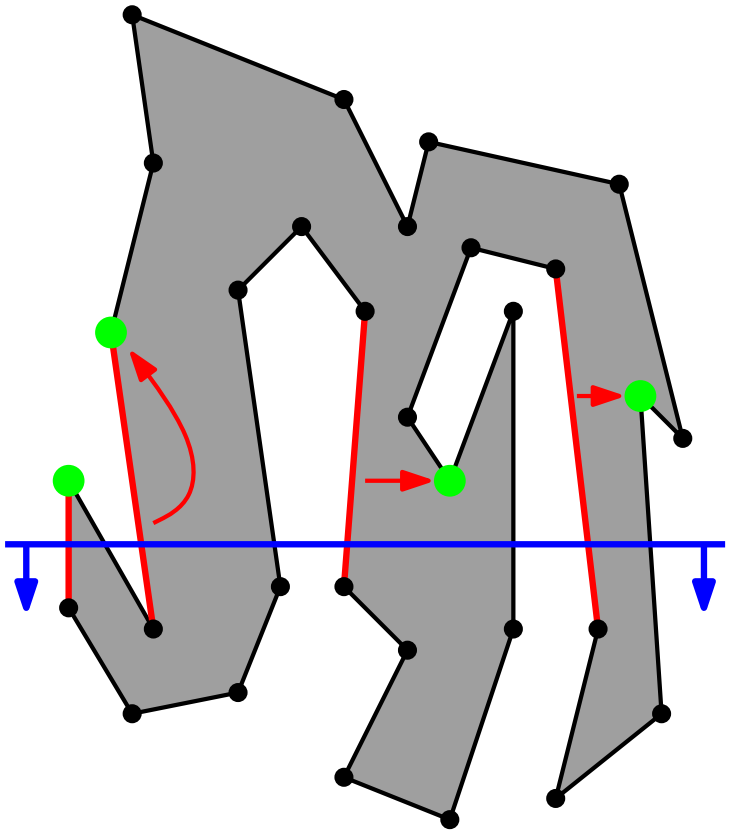
Where can a diagonal from a split vertex go?

Perhaps the last vertex passed in the same “component”?



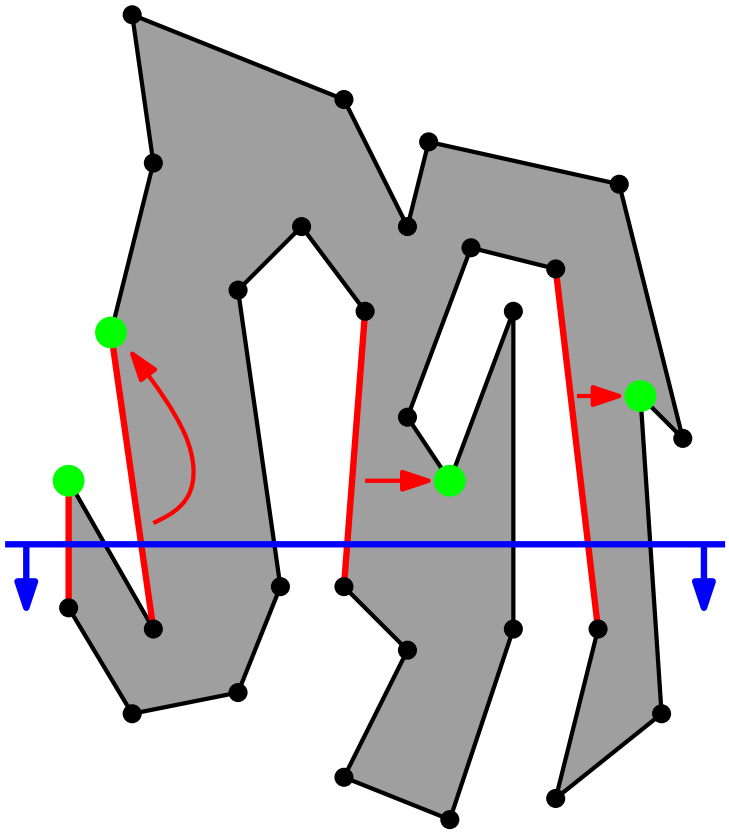
Status of sweep

The status is the set of edges intersecting the sweep line that have the polygon to their right, sorted from left to right, and each with their *helper*: the last vertex passed in that component



Helpers of edges!

The helper for an edge e that has the polygon right of it, and a position of the sweep line, is the lowest vertex v above the sweep line such that the horizontal line segment connecting e and v is inside the polygon



Status structure, event list

The status structure stores all edges that have the polygon to the right, with their helper, sorted from left to right in the leaves of a balanced binary search tree

The events happen only at the vertices: sort them by y-coordinate and put them in a list (or array, or tree)

Main algorithm

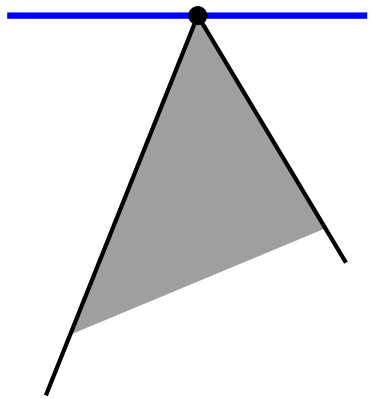
Initialize the event list (all vertices sorted by decreasing y-coordinate) and the status structure (empty)

While there are still events in the event list, remove the first (topmost) one and handle it

Event handling

Start vertex v:

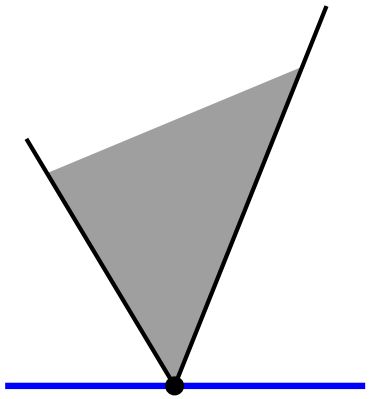
- Insert the counterclockwise incident edge in T with v as the helper



Event handling

End vertex v:

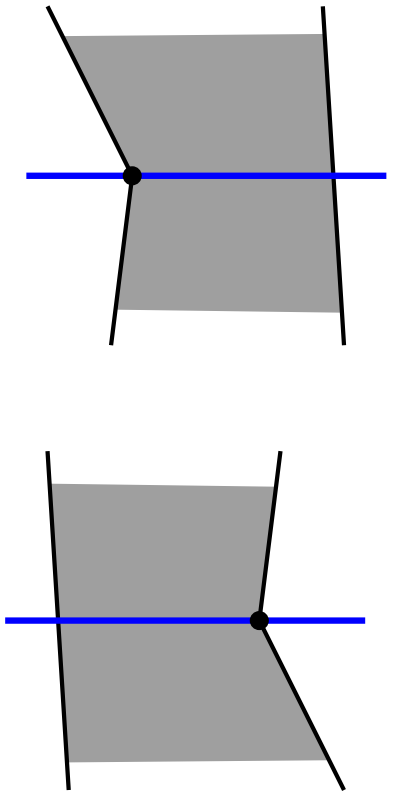
- Delete the clockwise incident edge and its helper from T



Event handling

Regular vertex v:

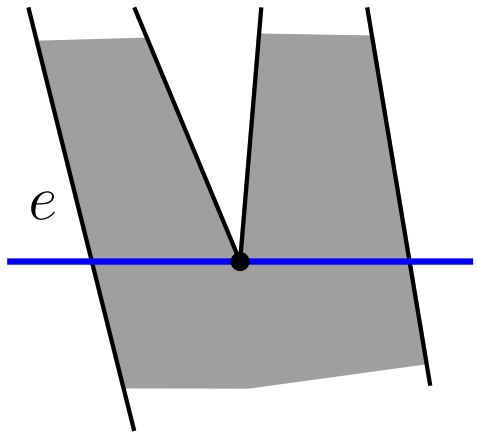
- If the polygon is right of the two incident edges, then replace the upper edge by the lower edge in T, and make v the helper
- If the polygon is left of the two incident edges, then find the edge e directly left of v, and replace its helper by v



Event handling

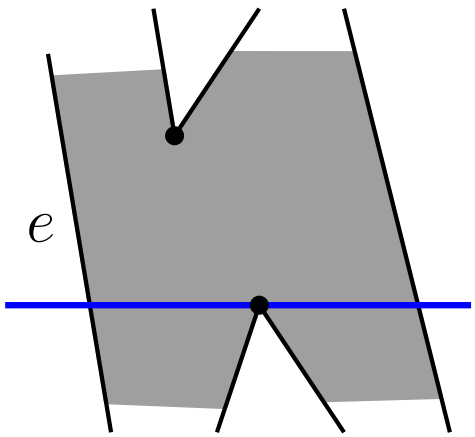
Merge vertex v:

- Remove the edge clockwise from v from T
- Find the edge e directly left of v, and replace its helper by v



Split vertex v:

- Find the edge e directly left of v, and choose as a diagonal the edge between its helper and v
- Replace the helper of e by v
- Insert the edge counterclockwise from v in T, with v as its helper



Efficiency

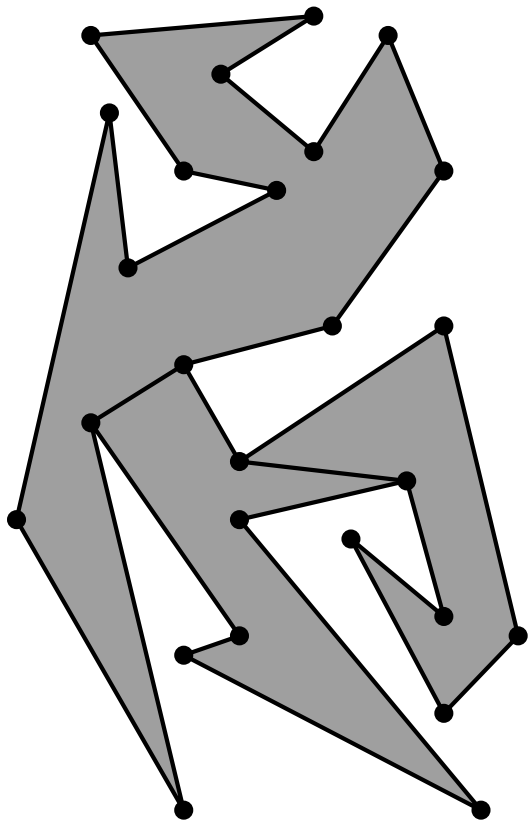
Sorting all events by y-coordinate takes $O(n \log n)$ time

Every event takes $O(\log n)$ time, because it only involves querying, inserting and deleting in T

Representation

A simple polygon with some diagonals is a subdivision $\Rightarrow$ use a DCEL

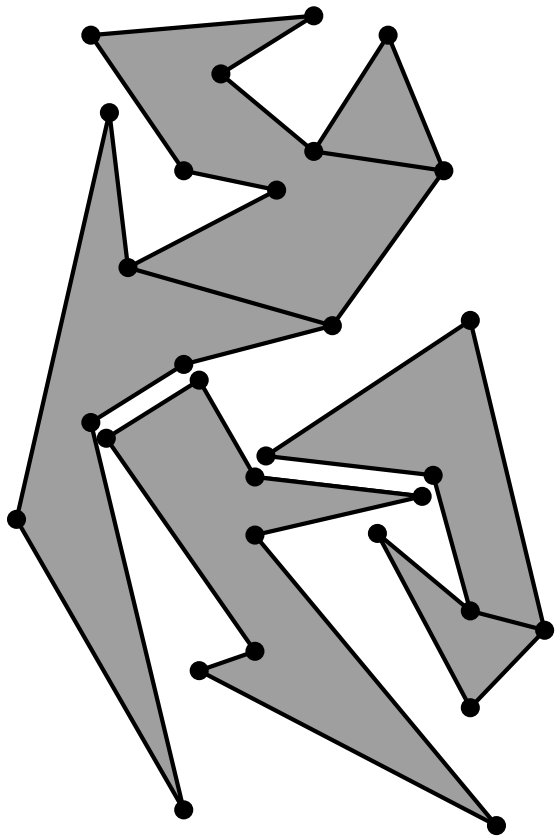
Question: How many diagonals may be chosen to the same vertex?



More sweeping

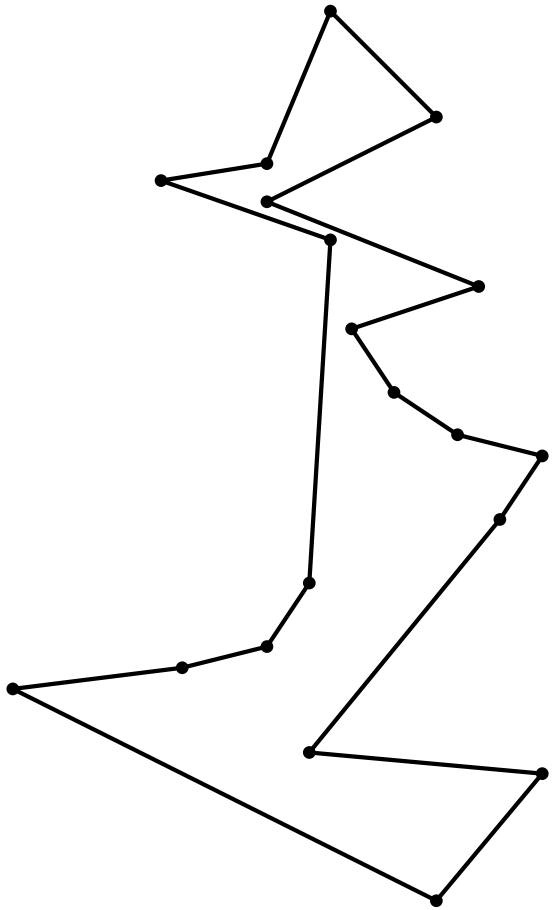
With an upward sweep in each subpolygon, we can find a diagonal down from every merge vertex (which is a split vertex for an upward sweep!)

This makes all subpolygons y-monotone



Theorem: A simple polygon with n vertices can be partitioned into y-monotone pieces in $O(n \log n)$ time

Proof?

This Friday:

How to triangulate a y-monotone polygon?