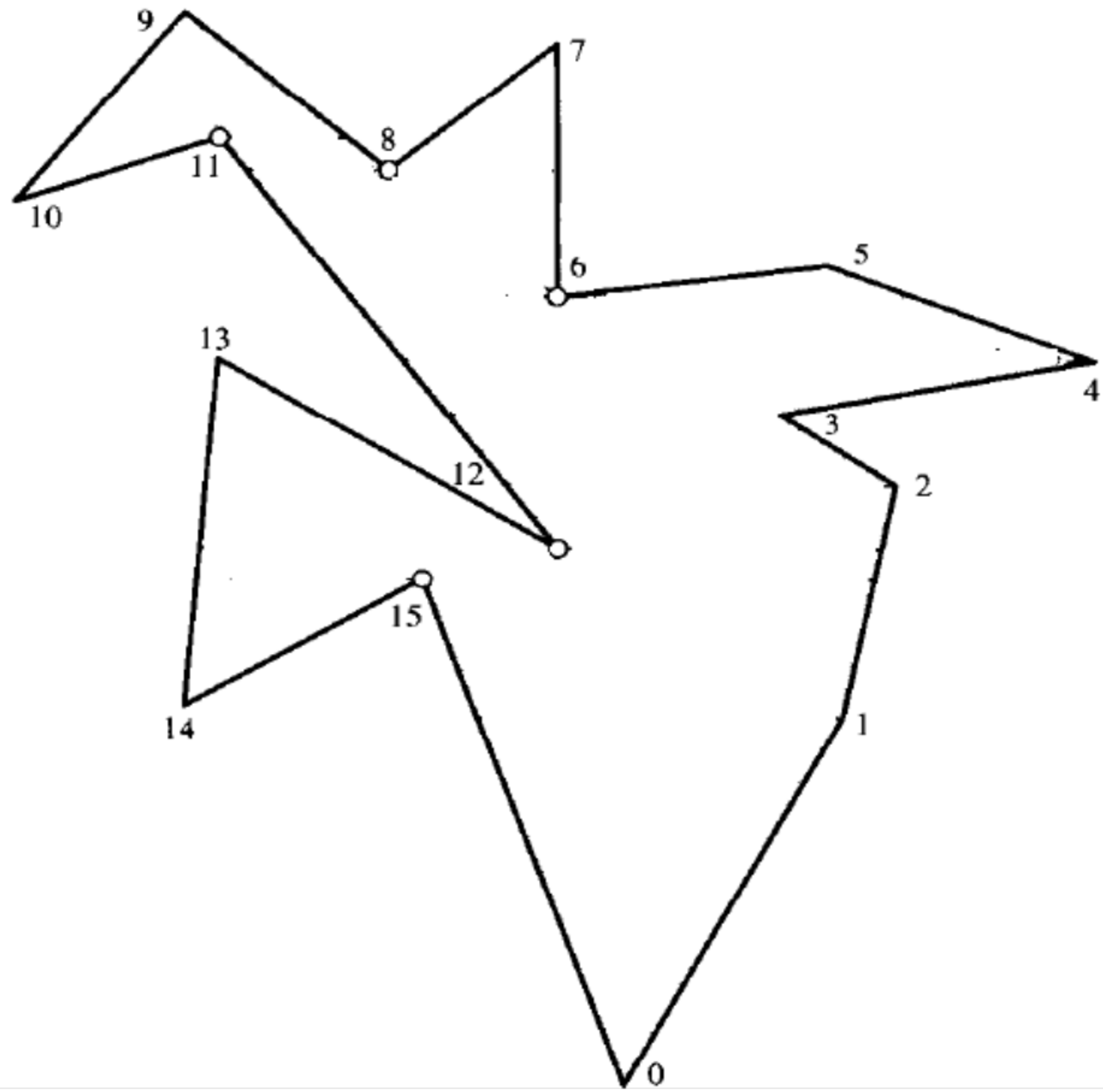
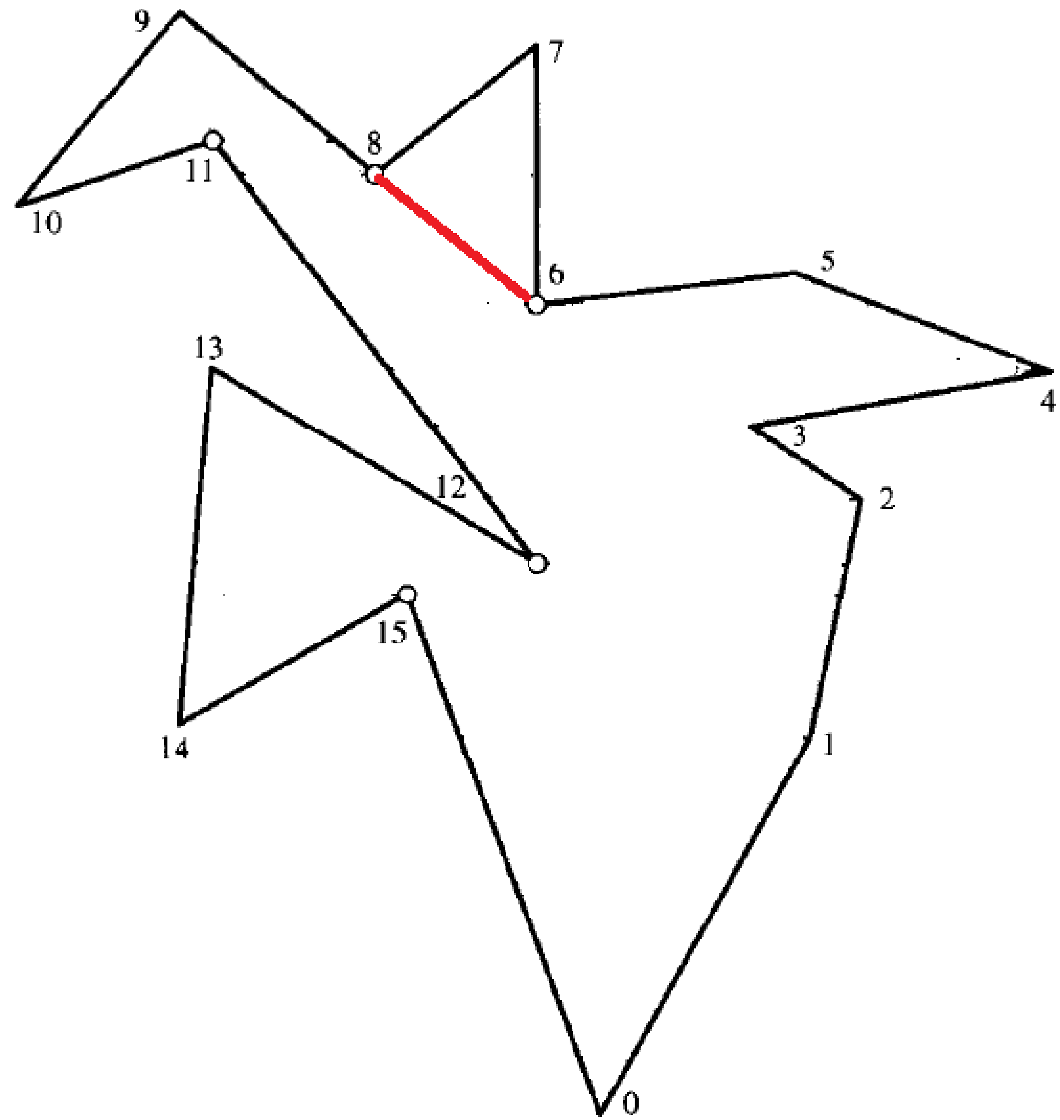
Algorithm to triangulate a non-monotone (normal) polygon

- Step 1: Partition a Polygon to monotone pieces
- Step 2 :Triangulate each monotone piece (can be done in linear time)
- If step 1 can be done efficiently (less than $O(n^2)$), then we can develop an efficient algorithm than the current $O(n^2)$ algorithm for triangulating a polygon
- We proceed focusing on a normal polygon

To make P monotone:

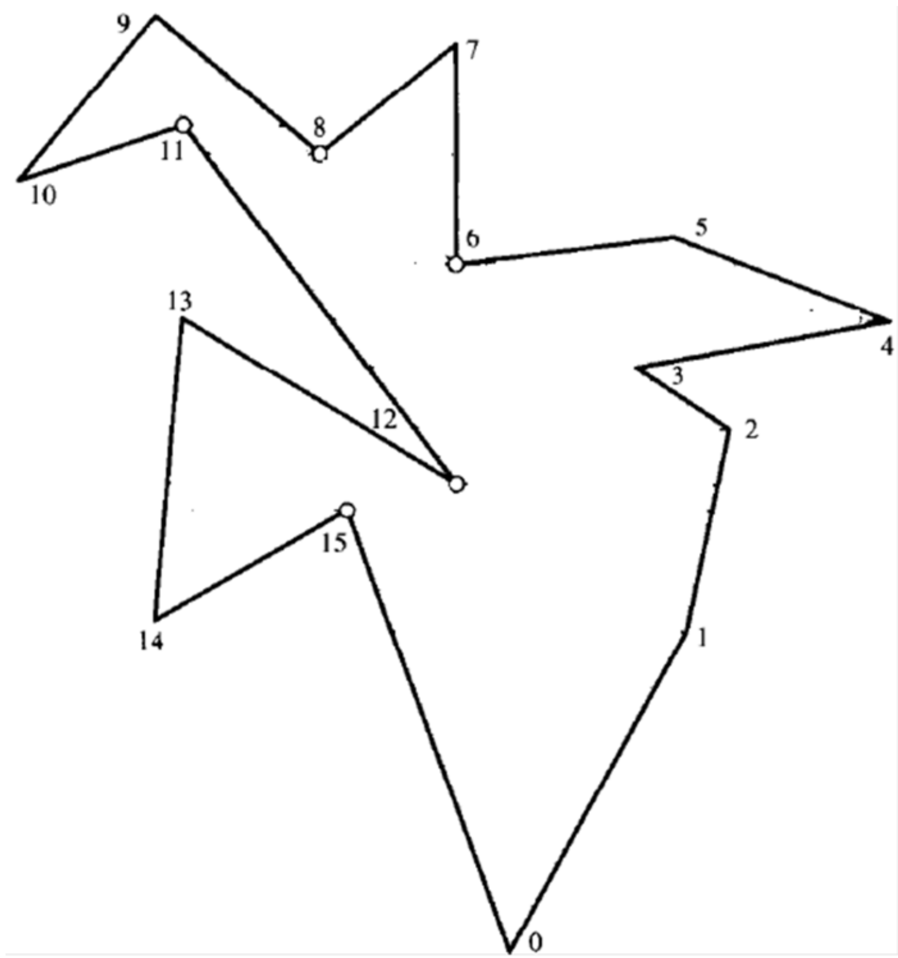


- Remove/ Break interior cusps
- Consider vertex 8



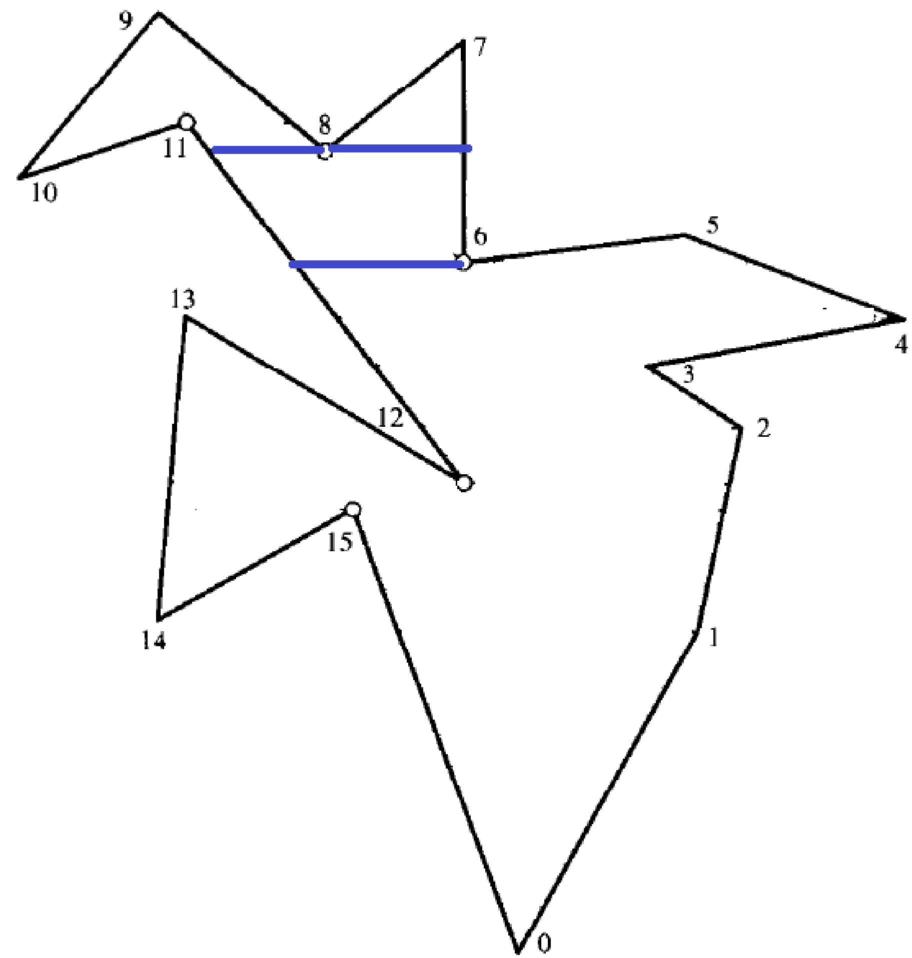
Δ6 7 8 is monotone

How do we select a vertex (eg: vertex 6) from many choices ?



Restrict our choice of a vertex to connect to :

- If we can restrict the choice of a vertex



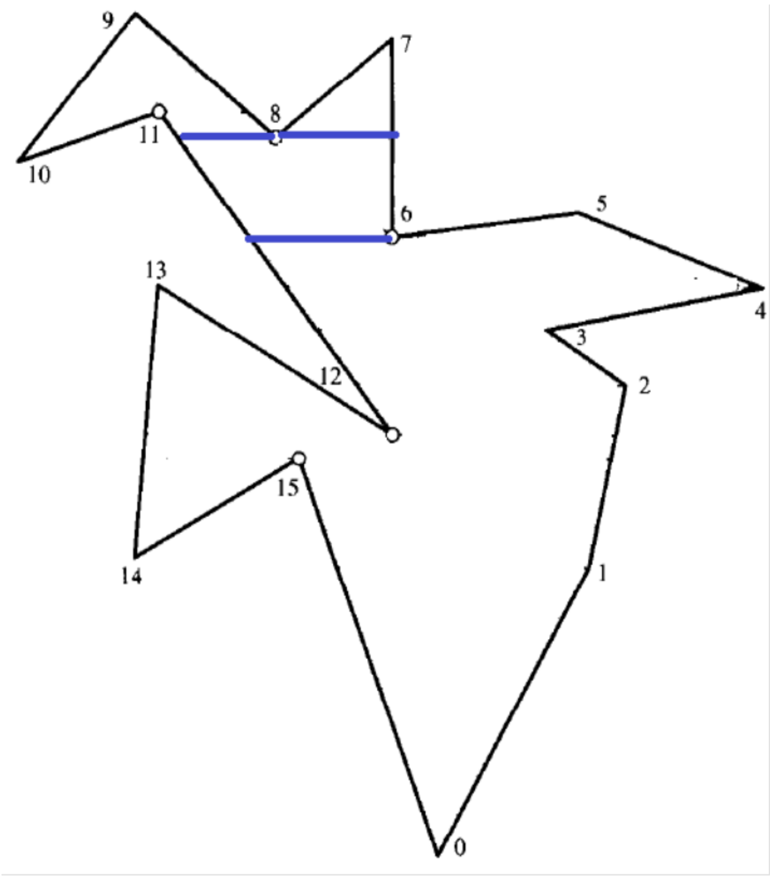
- Trapezoid
- What is a trapezoid?

Trapezoid

- Trapezoid: A convex quadrilateral with two parallel edges
- To summarize what we are doing up to now :
- We are trying to break the interior cusps of a Polygon
- By connecting vertex v_1 (reflex vertex which caused non-monotonicity) to another vertex v_2
- By introducing a trapezoid structure, we are restricting the choice for v_2
- First we trapezoidalize the polygon, then we divide it in to monotone pieces by breaking the interior cusps

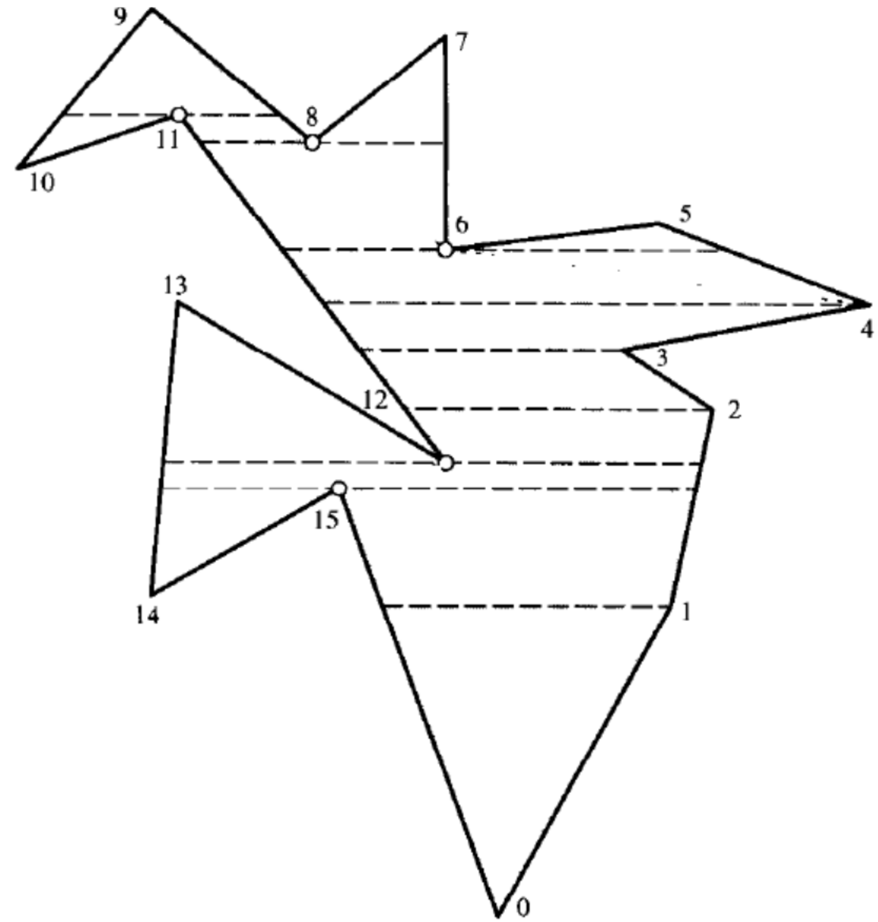
Trapezoidalization[Chazelle & Incerpi,1984]

- We have drawn horizontal lines through a vertex of a polygon
- We can either draw horizontal / vertical lines
- It can be a horizontal or vertical trapezoidalization.
- We focus on horizontal trapezoidalization.
- Assumption: No two vertices lie on a horizontal line



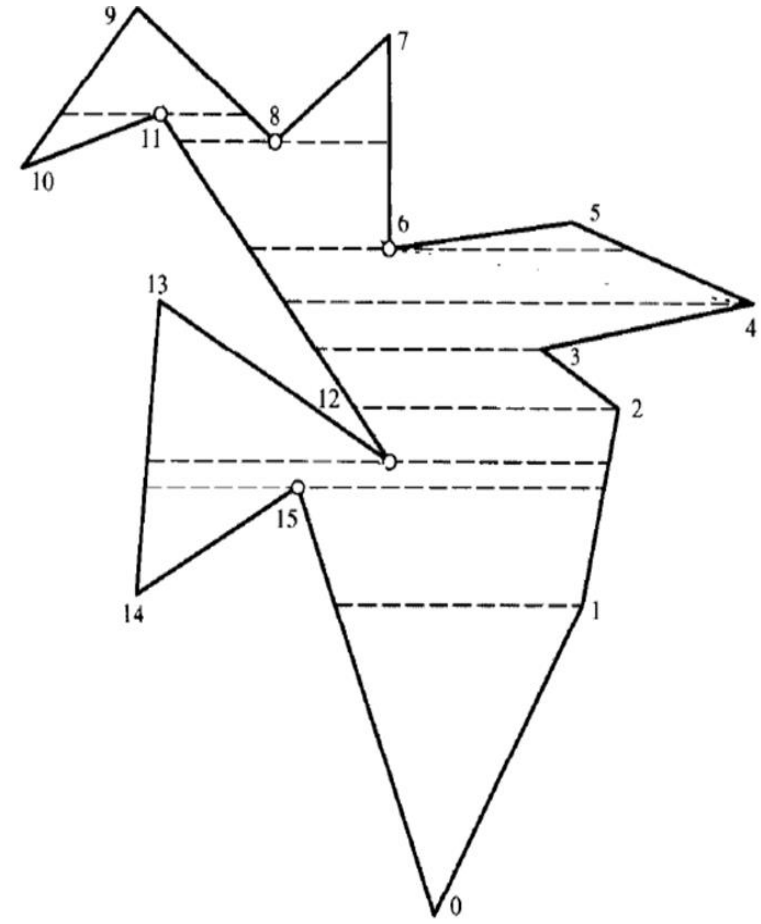
Horizontal trapezoidalization

- Polygon will be partitioned in to Trapezoids by drawing horizontal lines through a vertex of a polygon



- P completely trapezoidalized?

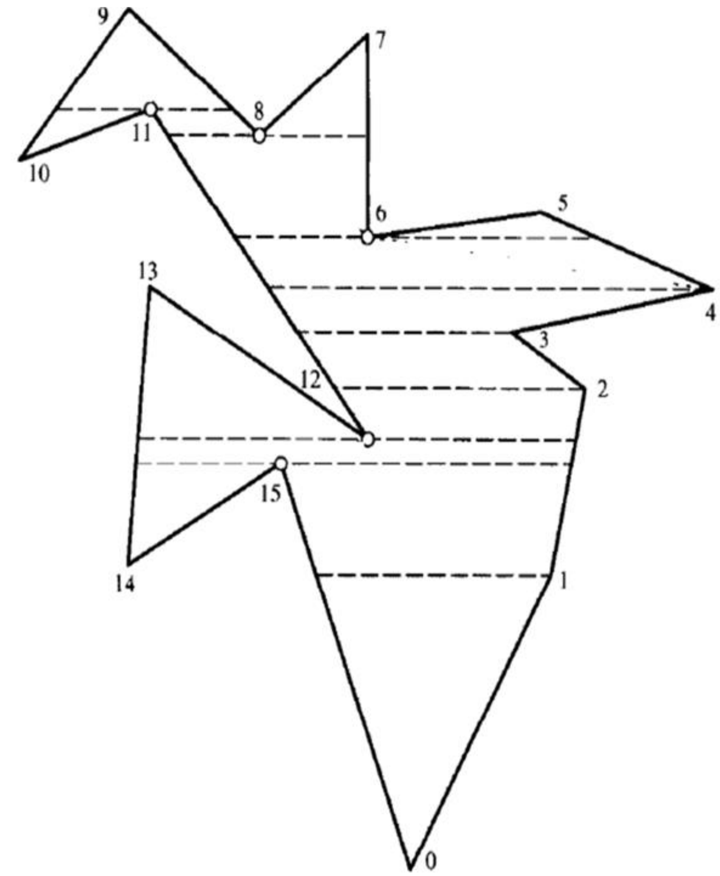
Horizontal trapezoidalization



- Triangle can be viewed as a degenerate trapezoid with one of the parallel edges of zero length
- Vertices through which the horizontal lines are drawn are known as **Supporting Vertices**

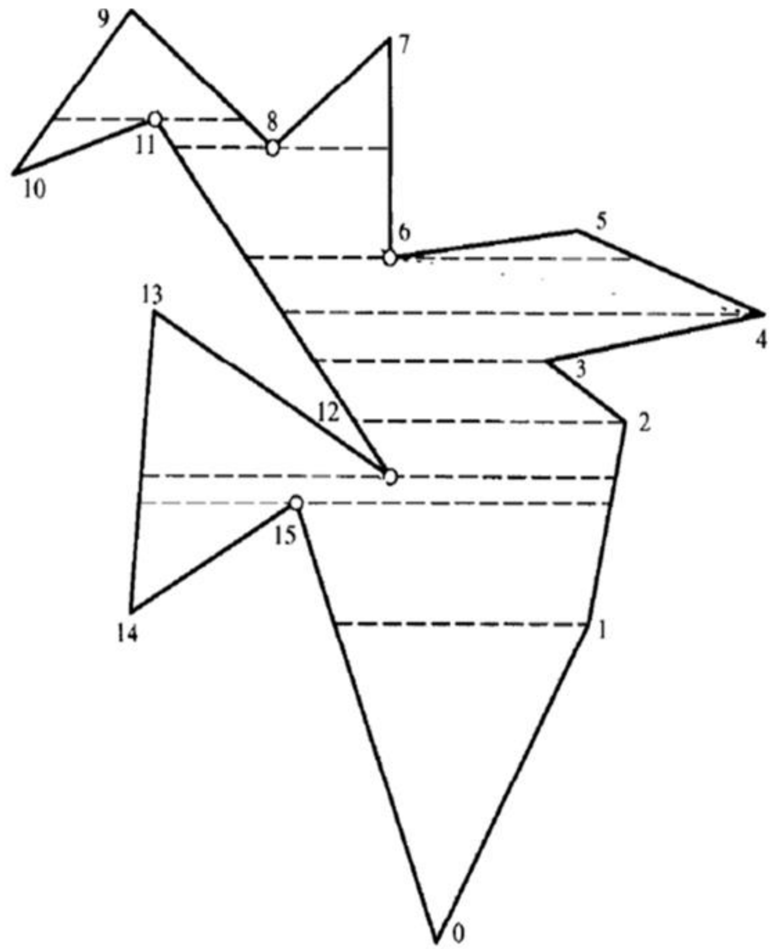
Horizontal trapezoidalization

- A horizontal trapezoidalization of a polygon (P) is obtained by :
- Drawing a horizontal line through every vertex of P
- Formally:
- Let s be a maximal (open) horizontal line which passes through v
- s ⊂ P and s ∩ ∂P = v
- s is either to left or right or to both directions on a vertex
- s=v



Removing the interior cusp

- Every trapezoid has exactly two supporting vertices
- One on its upper edge and one on its lower edge
- If the supporting vertex is on the interior of an upper or lower edge, then it is an interior cusp
- For upward pointing cusps: connect the supporting vertices on the trapezoid upward the cusp
- For downward pointing cusps: connect the supporting vertices on the trapezoid downward the cusp

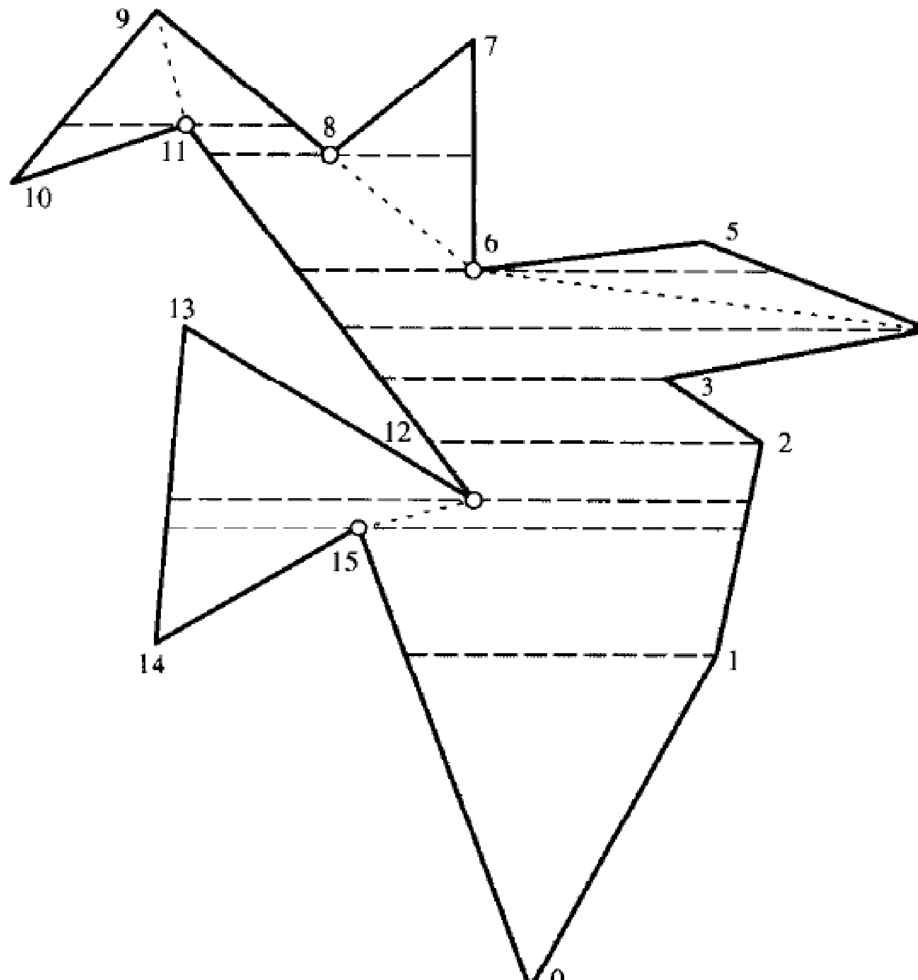


Removing the interior cusp

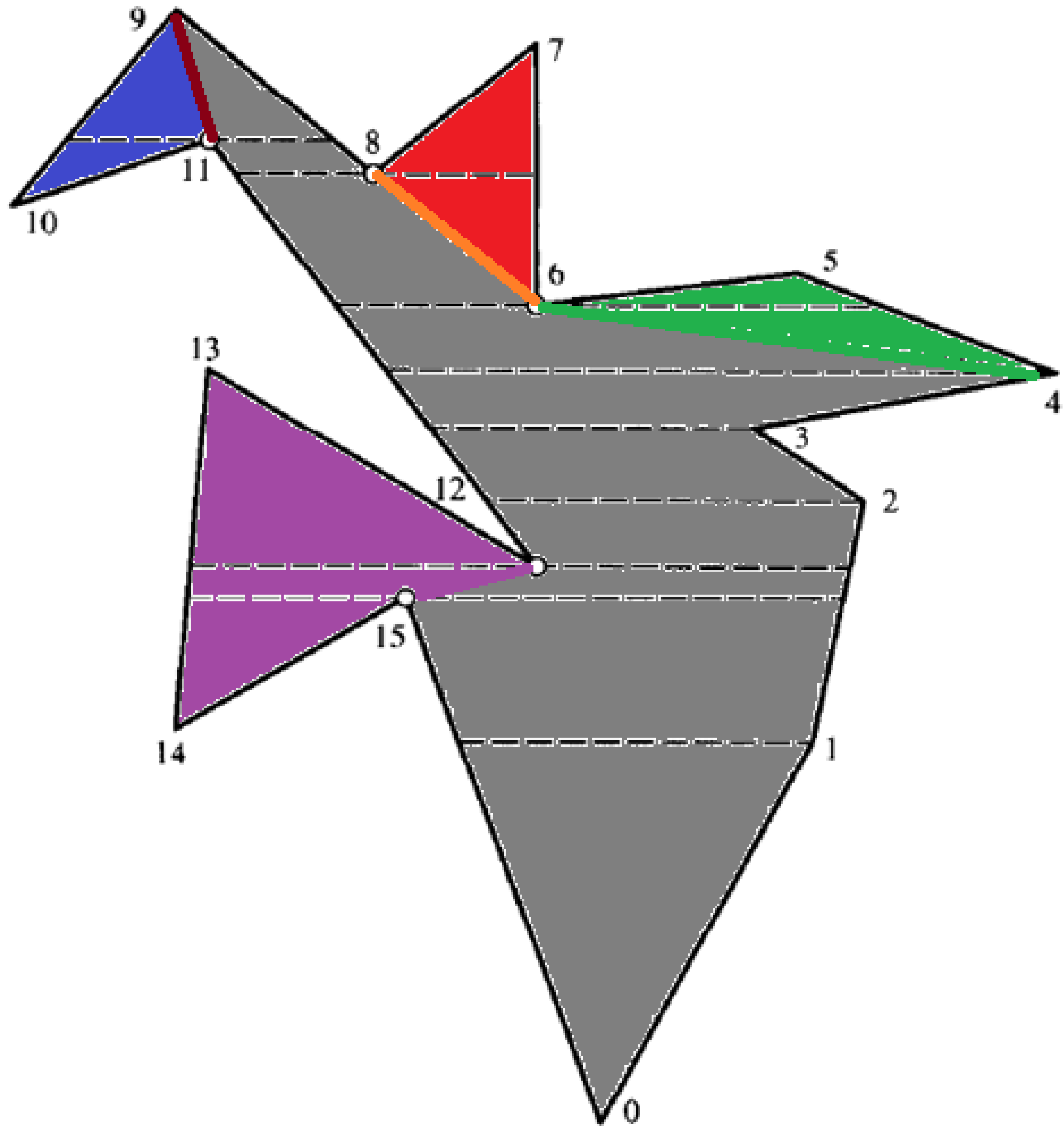
- Downward pointing cusps

  Eg: 8,6,12
- Upward pointing cusp

  Eg: 11, 15

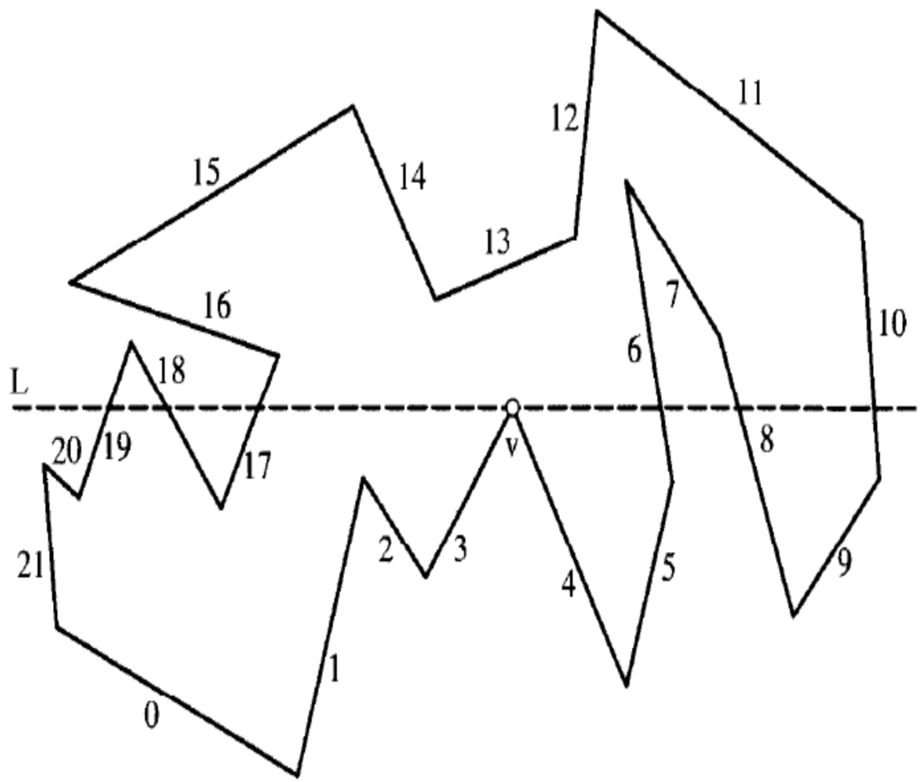


Monotone sub polygons



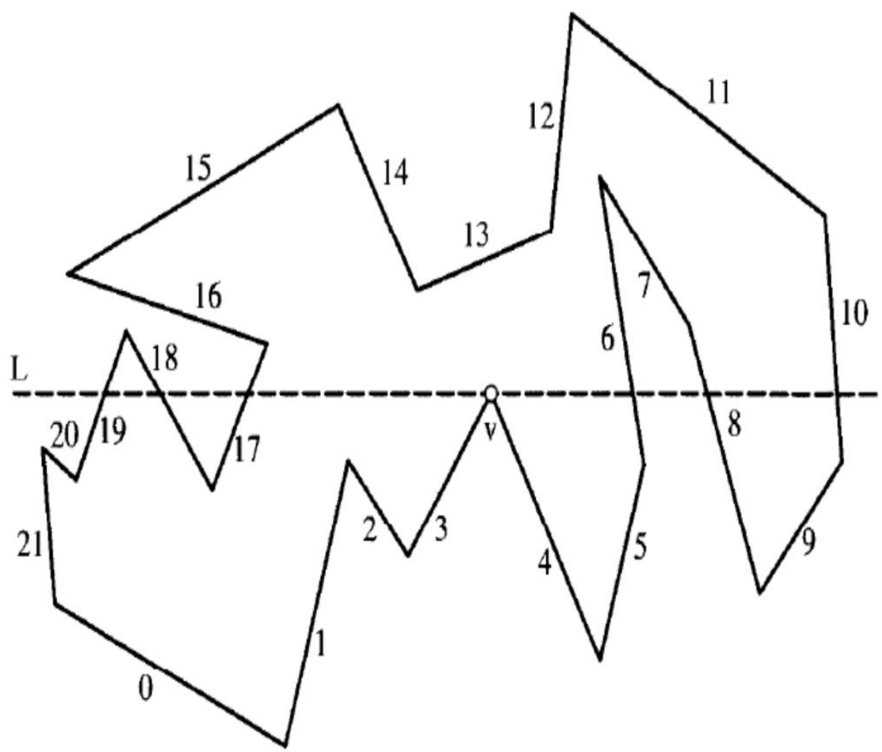
Idea of the Algorithm for trapezoidalization

- Uses a technique : Line Sweep (Plane Sweep)
- Line sweep (Nievergelt & Preparata 1982)
- Sweep a horizontal line over the plane maintaining a data structure along the line



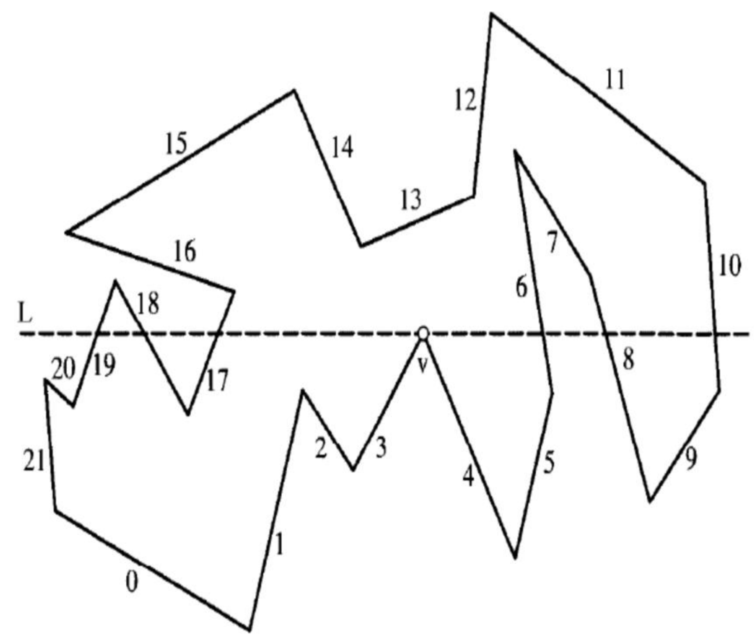
Line Sweep

- The horizontal line L sweeps downward stopping at each vertex
- The sweep stops at discrete events and the data structure is updated



Updating the data structure

- The processing required at each vertex is finding the edge immediately to the left and immediately to the right of v along L
- To do this efficiently, a sorted list (LIST) of polygon edges pierced by L is maintained all times
- Hence, the vertices should be sorted with respect to x coordinate (For sorting : O(n logn))



References

- J. O'Rourke: Art Gallery Theorems and Algorithms
- J. O Rourke, *Computational Geometry in C,* 2/e, Cambridge University Press, 1998 *)*
- *https://www.cs.jhu.edu/~misha/Spring16/05.pdf*
  - *From John Hopkins University*

Thank you