CSCI 4560/6560 Computational Geometry

https://www.cs.rpi.edu/~cutler/classes/computationalgeometry/S22/

Lecture 5: Triangulation, part 2

Outline for Today

- Homework 2 Posted
- Last Time: Art Gallery Problem & Triangulation
- Improved Triangulation Algorithm
- Definition: Monotone Polygon
- Splitting into Monotone Polygons
- Triangulating a Monotone Polygon
- Analysis of Improved Triangulation Algorithm
- Future Lecture: Additional Triangulation Goals

Homework 2

- Use CGAL's Surface Mesh (Halfedge) data structure

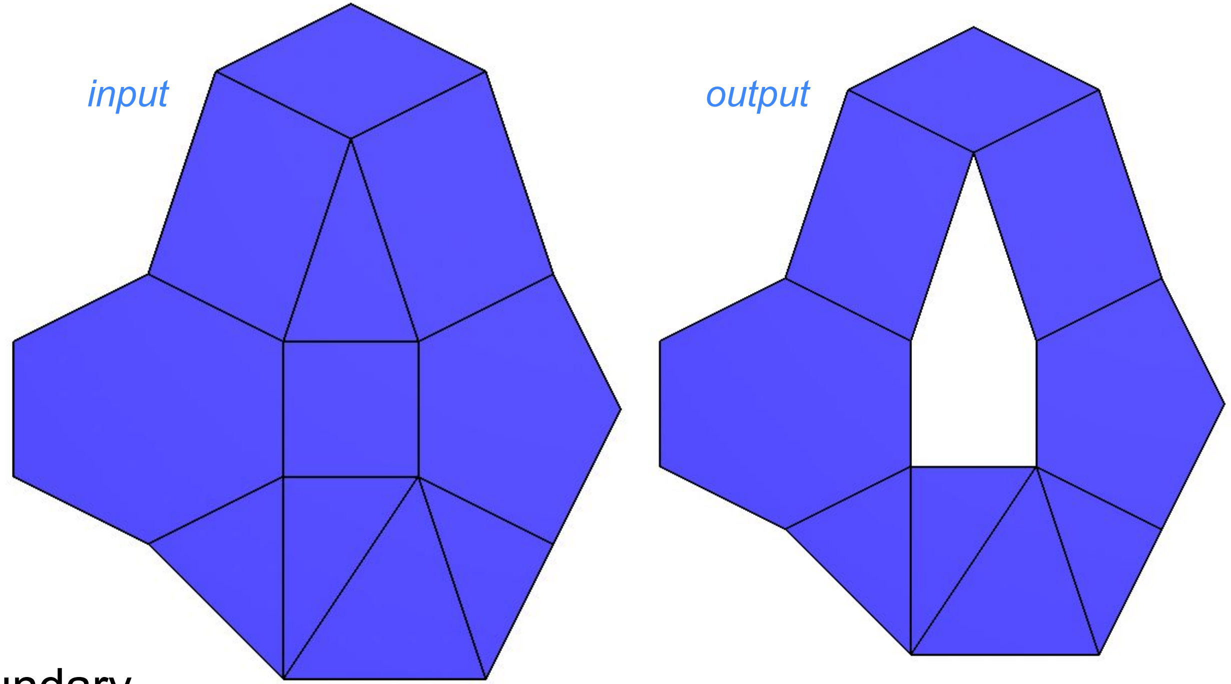
---

use case 1
- Input: all edges
- Output: all faces on any boundary

---

use case 2
- Input: 1 edge on a boundary
- Output: all faces on that boundary

- *Posted late… deadline extended until Monday 1/31, but please make progress before Friday, so we can discuss questions :)*



Homework 2

- Each Halfedge stores:
  - **vertex** at end of directed edge
  - **symmetric** halfedge
  - **face** to left of edge
  - **next** points to the Halfedge counter-clockwise around face on left

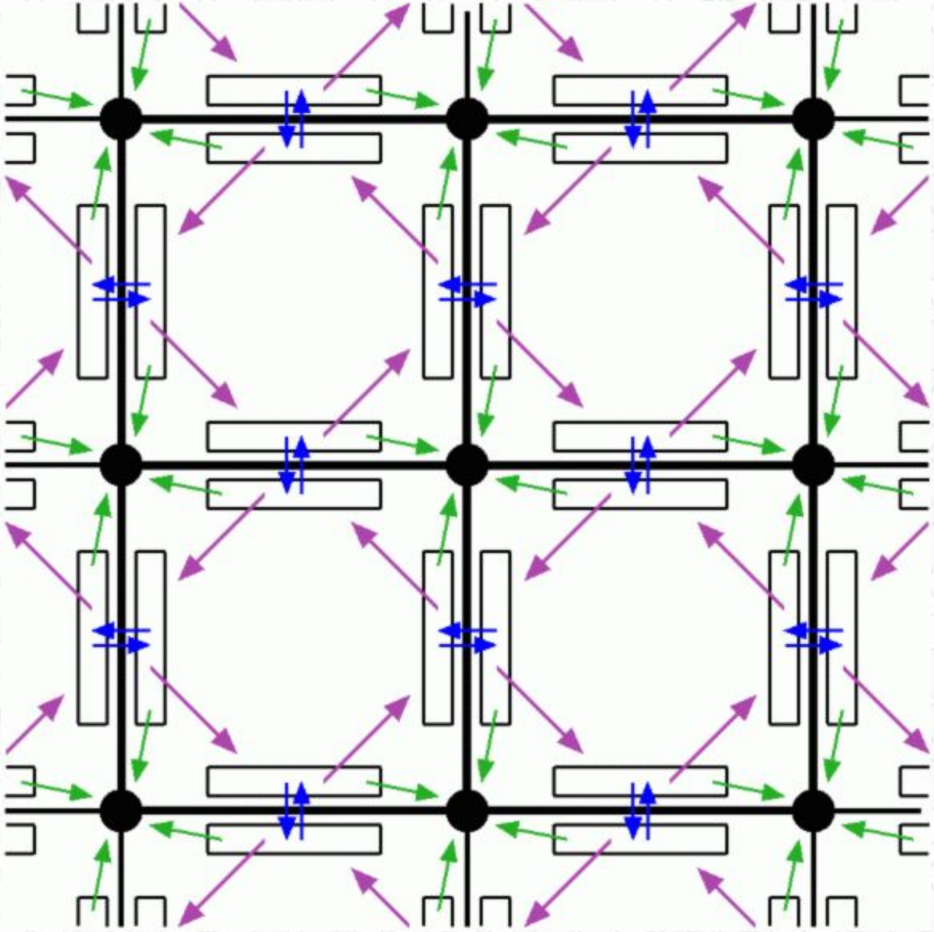


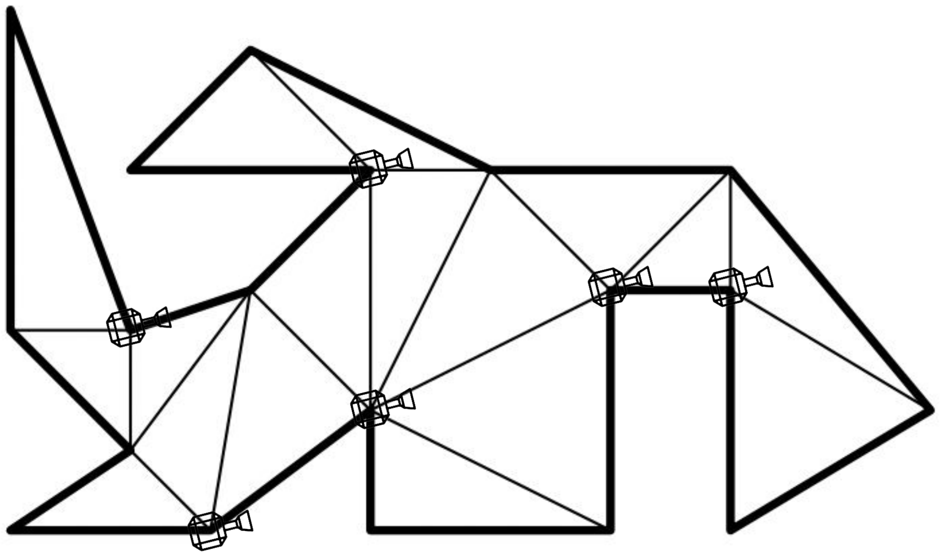
Image from Justin Legakis

Outline for Today

- Homework 2 Posted
- Last Time: Art Gallery Problem & Triangulation
- Improved Triangulation Algorithm
- Definition: Monotone Polygon
- Splitting into Monotone Polygons
- Triangulating a Monotone Polygon
- Analysis of Improved Triangulation Algorithm
- Future Lecture: Additional Triangulation Goals

Last Time?

- The Art Gallery Problem: Place cameras for 100% coverage of a simple polygon (no interior holes).
- Triangulate, and place cameras on the ~⅓ of the vertices, ensuring every triangle has one vertex with a camera.

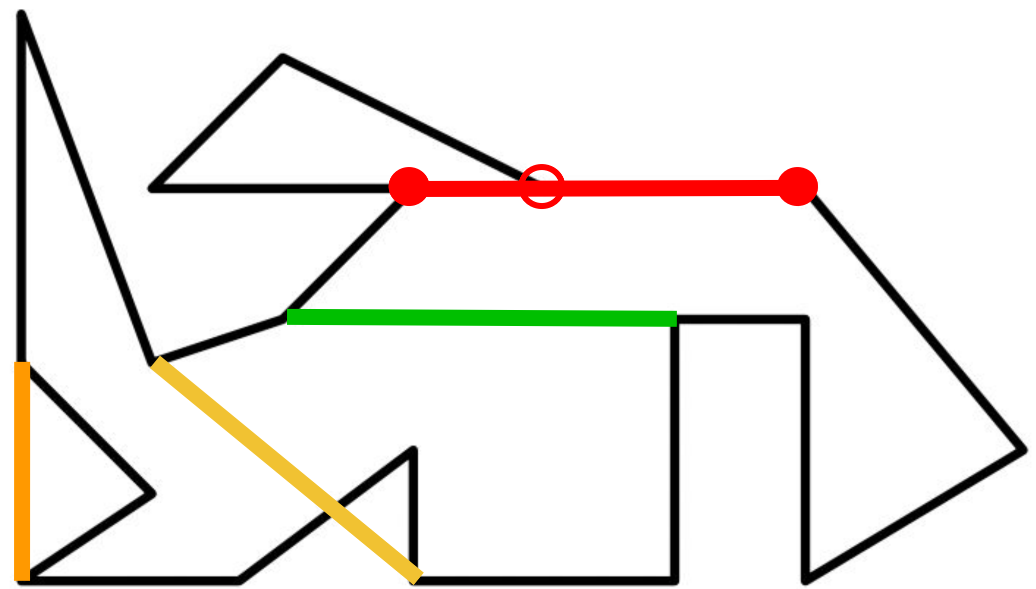


Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Cut the input on a “Diagonal” & Recurse

- Diagonal should connect two non-adjacent vertices on the polygonal boundary.

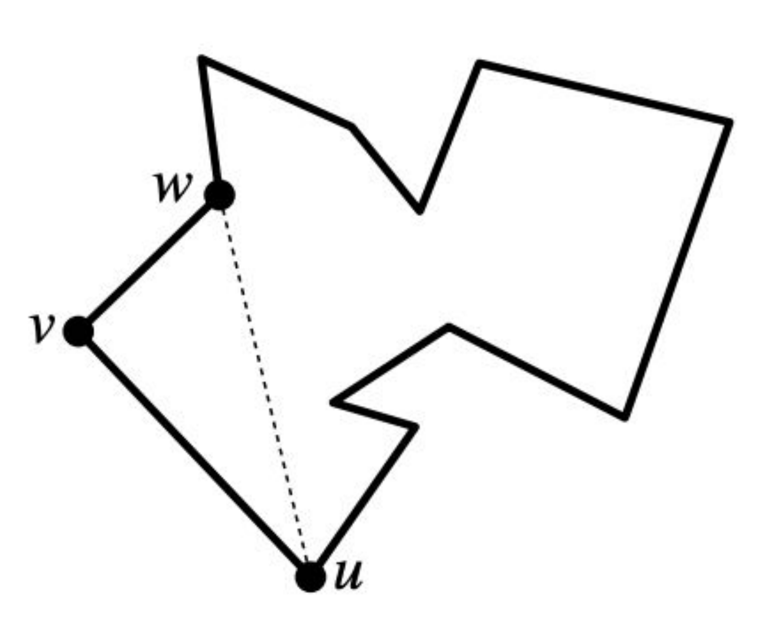
- Diagonal must not be outside the polygon.
- Diagonal may not cross any edge.
- Diagonal should not pass through any other vertex.



How do we find a Valid Diagonal?

- Start at the leftmost vertex, *v*
  - *NOTE: If two or more vertices have the sar*
- Find vertices *u* and *w,* adjacent to *v*

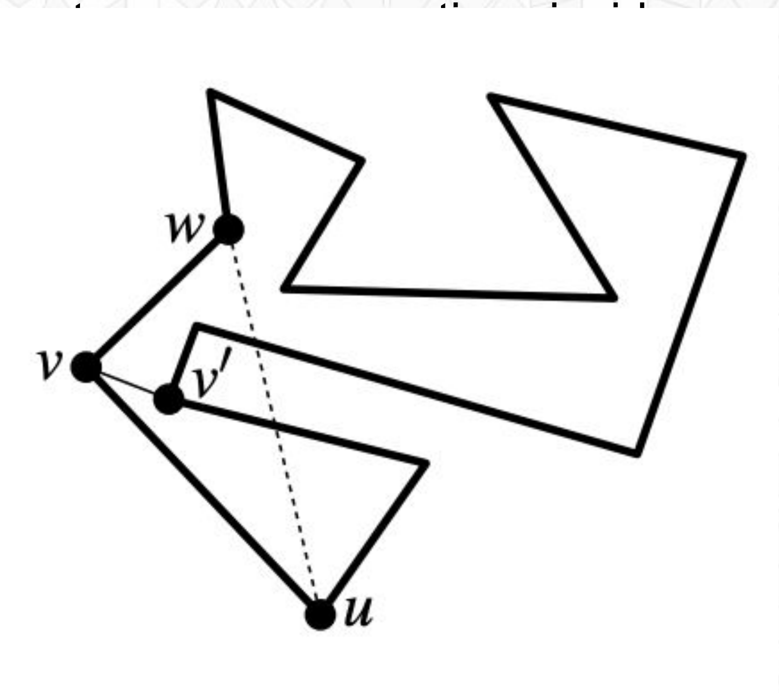
- Check if the line *uw* is a valid diagonal.
  - This line does not pass through *v.*
  - Does it intersect other line segments?
  - Does it pass through any other vertices?
  - Does it lie completely outside of the polygo is the rightmost vertex)



Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

How do we find a Valid Diagonal?

- If it does cross another line segment, there m
  the triangle *uvw.*
- Starting at the intersection, walk along the bo
- Choose the vertex *v'*,
  furthest from the line segment *uw*
- Draw the diagonal from *v* to *v'*



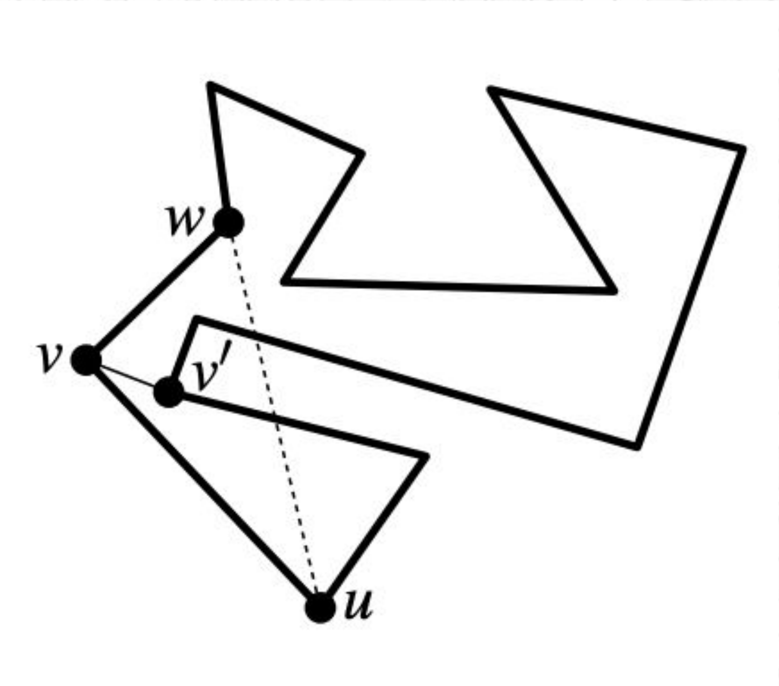
Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Cut on Diagonal & Recurse Analysis

- What is the worst case running time to triangulate a non-convex, simple polygon with n vertices?

- Identify a legal diagonal
  - O(n) in worst case
- Split into two smaller polygons
  - Worst case:
    m_1 = 3 vertices and
    m_2 = n-1 vertices

- Overall: $O(n^2)$ running time

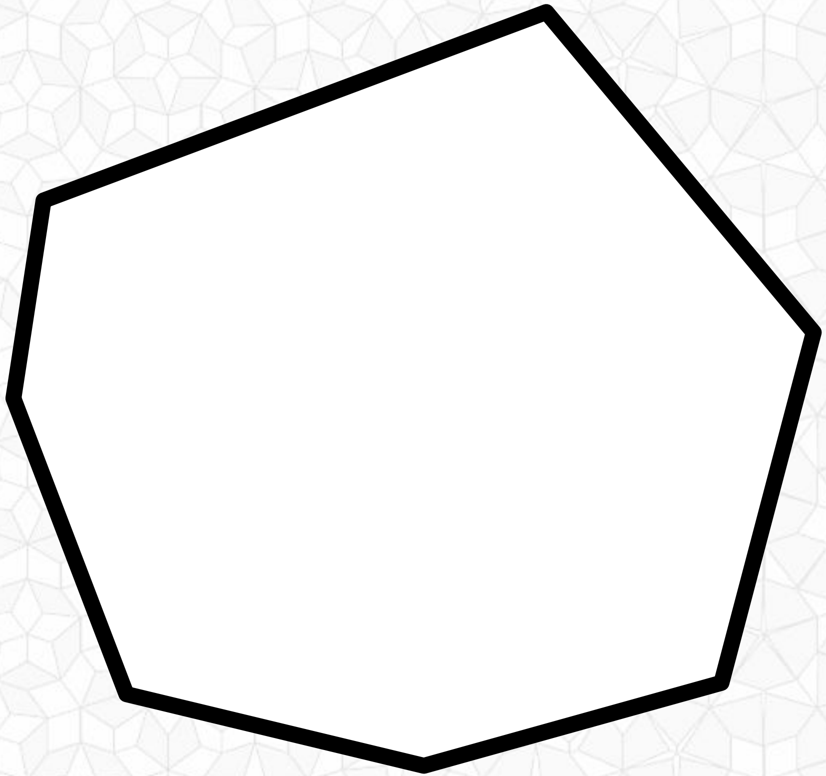


Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Outline for Today

- Homework 2 Posted
- Last Time: Art Gallery Problem & Triangulation
- Improved Triangulation Algorithm
- Definition: Monotone Polygon
- Splitting into Monotone Polygons
- Triangulating a Monotone Polygon
- Analysis of Improved Triangulation Algorithm
- Future Lecture: Additional Triangulation Goals

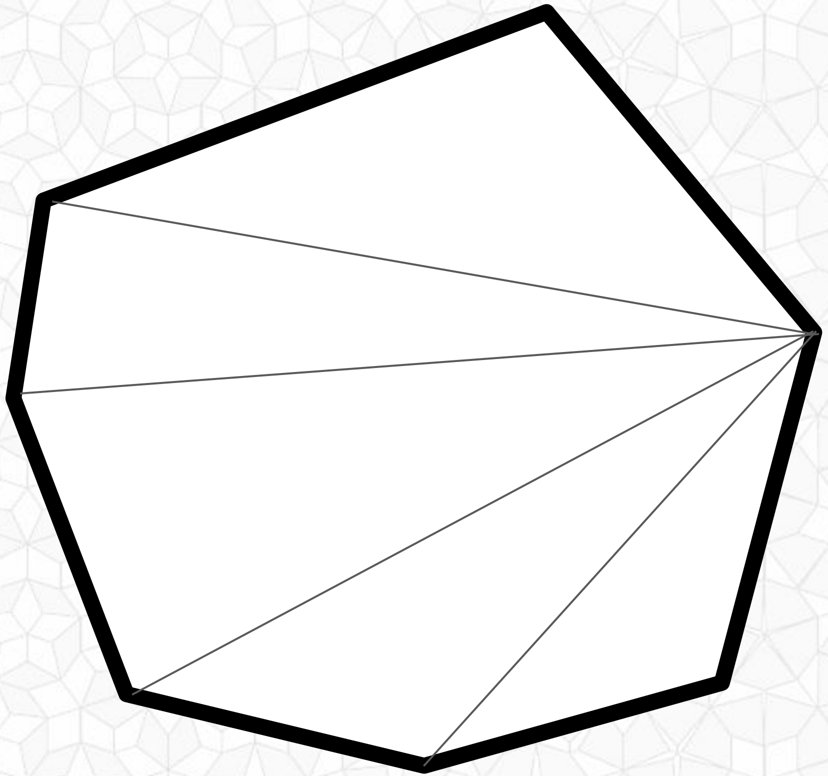
A Convex Polygon is easy to Triangulate



A Convex Polygon is easy to Triangulate

- Pick any vertex and connect it to every other vertex (except 2 adjacent vertices

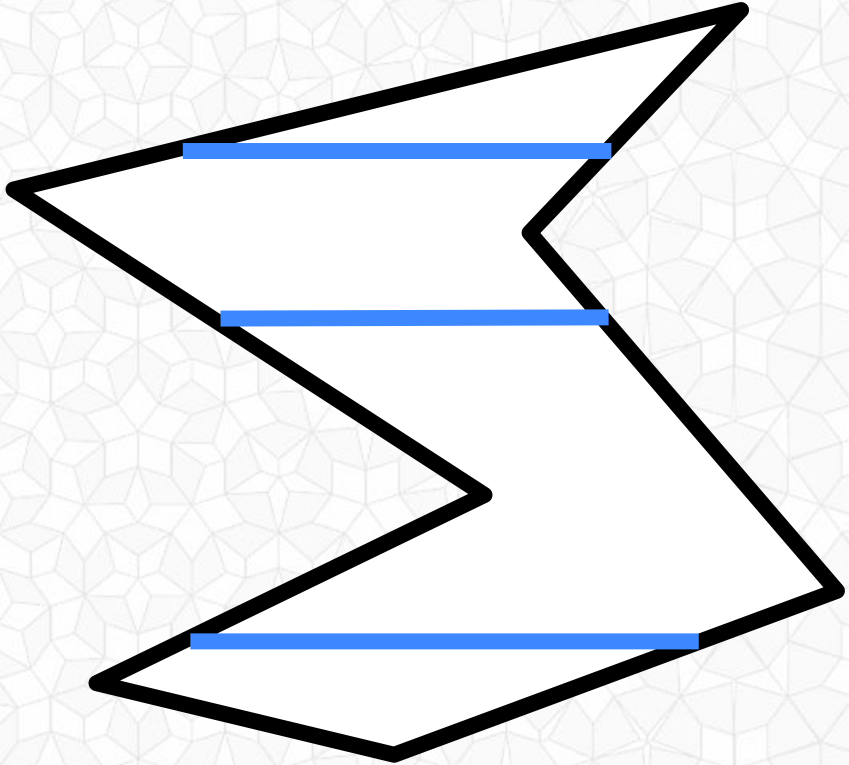
- *Unfortunately, breaking a non-convex polygon into convex polygons is not easy.*



Definition: Monotone with Respect to Y-Axis

- The intersection of the polygon with any line perpendicular to the y-axis is connected.

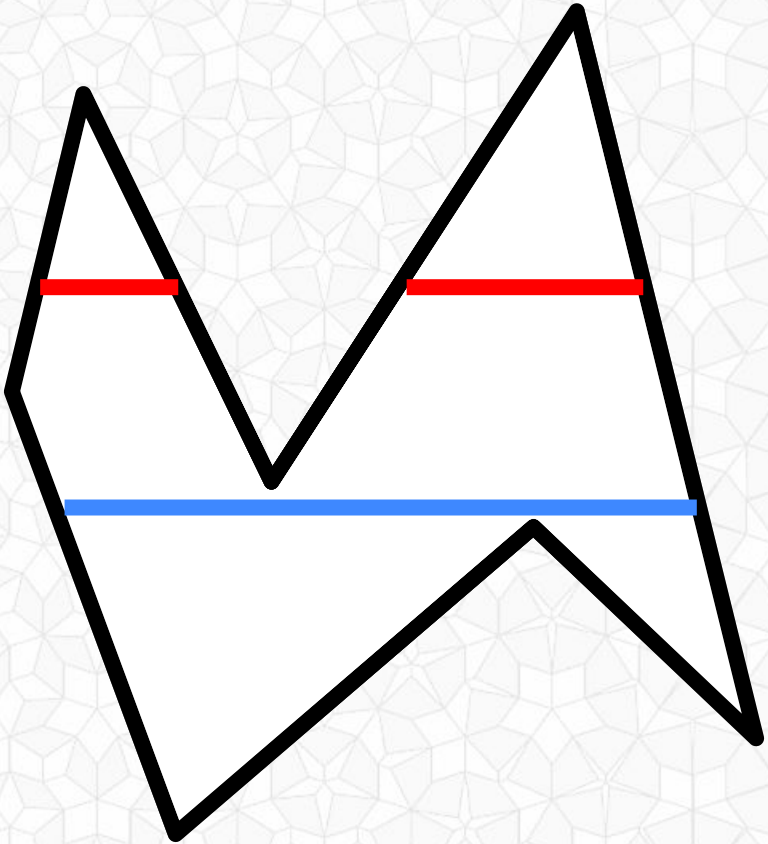
- The intersection is either
  - empty (above or below the polygon),
  - one point (top or bottom vertex), or
  - *a line segment.*



Not Monotone, with Respect to Y-Axis

- The intersection of the polygon with any line perpendicular to the y-axis is connected.

- The intersection is either
  - empty (above or below the polygon),
  - one point (top or bottom vertex), or
  - *a line segment.*

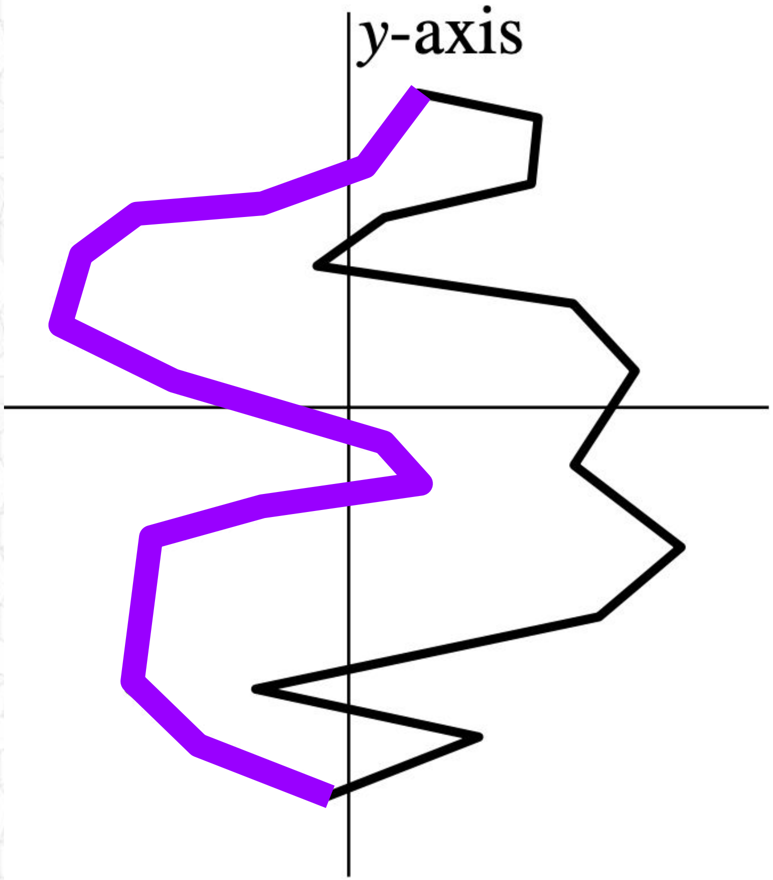


If a Polygon is Monotone…

- We can start from the top vertex (largest y coordinate), **and walk "down" the left side** to the bottom vertex (smallest y coordinate)

  Each step moves downwards or horizontally – *never upwards.*

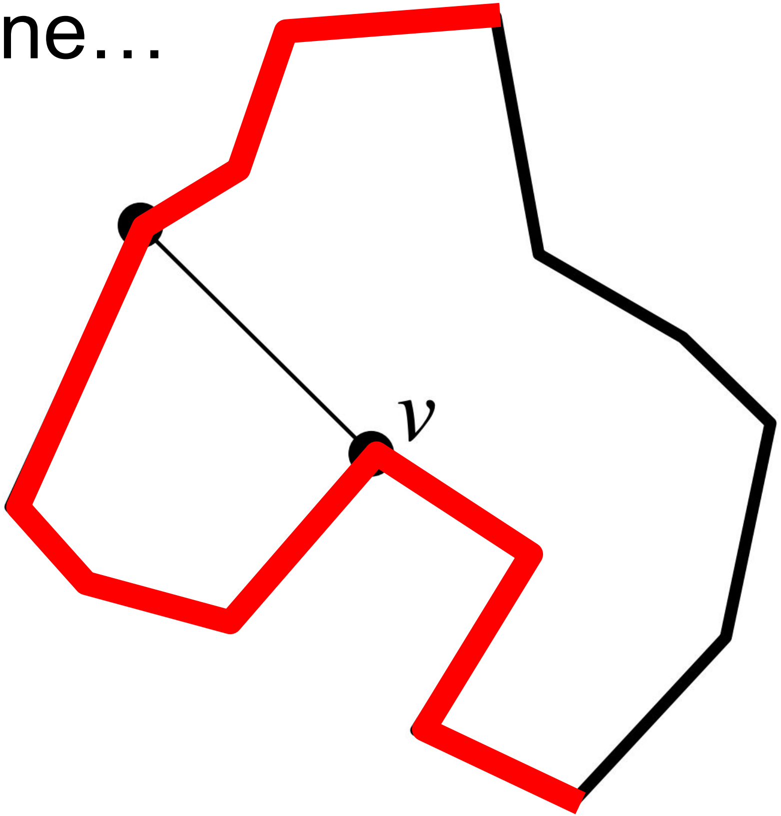
- Similarly we can walk down the right side of the polygon.



Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

This Polygon is not Monotone…

- The left side of this polygon ***does not monotonically decrease***
- We'll need to break this polygon into pieces…
- At vertex *v* – a "turn vertex"!

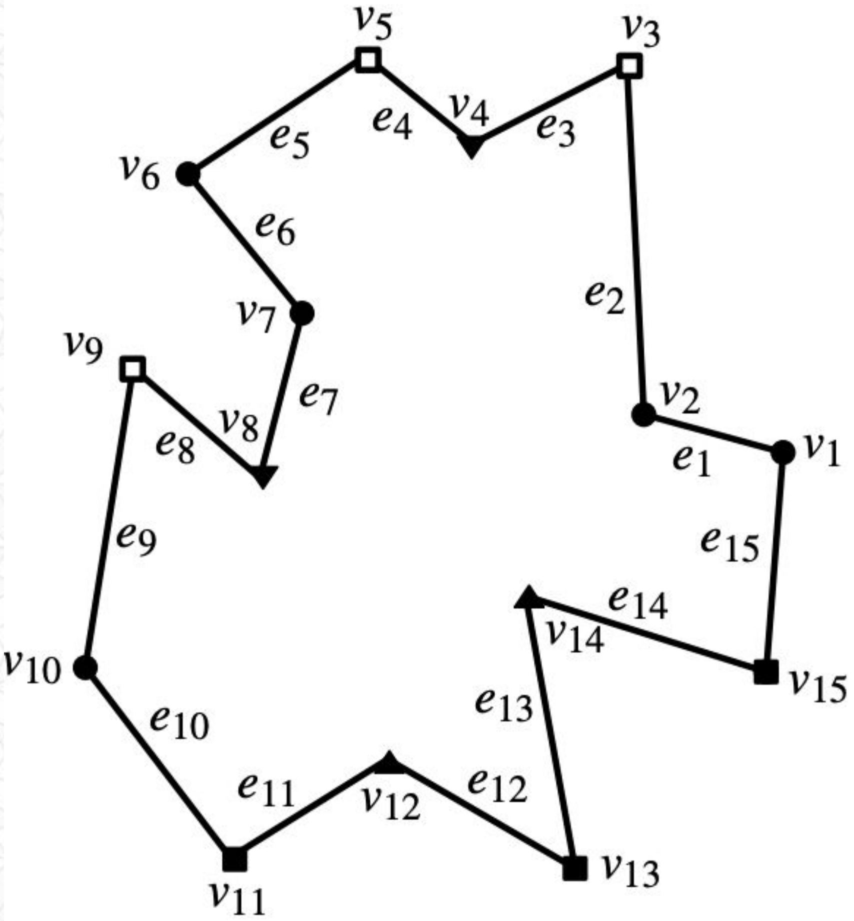


Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Outline for Today

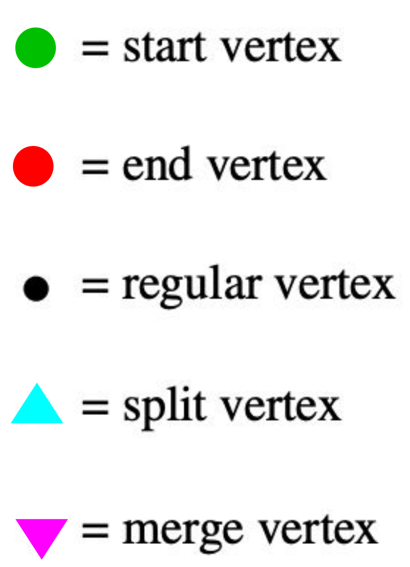
- Homework 2 Posted
- Last Time: Art Gallery Problem & Triangulation
- Improved Triangulation Algorithm
- Definition: Monotone Polygon
- Splitting into Monotone Polygons
- Triangulating a Monotone Polygon
- Analysis of Improved Triangulation Algorithm
- Future Lecture: Additional Triangulation Goals

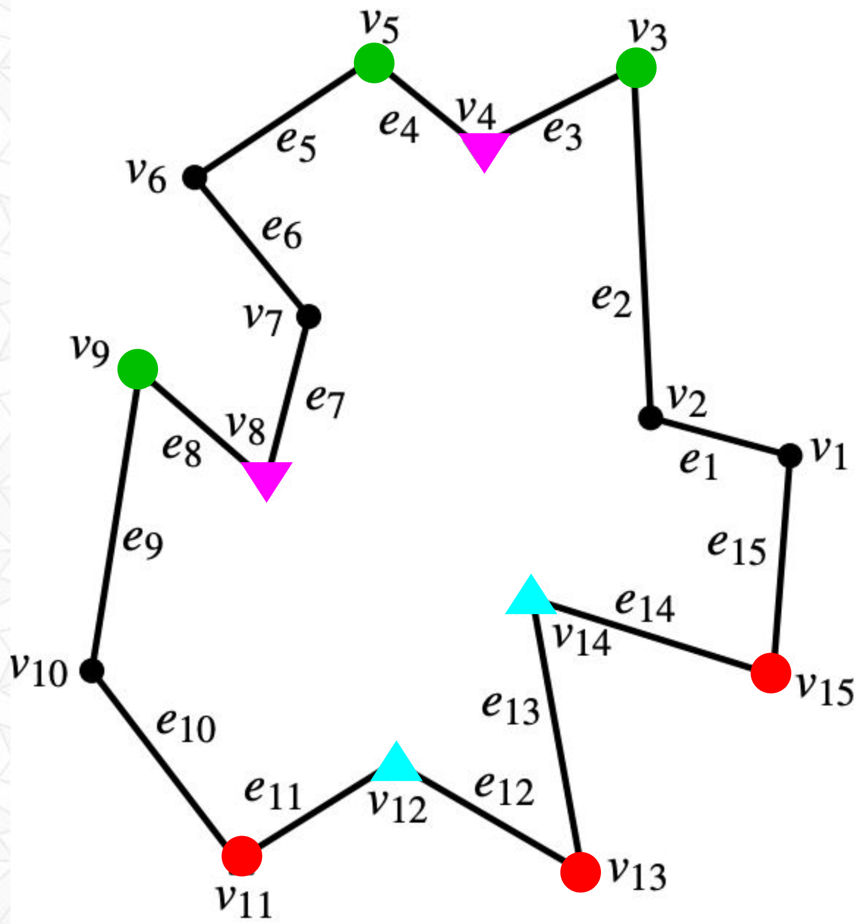
Identify Vertex Types



Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Identify Vertex Types

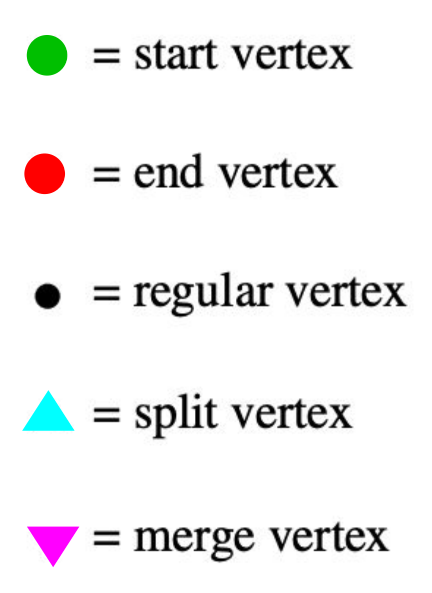


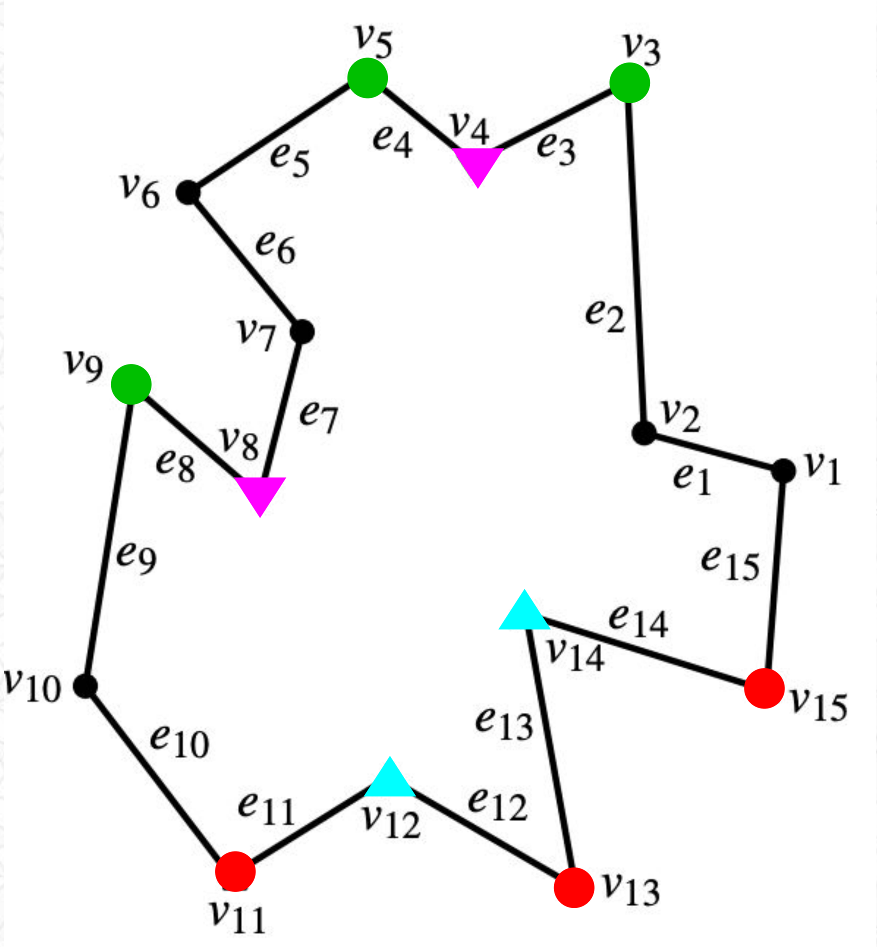


Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Identify Vertex Types

- Direction (up or down) of adjacent edges
- Interior angle at vertex (> 180° or < 180°)





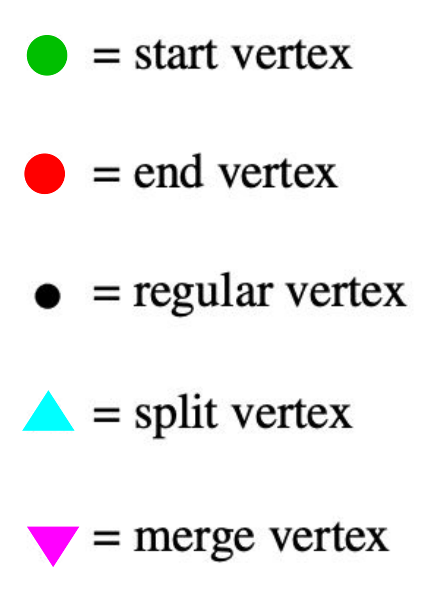
Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

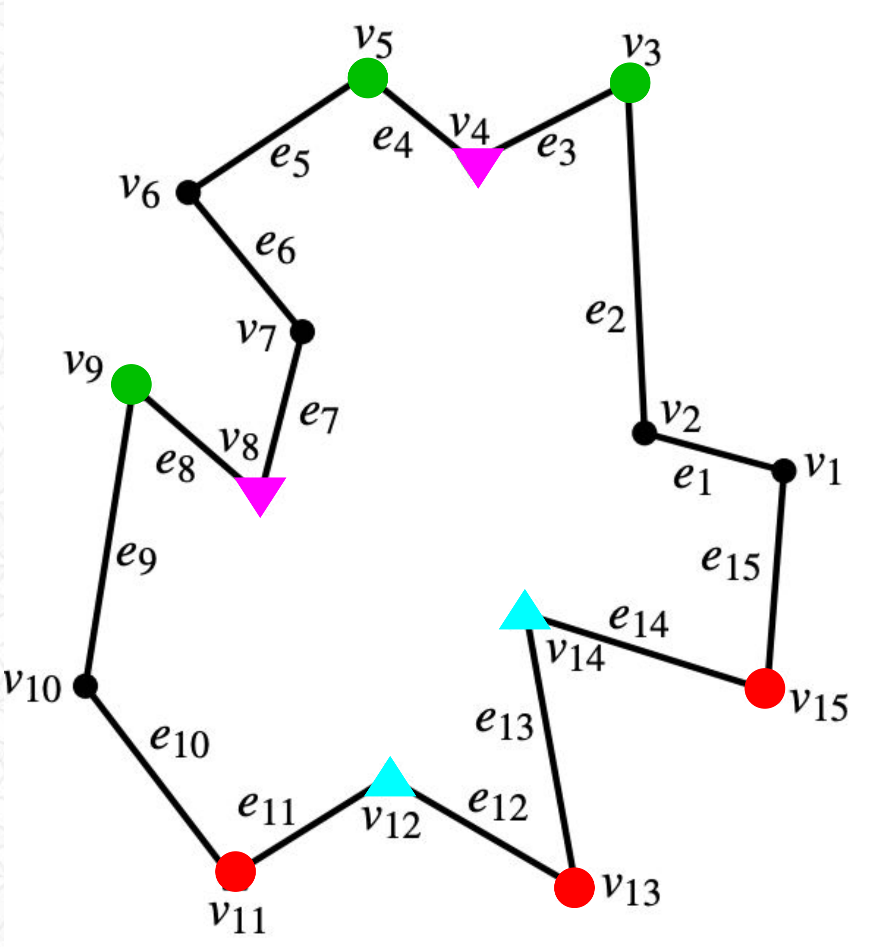
DEGENERACY NOTE

"Break Ties" consistently

- p is "below" q if $p_y < q_y$ or $p_y = q_y$ and $p_x > q_x$
- p is "above" q if $p_y > q_y$ or $p_y = q_y$ and $p_x < q_x$

Lemma 3.4: A polygon is *y*-monotone if it has no split vertices or merge vertices.

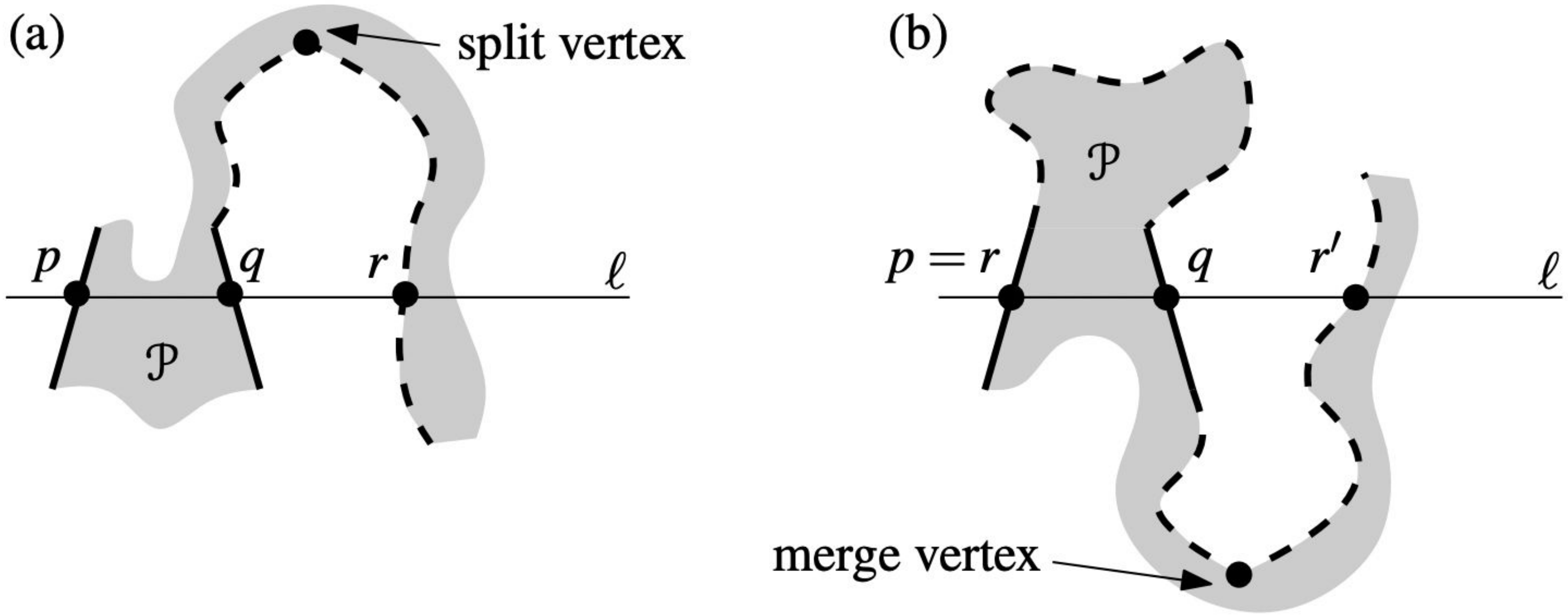




Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

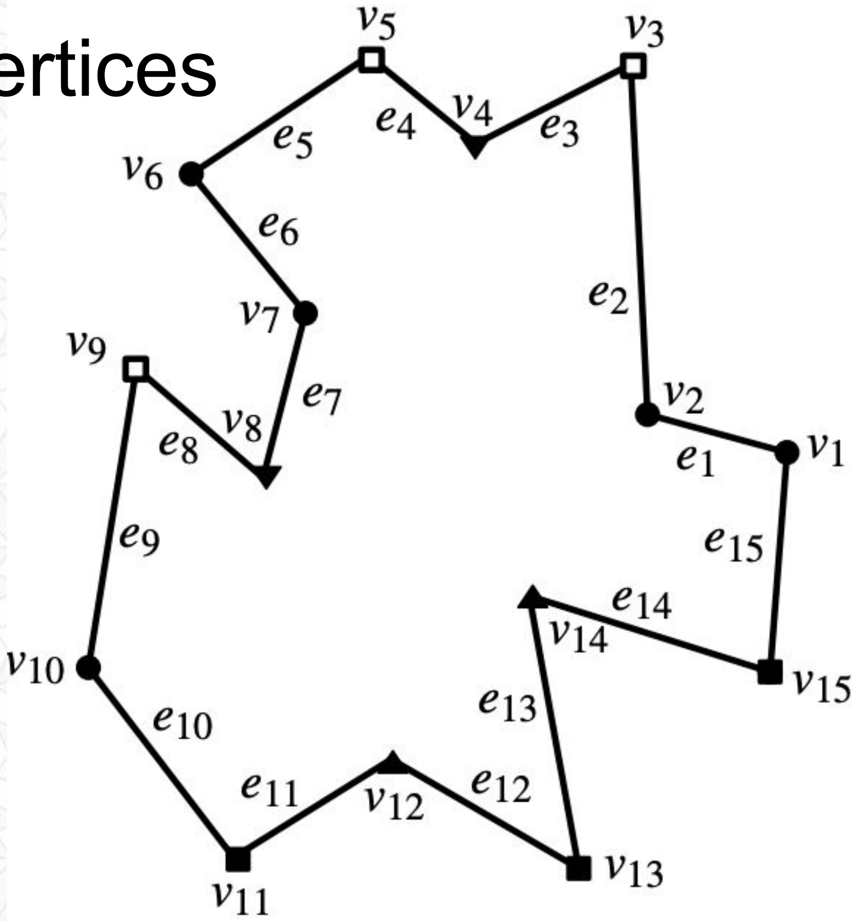
Lemma 3.4: A polygon is *y*-monotone if it has no split vertices or merge vertices.

A connected shape that crosses a horizontal sweep line at 3 points must either have a split vertex or a merge vertex!



Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Eliminate Merge & Split Vertices

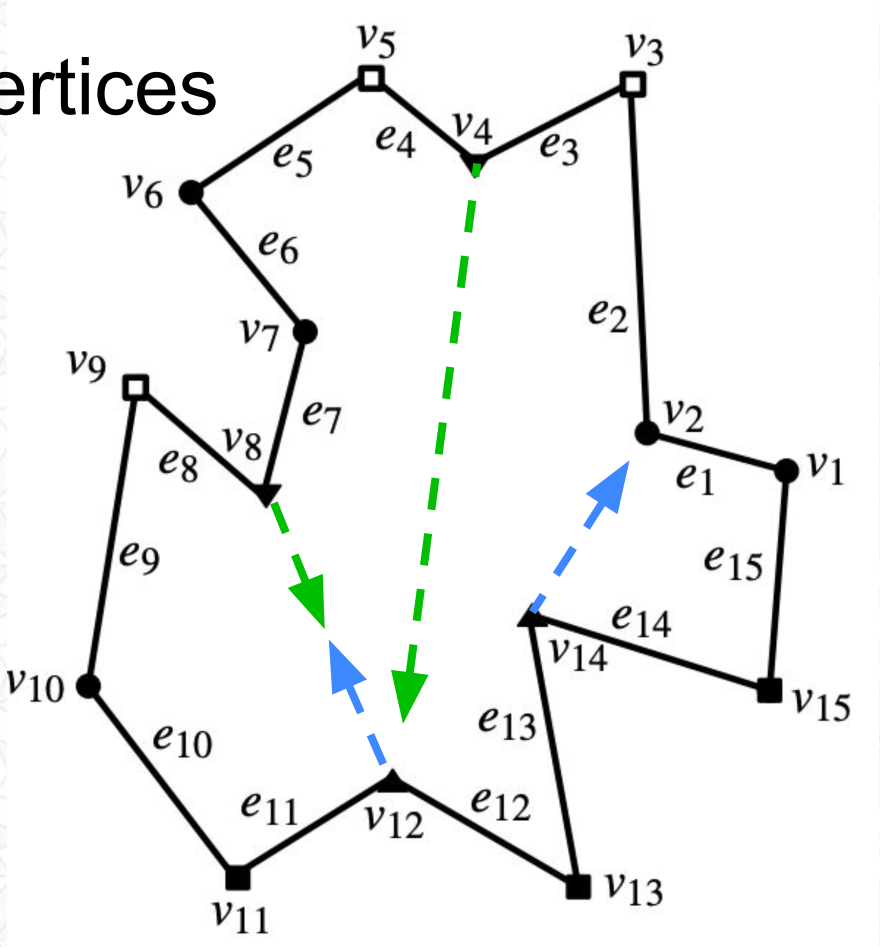


Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Eliminate Merge & Split Vertices

- Cut polygon on a diagonal going upwards from every split vertex.
- And downwards from every merge vertex.

- *Make sure these diagonals don't intersect the polygon or another diagonal!*



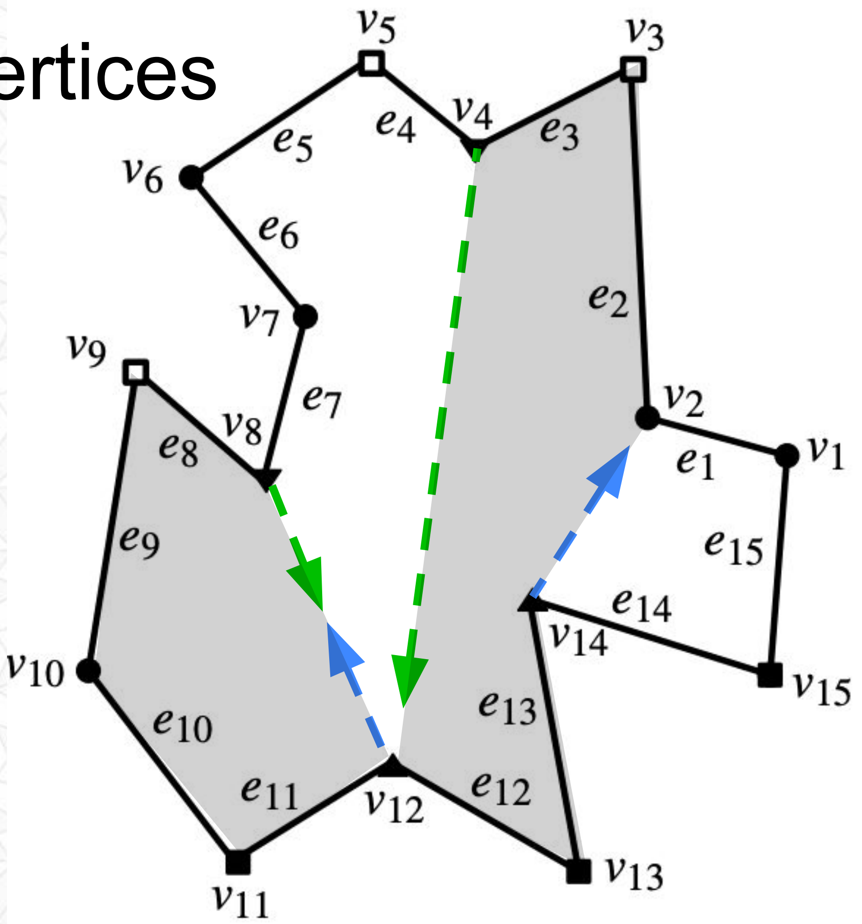
Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Eliminate Merge & Split Vertices

- Cut polygon on a diagonal going upwards from every split vertex.
- And downwards from every merge vertex.

- *Make sure these diagonals don't intersect the polygon or another diagonal!*

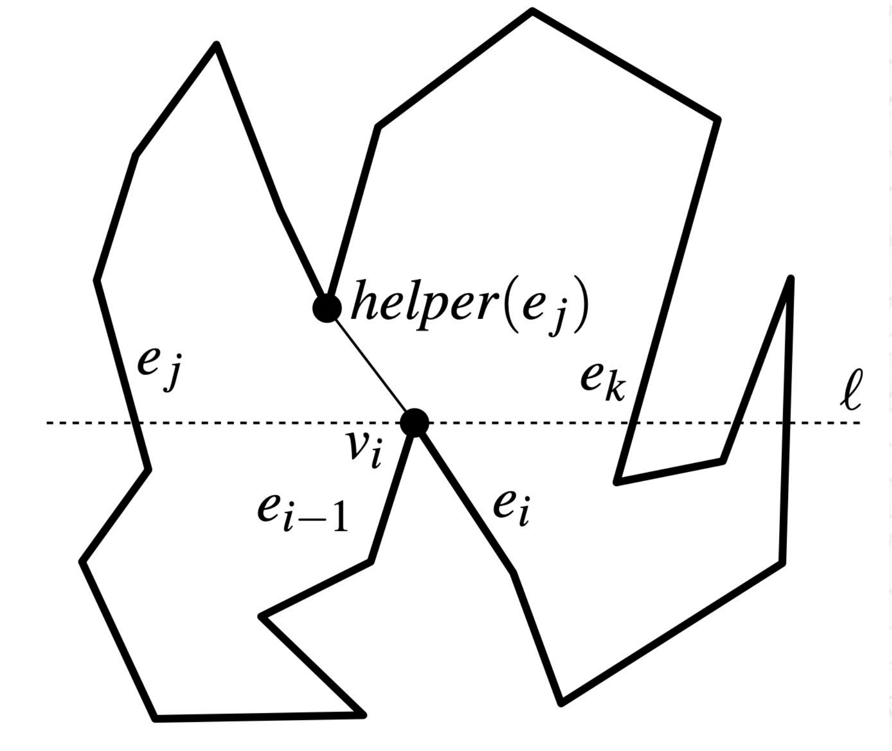
- *End result is monotone polygons!*



Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

How do we decide what to connect them to?

- Perform line sweep from top to bottom
- When we find split vertex v_i,
  connect it to a vertex above us…
- Which vertex?

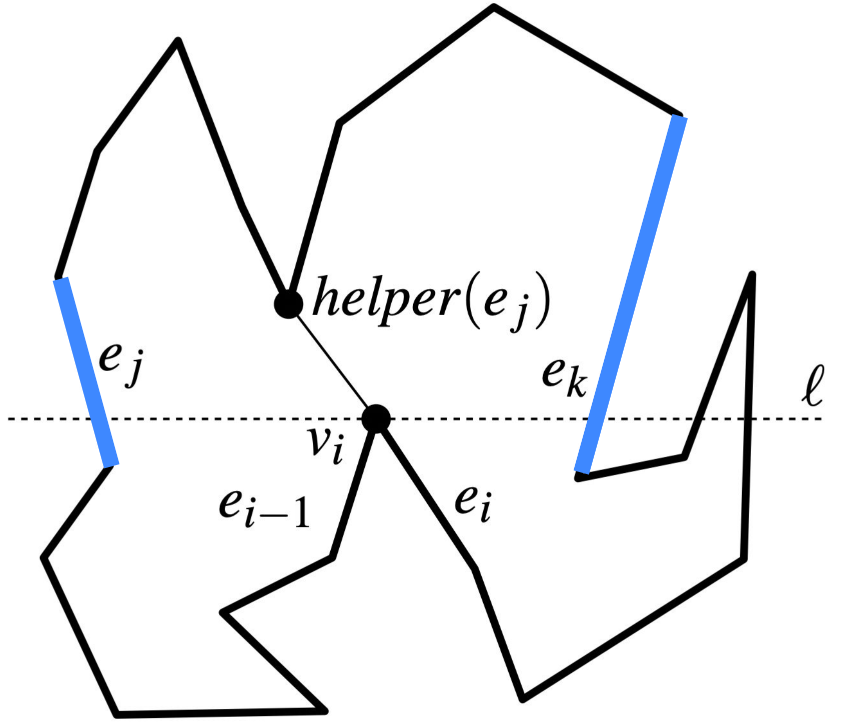


Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

How do we decide what to connect them to?

- Perform line sweep from top to bottom
- When we find split vertex v_i, connect it to a vertex above us…
- Which vertex?

- Find **line to left, e_j, and to right, e_k,** of v_i on the current sweep line.
- Locate the lowest point between these two lines (a merge vertex)
- If none, take the upper end point of edge e_j or edge e_k

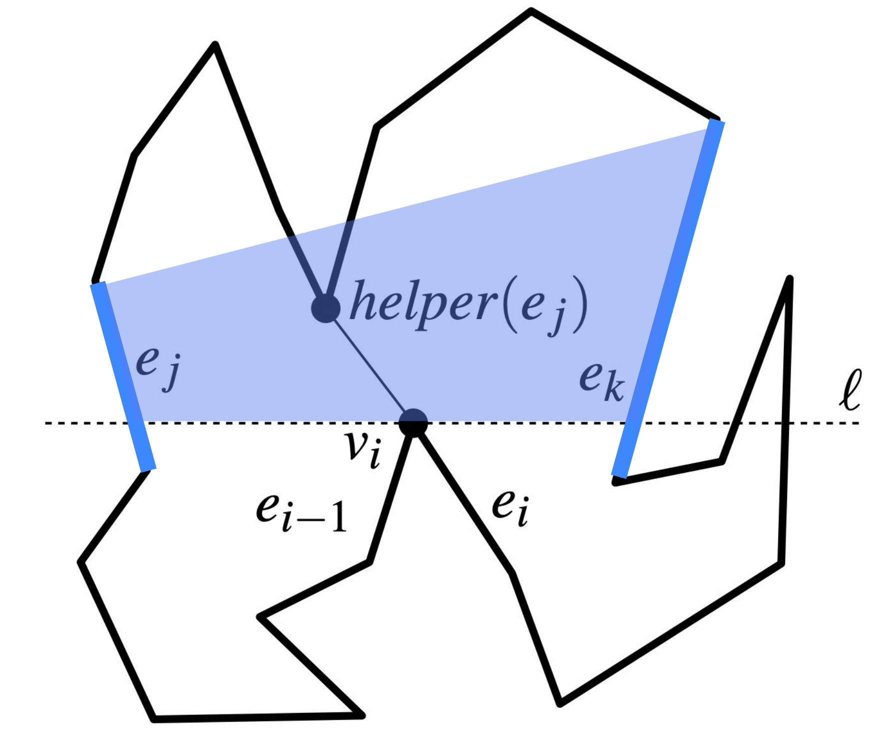


Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

How do we decide what to connect them to?

- Perform line sweep from top to bottom
- When we find split vertex v_i, connect it to a vertex above us…
- Which vertex?

- Find **line to left, e_j, and to right, e_k,** of v_i on the current sweep line.
- Locate the lowest point between these two lines (a merge vertex)
- If none, take the upper end point of edge e_j or edge e_k

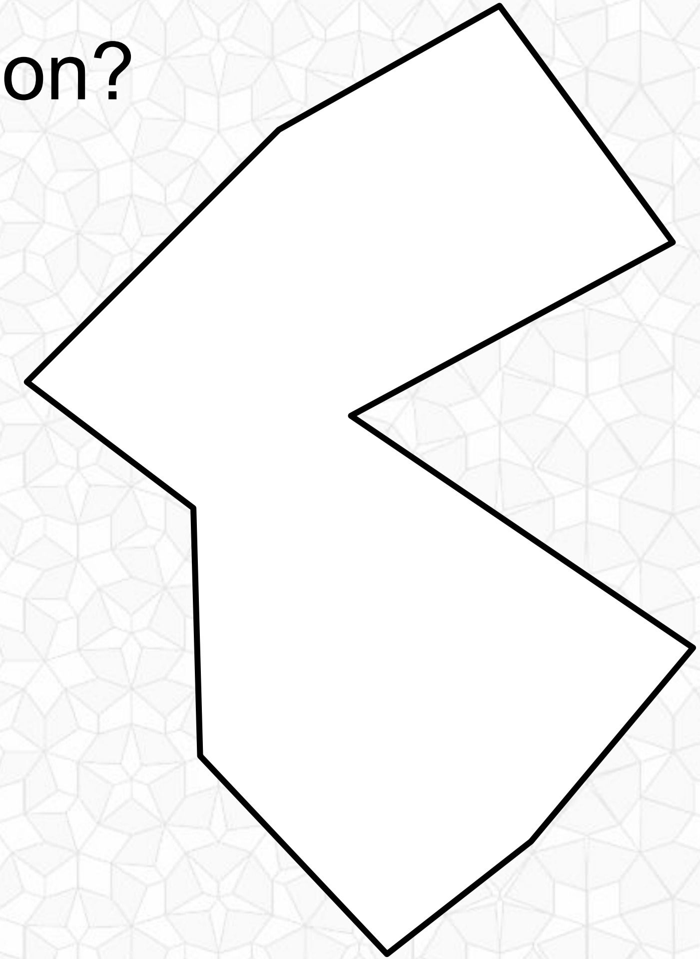


Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Outline for Today

- Homework 2 Posted
- Last Time: Art Gallery Problem & Triangulation
- Improved Triangulation Algorithm
- Definition: Monotone Polygon
- Splitting into Monotone Polygons
- Triangulating a Monotone Polygon
- Analysis of Improved Triangulation Algorithm
- Future Lecture: Additional Triangulation Goals

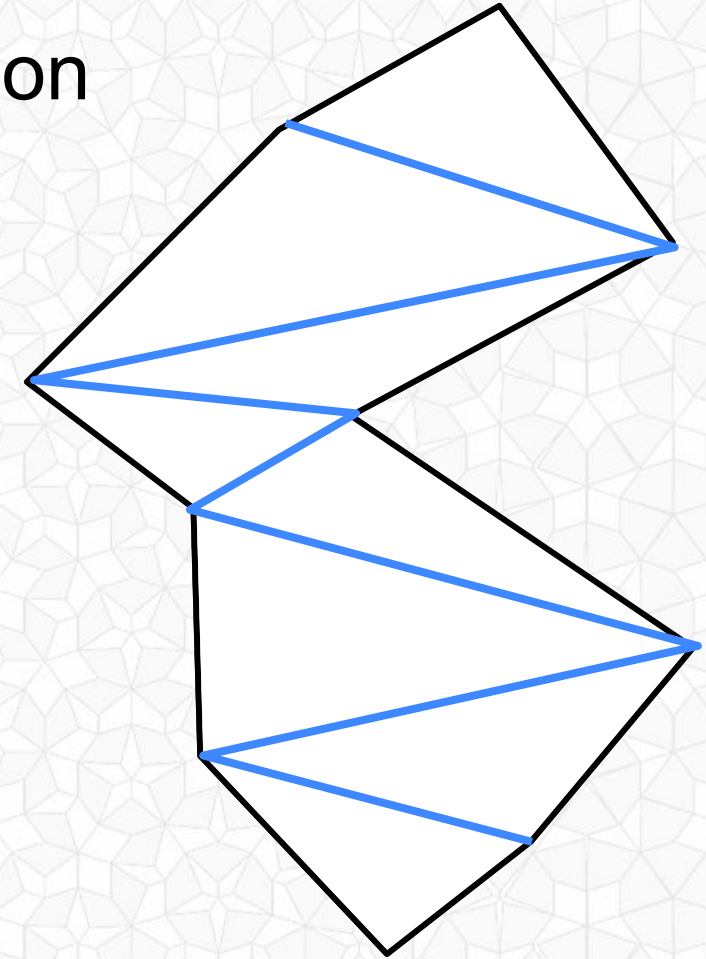
Triangulate a Monotone Polygon?



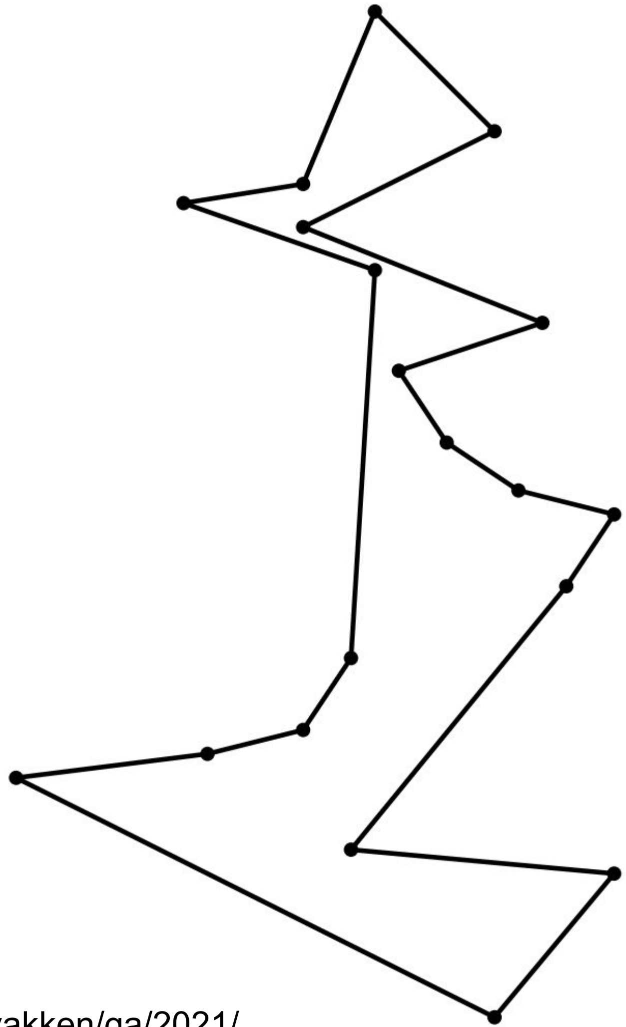
Triangulate a Monotone Polygon

- Can we always just draw a zig zag down the middle of a monotone polygon?

- Unfortunately no, it’s a little more complicated



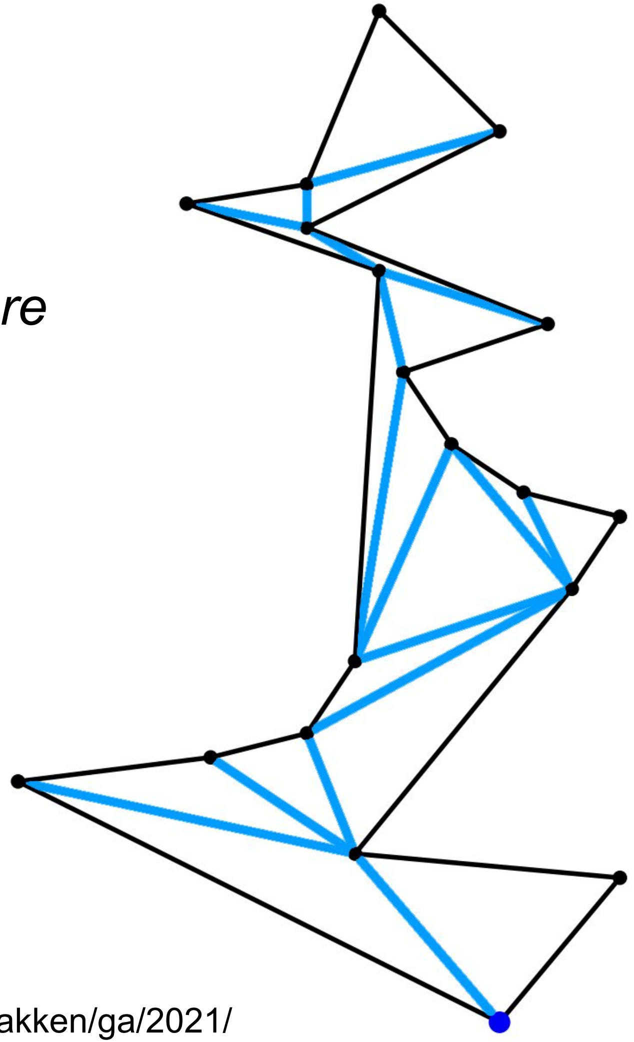
Triangulate a Monotone Polygon



Frank Staals, http://www.cs.uu.nl/docs/vakken/ga/2021/

Triangulate a Monotone Polygon

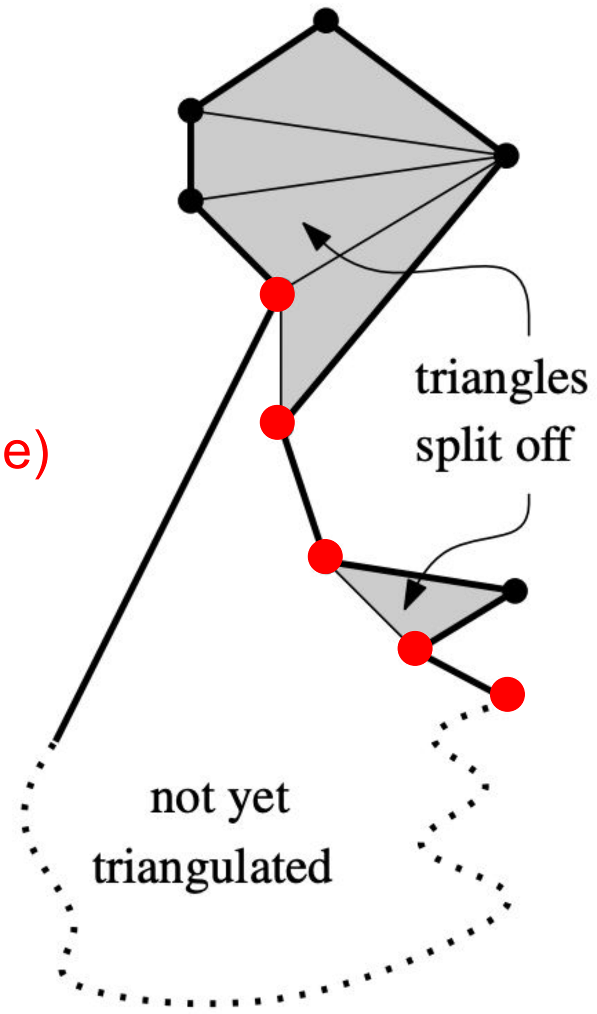
- Sort all of the points vertically
- Push top two points onto a *stack data structure*
- Process the remaining points,
  one at a time, from top to bottom
- If you can…
  - make a triangle with the new point
    and the last two points on the stack
  - & remove 1 point
  - & repeat
- If not, push the new point on the stack



Frank Staals, http://www.cs.uu.nl/docs/vakken/ga/2021/

Triangulate a Monotone Polygon

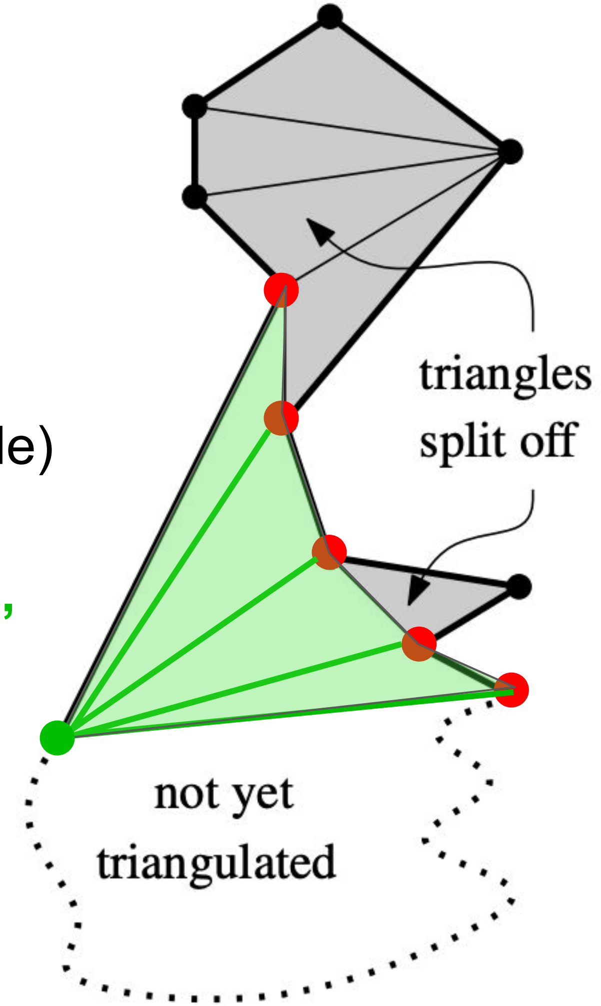
- **Vertices that have been finished**
- Triangles that have already been added
- Vertices currently on the stack form an "upside down funnel" on one side (e.g., right side)



Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 3

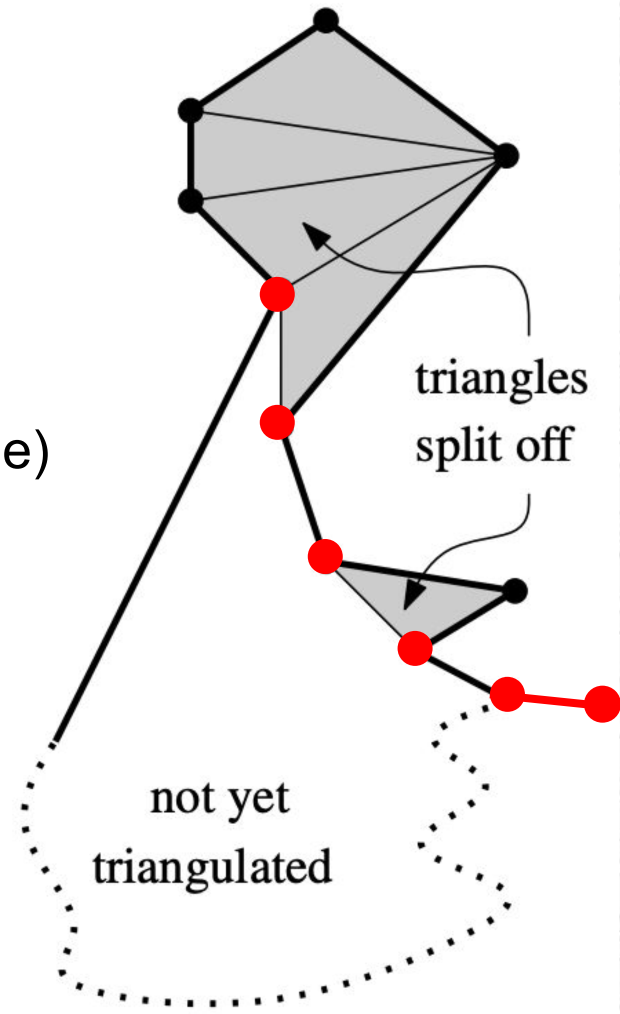
Triangulate a Monotone Polygon

- Vertices that have been finished
- Triangles that have already been added
- Vertices currently on the stack form an "upside down funnel" on one side (e.g., right side)
- The next vertex below us will:
  - **Be from the (left) side and create a "fan", Leaving only 2 vertices on the stack**



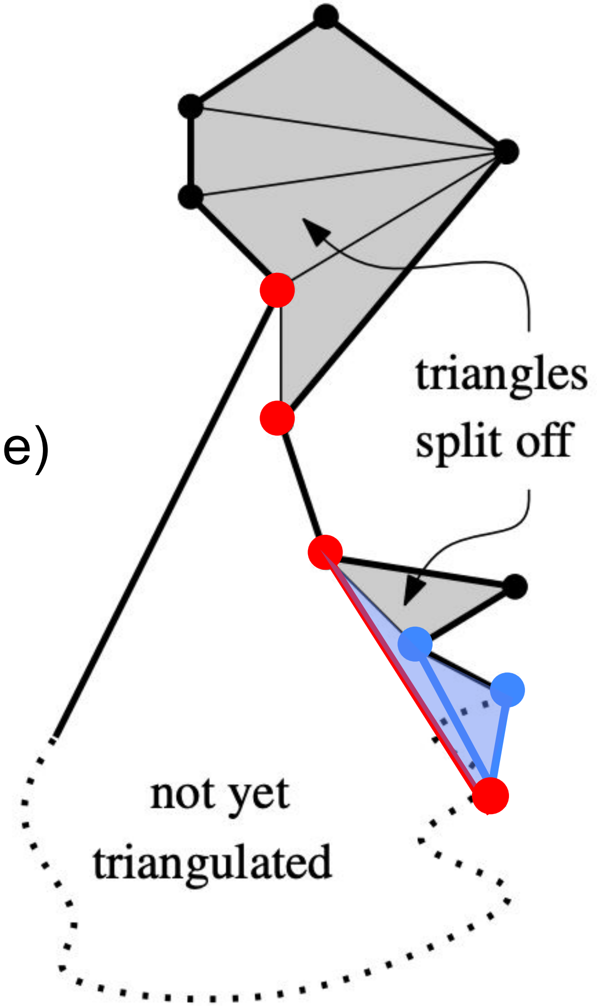
Triangulate a Monotone Polygon

- Vertices that have been finished
- Triangles that have already been added
- Vertices currently on the stack form an "upside down funnel" on one side (e.g., right side)
- The next vertex below us will:
  - Be from the (left) side and create a "fan", Leaving only 2 vertices on the stack
  - Be on the (right) side and:
    - **Bend the funnel further from vertical axis**



Triangulate a Monotone Polygon

- Vertices that have been finished
- Triangles that have already been added
- Vertices currently on the stack form an "upside down funnel" on one side (e.g., right side)
- The next vertex below us will:
  - Be from the (left) side and create a "fan", Leaving only 2 vertices on the stack
  - Be on the (right) side and:
    - Bend the funnel further from vertical axis
    - **Form one or more triangles**

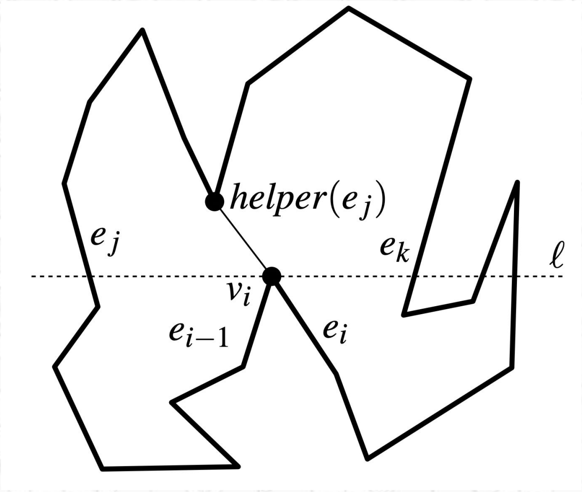


Outline for Today

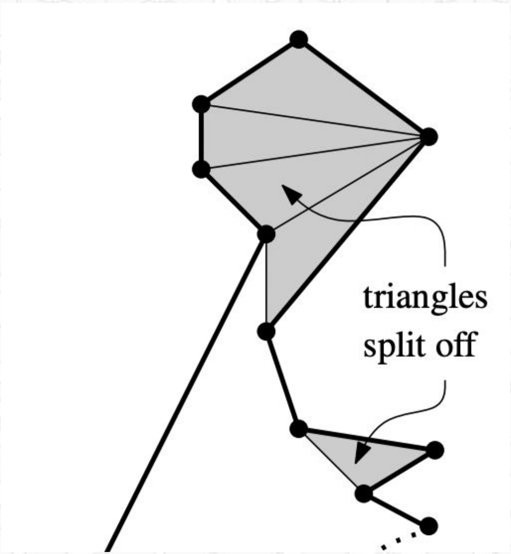
- Homework 2 Posted
- Last Time: Art Gallery Problem & Triangulation
- Improved Triangulation Algorithm
- Definition: Monotone Polygon
- Splitting into Monotone Polygons
- Triangulating a Monotone Polygon
- Analysis of Improved Triangulation Algorithm
- Future Lecture: Additional Triangulation Goals

Analysis?

- Line sweep algorithm: cut into monotone polygons



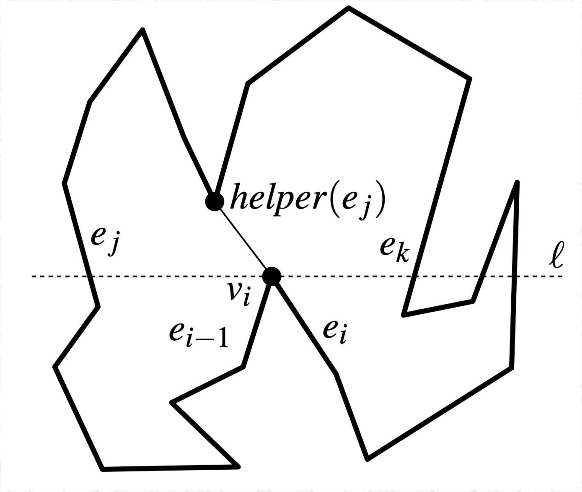
- Use stack to triangulate monotone polygon

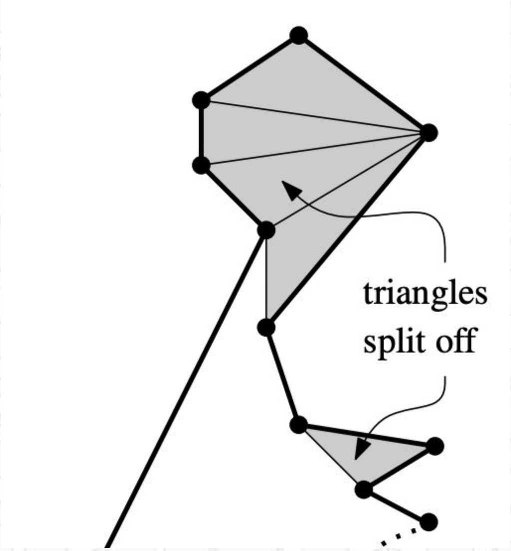


- Overall →

Analysis?

- Line sweep algorithm: cut into monotone polygons
    - Sort all vertices vertically -
    - Maintain horizontal sorting of active vertices -
    - Locate “helper” vertex for each split/merge -
    - →
- Use stack to triangulate monotone polygon
    - Don’t need to sort (just walk boundary)
    - Each vertex is added once -
    - Each vertex (beyond first two) adds one ck triangle when it is removed from stack -
    - →
- Overall →



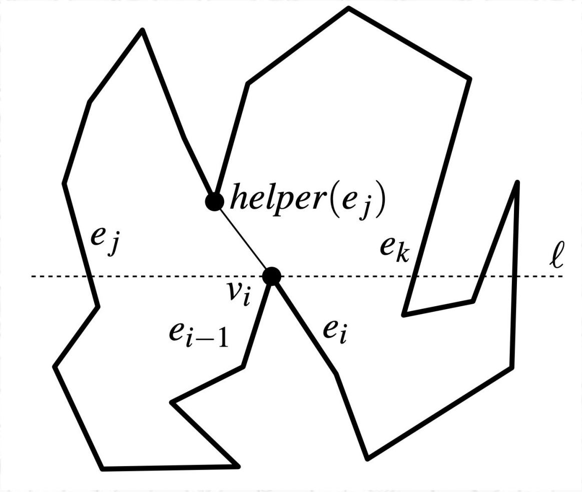


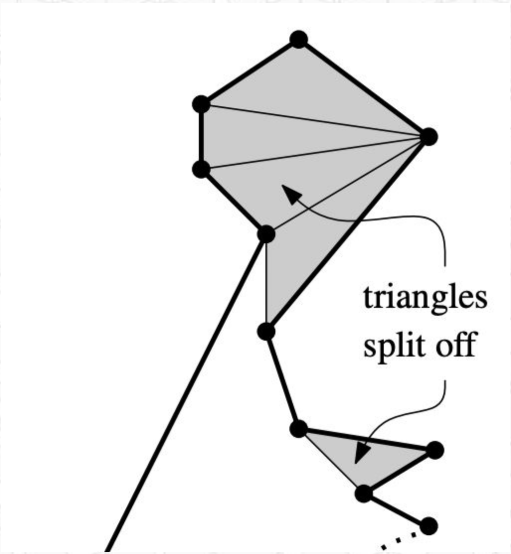
Analysis?

- Line sweep algorithm: cut into monotone polygons
  - Sort all vertices vertically - *O(n log n)*
  - Maintain horizontal sorting of active vertices - *O(log n)*
  - Locate “helper” vertex for each split/merge - *O(log n)*
  - → *O(n log n)*

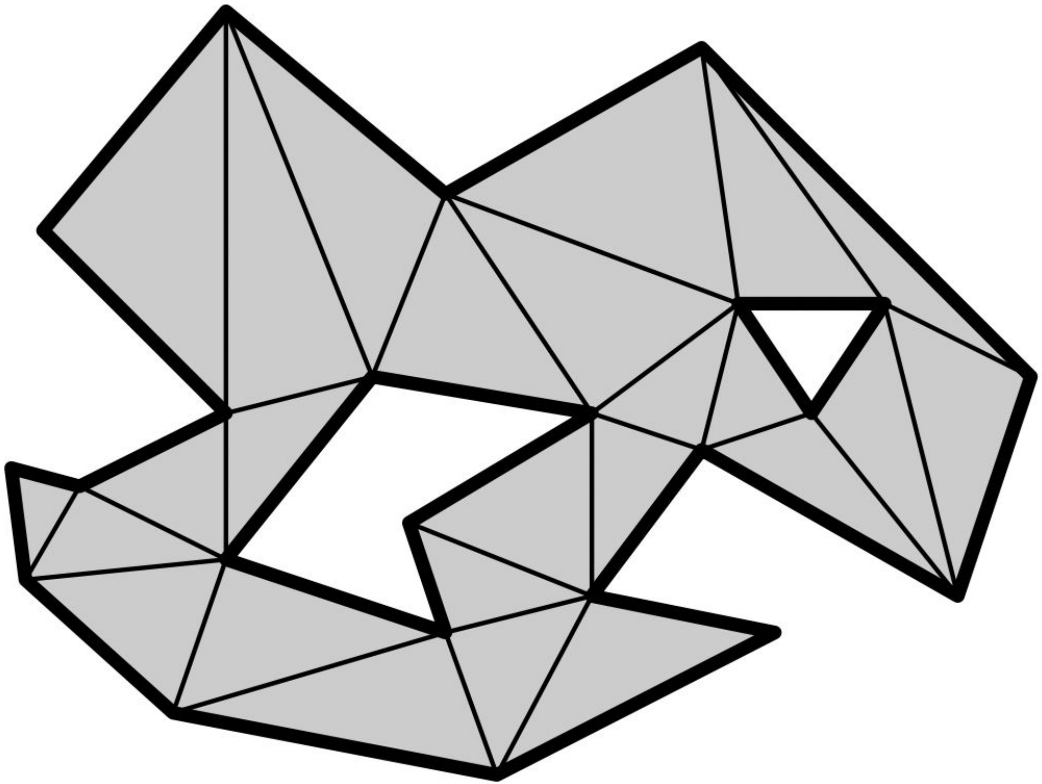
- Use stack to triangulate monotone polygon
  - Don’t need to sort (just walk boundary)
  - Each vertex is added once - *O(1)*
  - Each vertex (beyond first two) adds one ck triangle when it is removed from stack - *O(1)*
  - → *O(n)*

- Overall → *O(n log n)*
  Better than $O(n^2)$ *algorithm from previous lecture!*



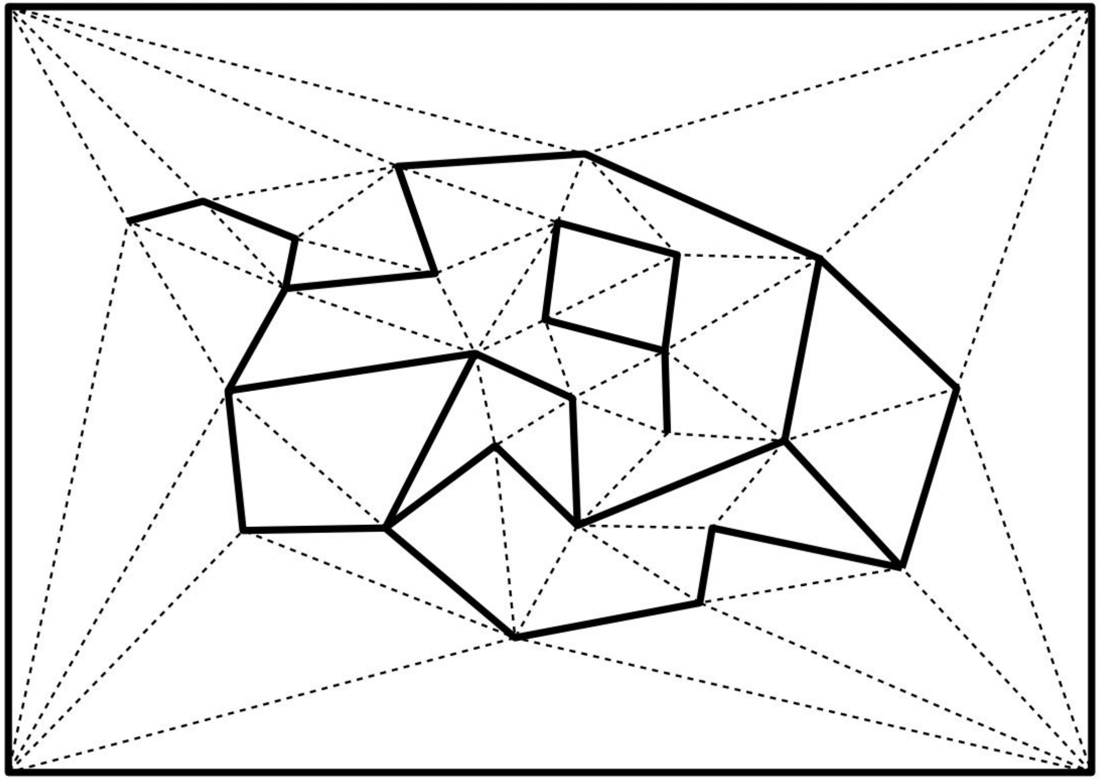


Also Works for Non-Simple Polygons (w/ interior holes)



Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

And it also works for Arbitrary Planar Subdivisions



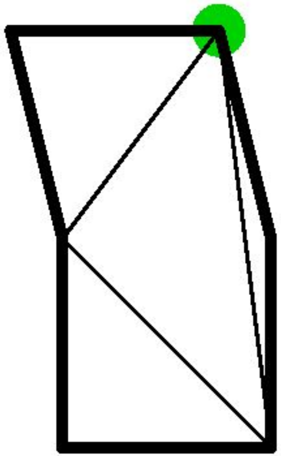
Computational Geometry Algorithms and Applications, de Berg, Cheong, van Kreveld and Overmars, Chapter 3

Outline for Today

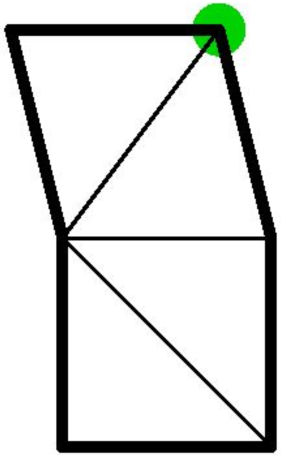
- Homework 2 Posted
- Last Time: Art Gallery Problem & Triangulation
- Improved Triangulation Algorithm
- Definition: Monotone Polygon
- Splitting into Monotone Polygons
- Triangulating a Monotone Polygon
- Analysis of Improved Triangulation Algorithm
- Future Lecture: Additional Triangulation Goals

Element Quality and Deformation Simulation

- The triangulation of a polygon is not unique!



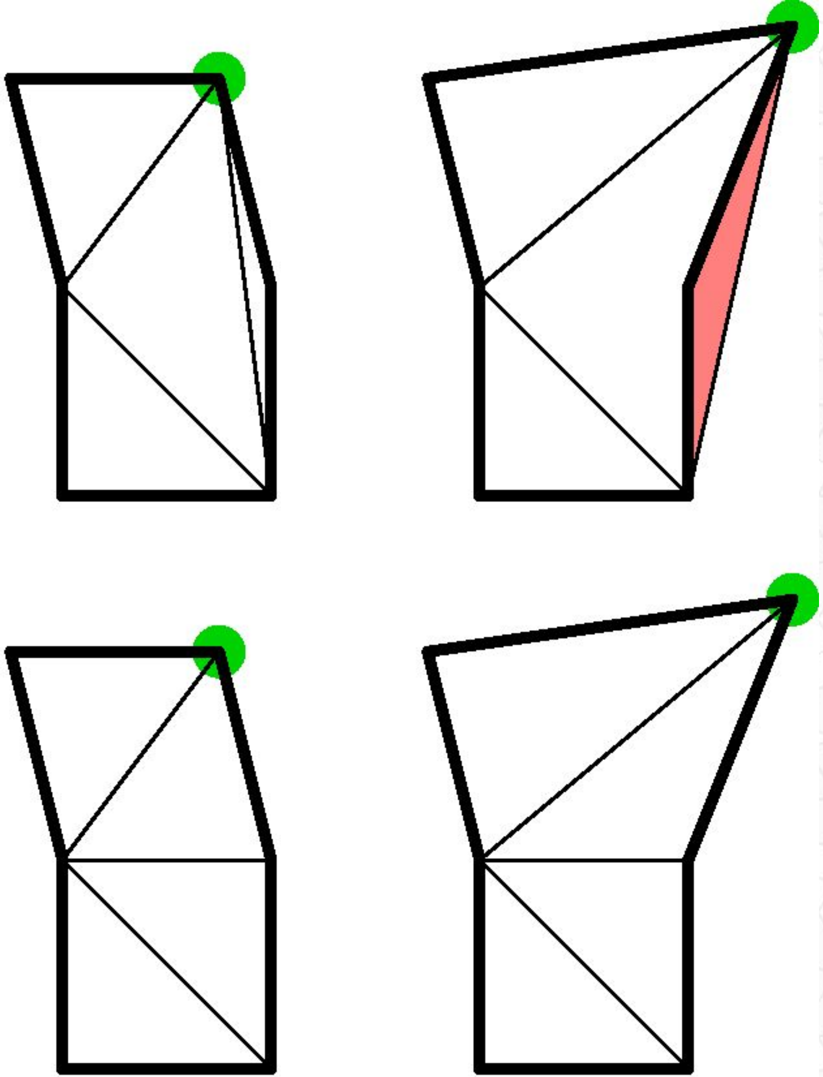
- Do we care which triangulation is produced?
- Are some triangulations Better for some applications?



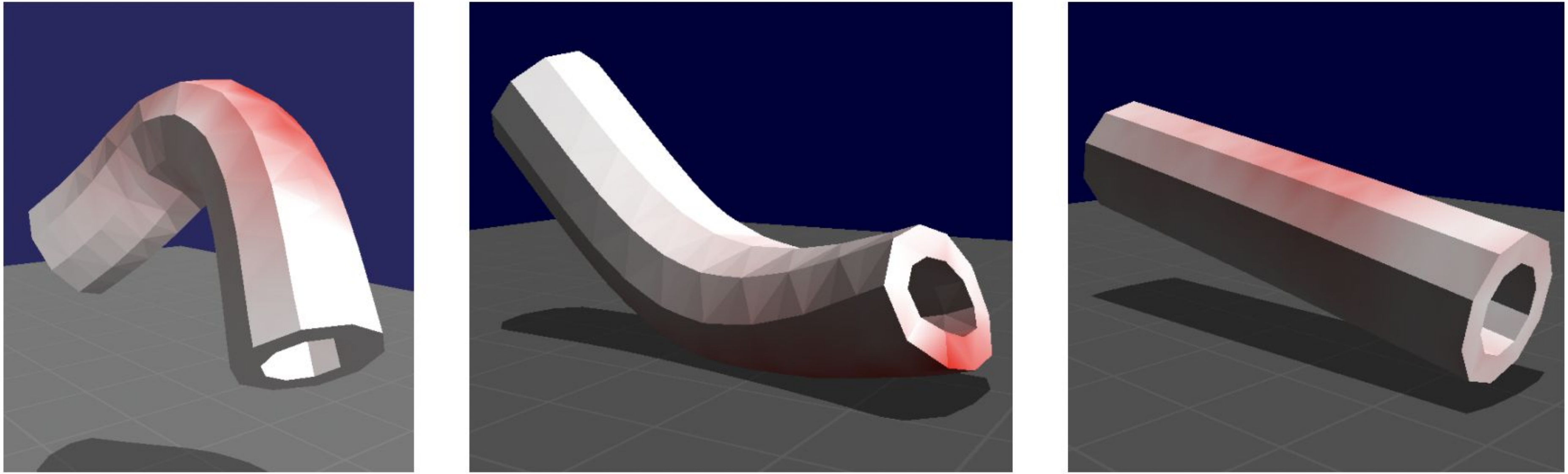
Element Quality and Deformation Simulation

- The triangulation of a polygon is not unique!

- Do we care which triangulation is produced?
- Are some triangulations Better for some applications?



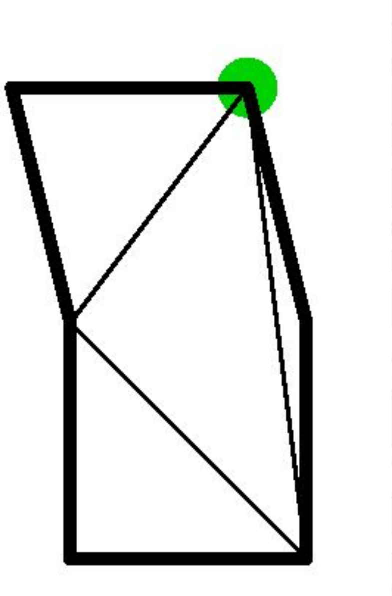
Element Quality and Deformation Simulation

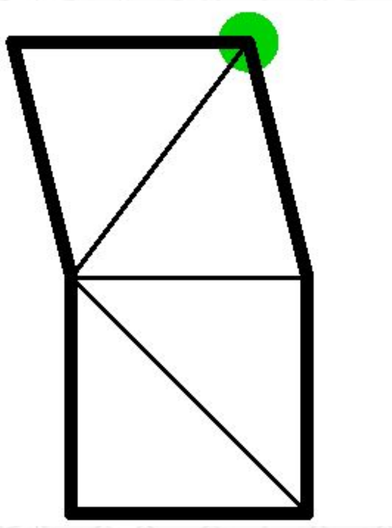


Mueller, Dorsey, McMillan, Jagnow, & Cutler
Stable Real-Time Deformations
Symposium on Computer Animation 2002

Degenerate/Ill-conditioned 2D Elements

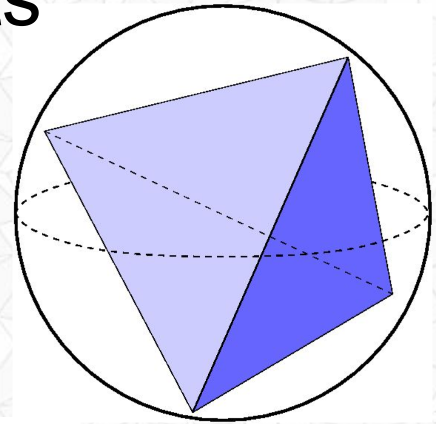
- a.k.a. how “equilateral” are the *triangles*?
  - Maximize the minimum angle
  - Minimize the maximum angle
  - Maximize the shortest edge
  - Ratio of longest edge to shortest edge
  - Ratio of area to area of circumscribed circle

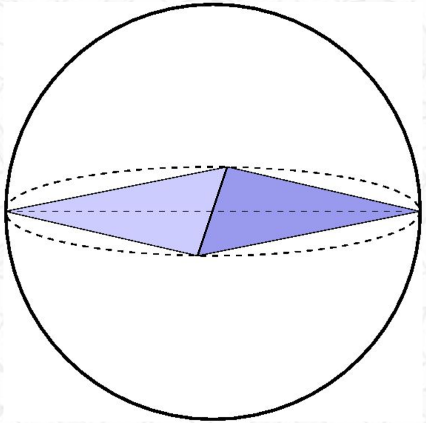


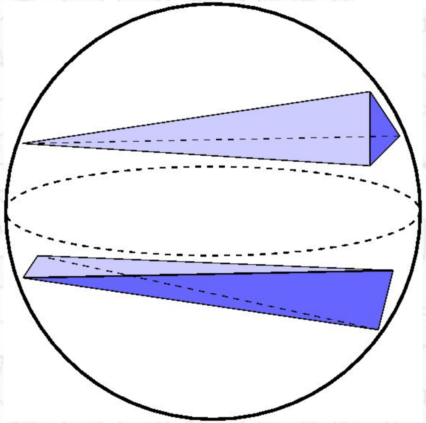


Degenerate/Ill-conditioned 3D Elements

- a.k.a. how “equilateral” are the *tetrahedra*?
  - Ratio of volume^2 to surface area^3
  - Smallest *solid* angle
  - Ratio of volume to volume of smallest circumscribed sphere

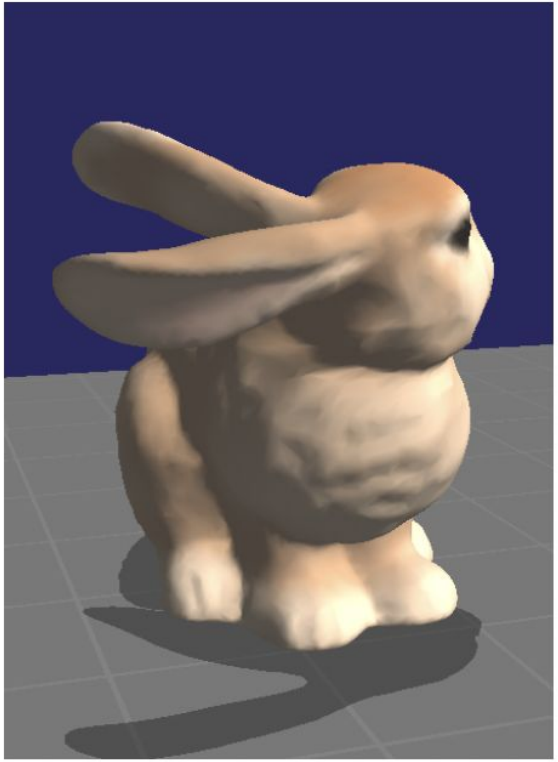






Element Quality and Deformation Simulation





Multiple Materials

Mueller, Dorsey, McMillan, Jagnow, & Cutler
Stable Real-Time Deformations
Symposium on Computer Animation 2002