Polygon Partitioning

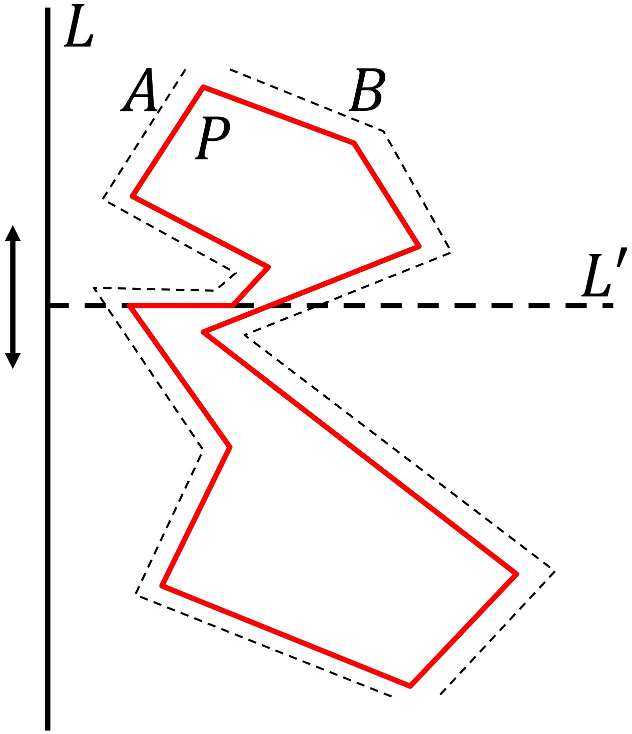
O'Rourke, Chapter 2

Outline

- Triangle Partitions
- Convex Partitions

Monotonicity

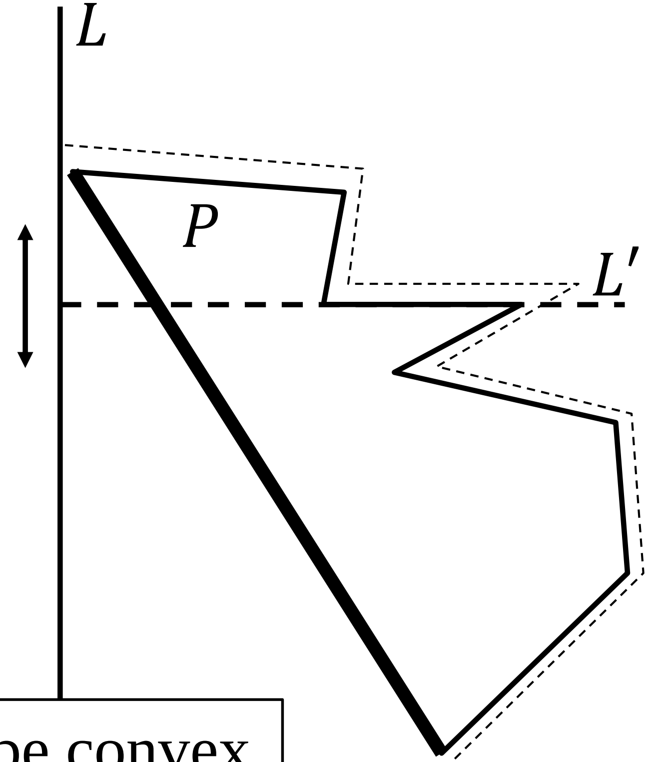
A polygonal P is *monotone w.r.t. a line* L if its boundary can be split into two polygon chains, A and B, such that each chain is monotonic w.r.t. L.



Monotonicity

A polygonal P is *monotone w.r.t. a line* L if its boundary can be split into two polygon chains, A and B, such that each chain is monotonic w.r.t. L.

A polygon P is a *monotone mountain* w.r.t. L if it is monotone w.r.t. L and one of the two chains (the *base*) is a single segment.



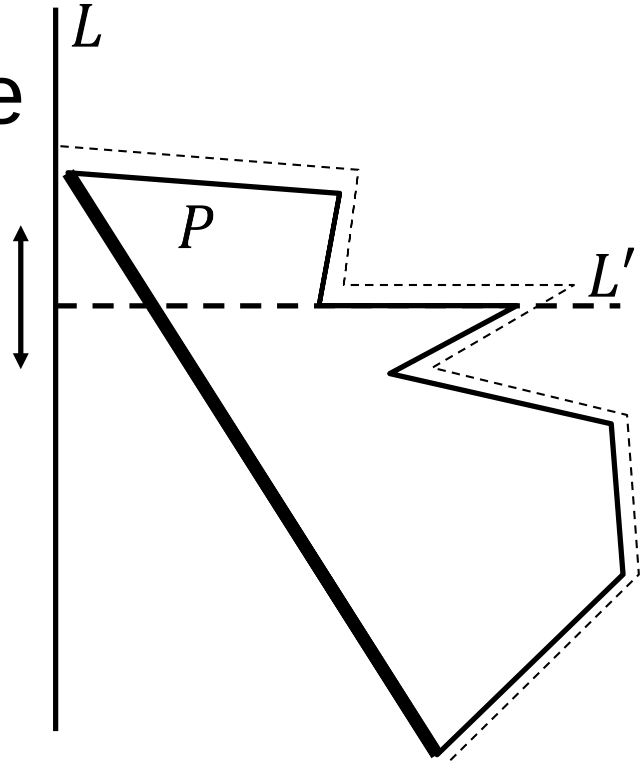
Note: Both endpoints of the base have to be convex

Claim

Every strictly convex vertex of a monotone mountain that is not on the base is an ear tip.

WLOG, we will assume that L is vertical and that the base is to the left.

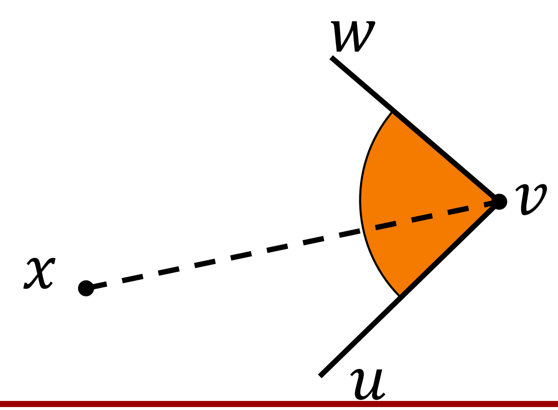
We will first show that the diagonal has to be interior and then that it cannot intersect the polygon.



Proofs

Recall:

Given successive vertices u, v, w on a polygon, and given some other vertex x on the polygon, the segment $\overline{vx}$ is locally interior at v if it is strictly interior to region swept out clockwise from $\overline{vu}$ to $\overline{vw}$.

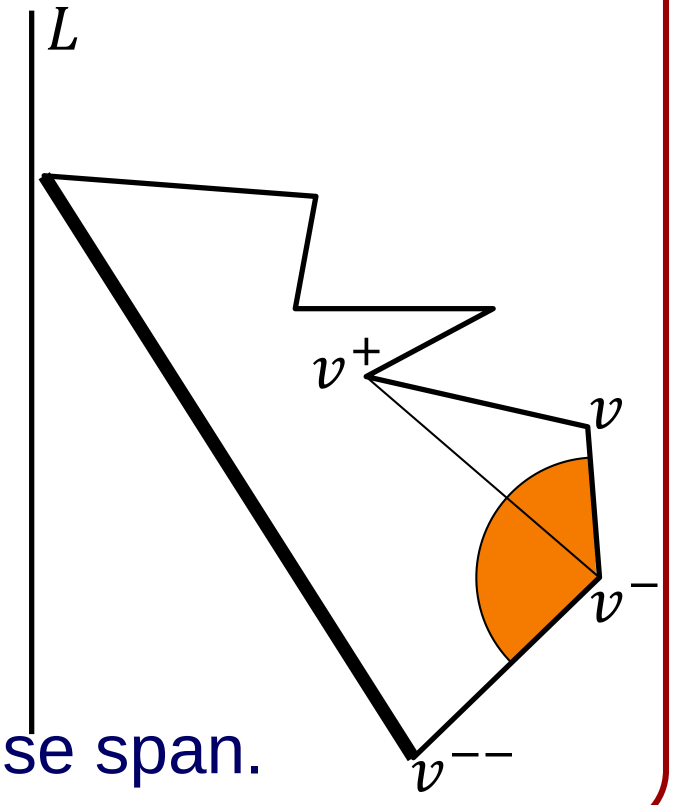


Proofs

Assume $\exists\{v^-, v, v^+\} \subset P$, with v strictly convex, s.t. $\overline{v^- v^+}$ is not an interior diagonal.

Locally Interior at v^-:
$\overline{v^+ v^-}$ cannot be locally exterior at v^- because:

- If v^- is not on the base, then $\overline{v^{--} v^-}$ must be below v^+. Since v is strictly convex, v is to the right of $\overline{v^- v^+}$. But v^+ is also above $\overline{v^- v^{--}}$ so $\overline{v^- v^+}$ must be in the clockwise span.



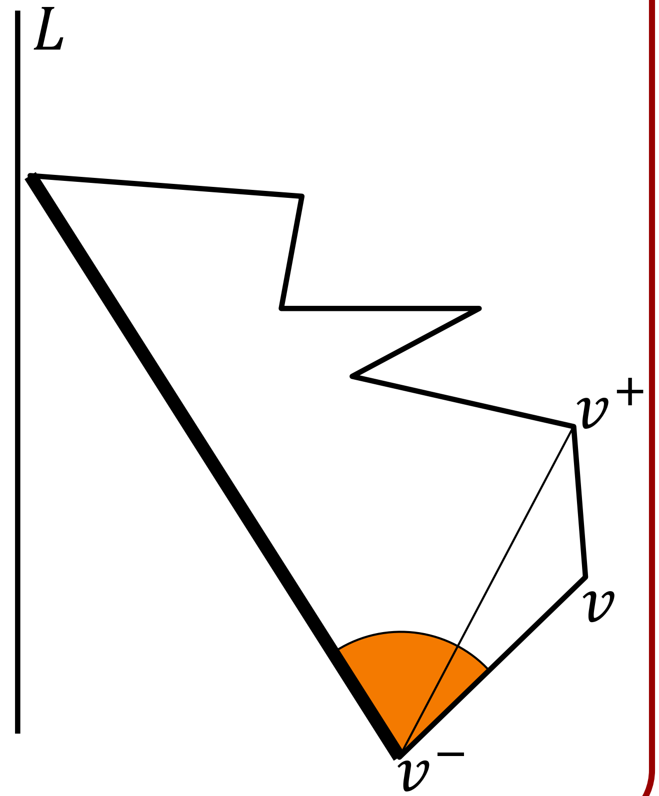
Proofs

Assume $\exists\{v^-, v, v^+\} \subset P$, with v strictly convex, s.t. $\overline{v^- v^+}$ is not an interior diagonal.

<u>Locally Interior at v^-</u>:

$\overline{v^+ v^-}$ cannot be locally exterior at v^- because:

- If v^- is on the base, then (again) since v is strictly convex v is to the right of $\overline{v^- v^+}$. But v^+ is also be to the right of the base so $\overline{v^- v^+}$ must be in the clockwise span.

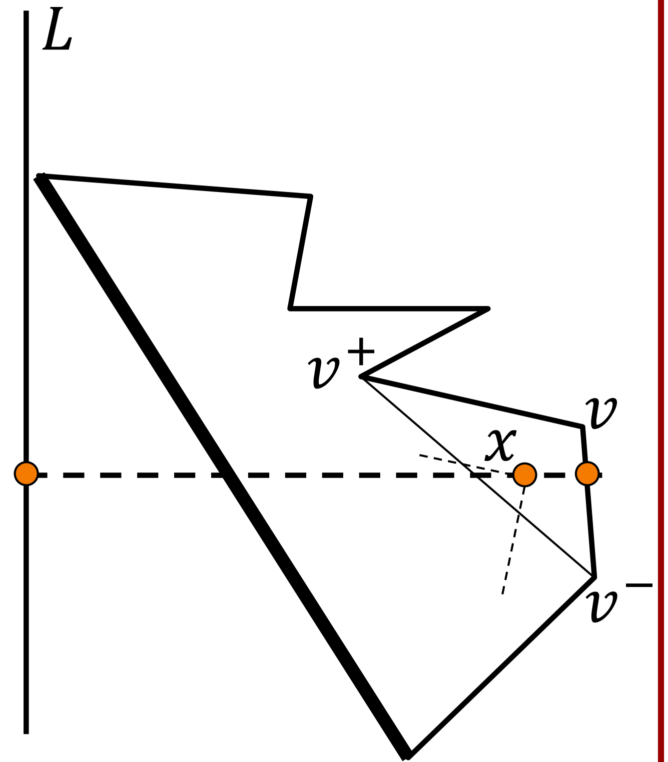


Proofs

Assume $\exists\{v^-, v, v^+\} \subset P$, with v strictly convex, s.t. $\overline{v^- v^+}$ is not an interior diagonal.

$\Rightarrow$ If $\overline{v^- v^+}$ is not an interior diagonal then $\exists x \in P$, reflex, with x interior to $\Delta v^- v v^+$.

- Interior $\Rightarrow$ it cannot lie on the chain $\overline{v^- v v^+}$.
- Reflex $\Rightarrow$ it is not on the base.

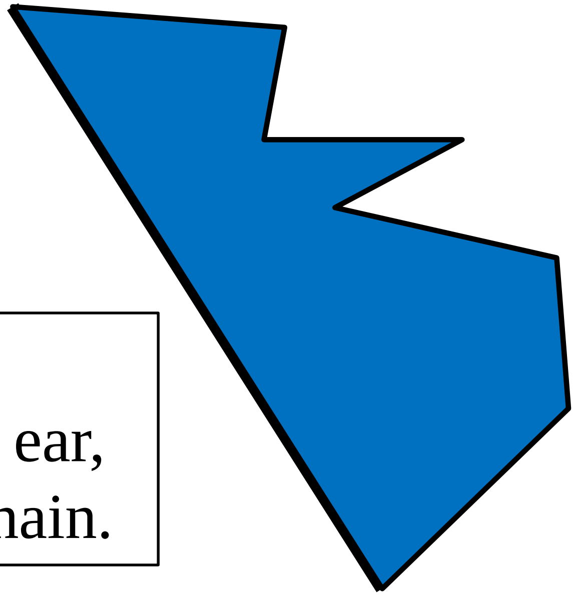


$\Rightarrow$ The horizontal through x intersects P in three points.

Monotone Mountain Triangulation

Approach:

1. Find a convex vertex not on the base.
2. Remove it.
3. Go to step 1.

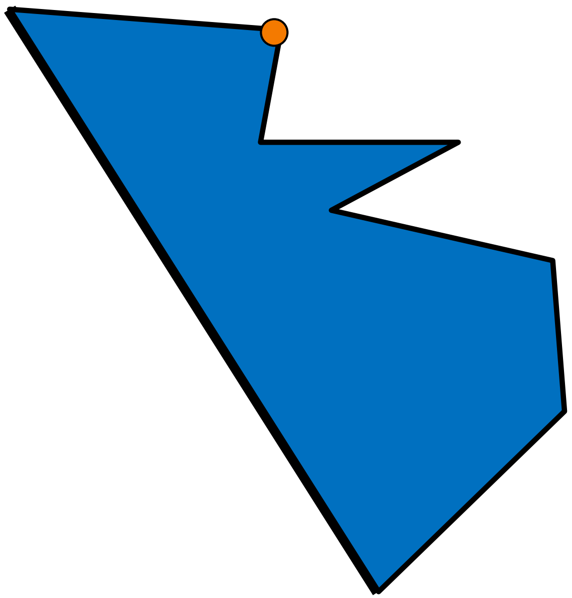


Note:
We can induct because when we remove an ear, we do not violate the monotonicity of the chain.

Monotone Mountain Triangulation

Approach:

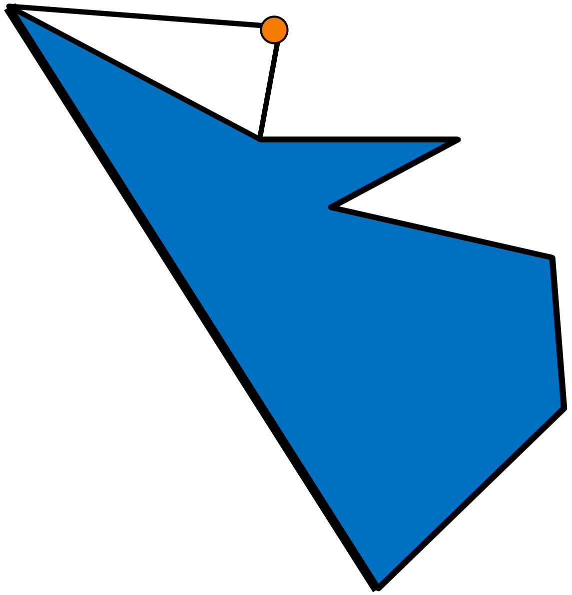
1. **Find a convex vertex not on the base.**
2. Remove it.
3. Go to step 1.



Monotone Mountain Triangulation

Approach:

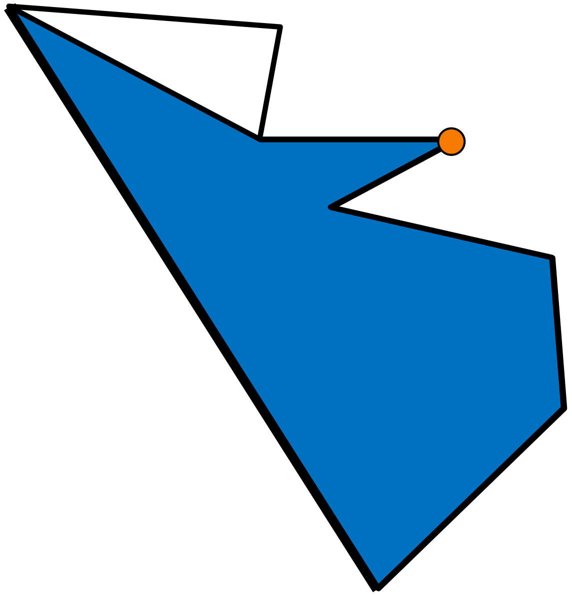
1. Find a convex vertex not on the base.
2. **Remove it.**
3. Go to step 1.



Monotone Mountain Triangulation

Approach:

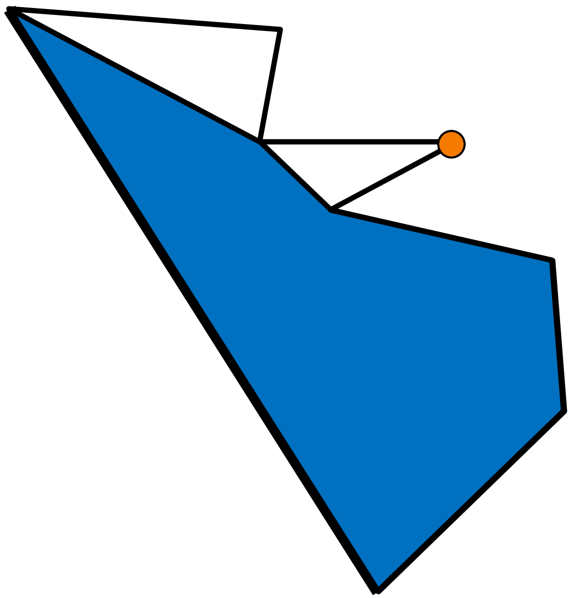
1. **Find a convex vertex not on the base.**
2. Remove it.
3. Go to step 1.



Monotone Mountain Triangulation

Approach:

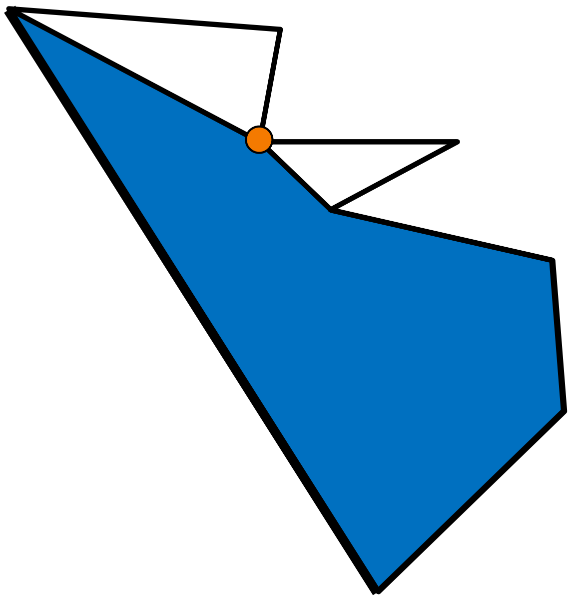
1. Find a convex vertex not on the base.
2. **Remove it.**
3. Go to step 1.



Monotone Mountain Triangulation

Approach:

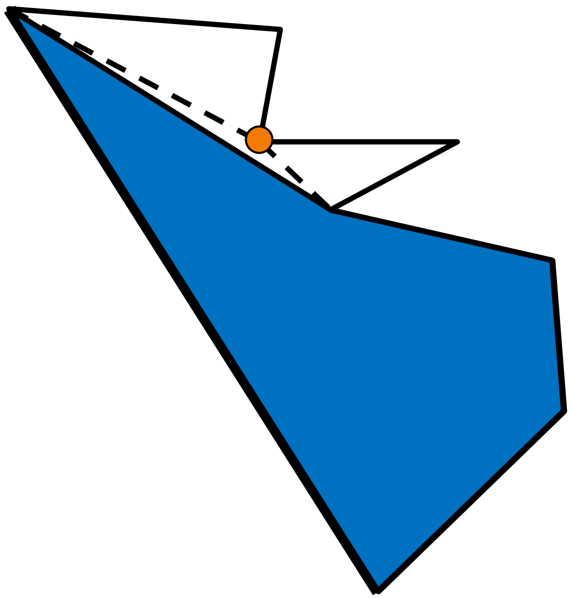
1. **Find a convex vertex not on the base.**
2. Remove it.
3. Go to step 1.



Monotone Mountain Triangulation

Approach:

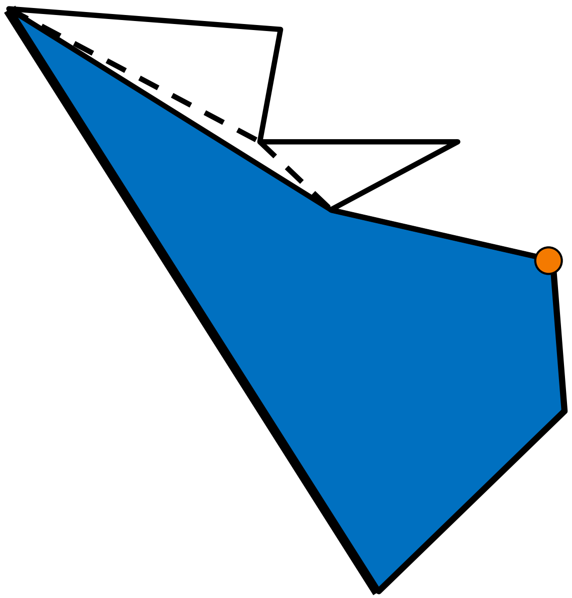
1. Find a convex vertex not on the base.
2. **Remove it.**
3. Go to step 1.



Monotone Mountain Triangulation

Approach:

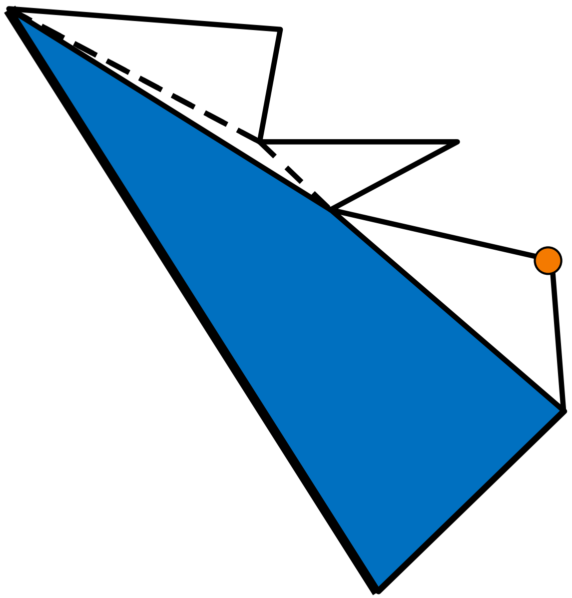
1. **Find a convex vertex not on the base.**
2. Remove it.
3. Go to step 1.



Monotone Mountain Triangulation

Approach:

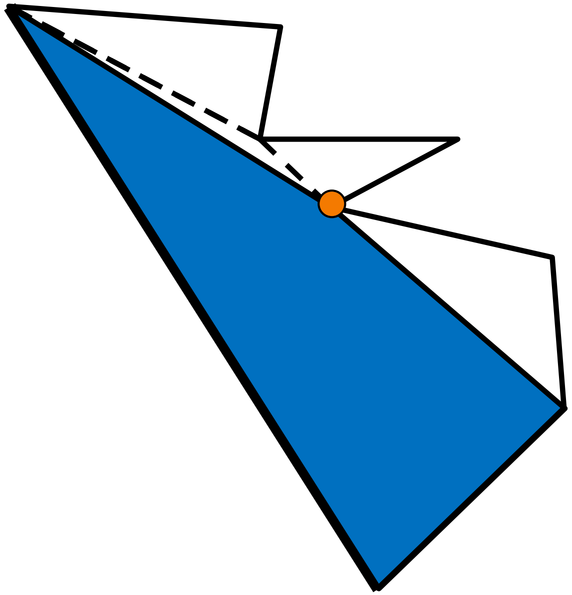
1. Find a convex vertex not on the base.
2. **Remove it.**
3. Go to step 1.



Monotone Mountain Triangulation

Approach:

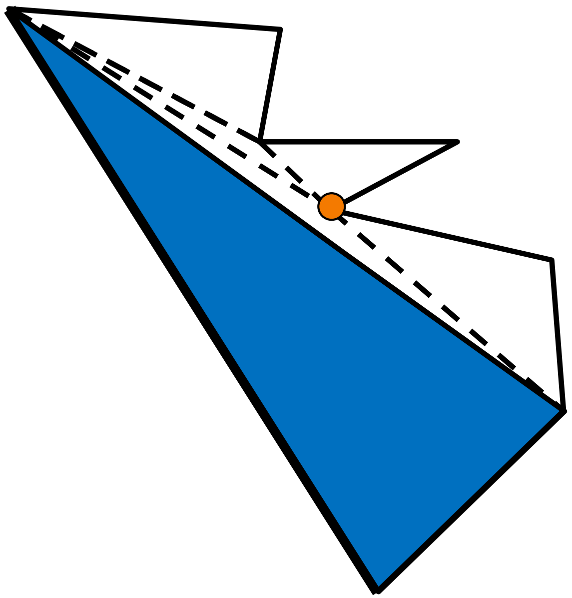
1. **Find a convex vertex not on the base.**
2. Remove it.
3. Go to step 1.



Monotone Mountain Triangulation

Approach:

1. Find a convex vertex not on the base.
2. **Remove it.**
3. Go to step 1.

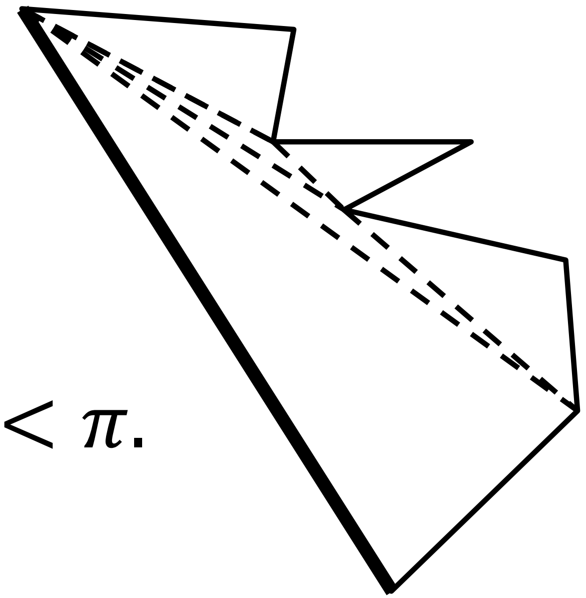


Monotone Mountain Triangulation

Approach:

1. Find a convex vertex not on the base.
2. Remove it.
3. Go to step 1.

To do this, we need to be able to quickly identify the next convex vertex.



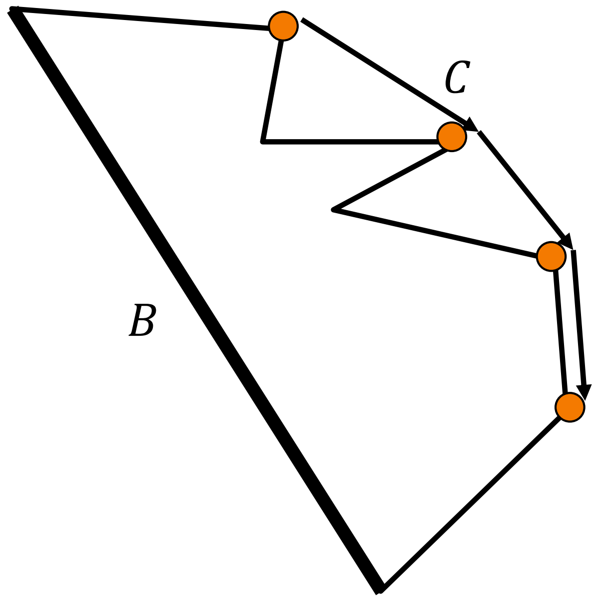
Recall:
Strictly convex $\Leftrightarrow$ interior angle $< \pi$.

Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$ FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)



Monotone Mountain Triangulation

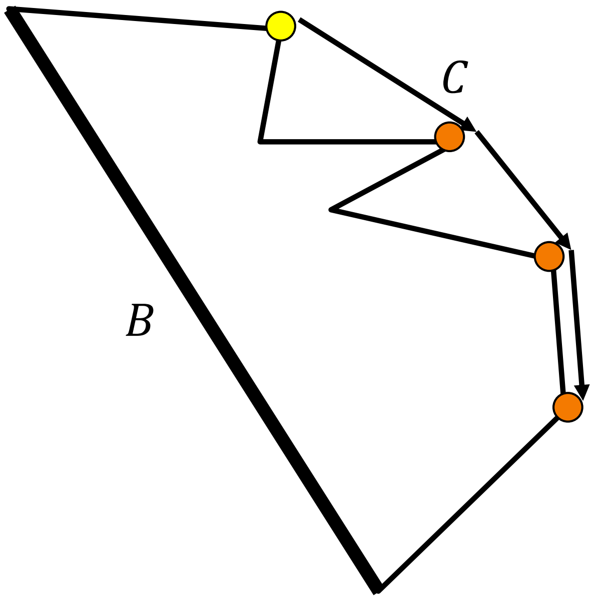
MonotoneMountainTriangulation(P):

B ←FindBase(P)

C ← LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

v ← First(C)



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

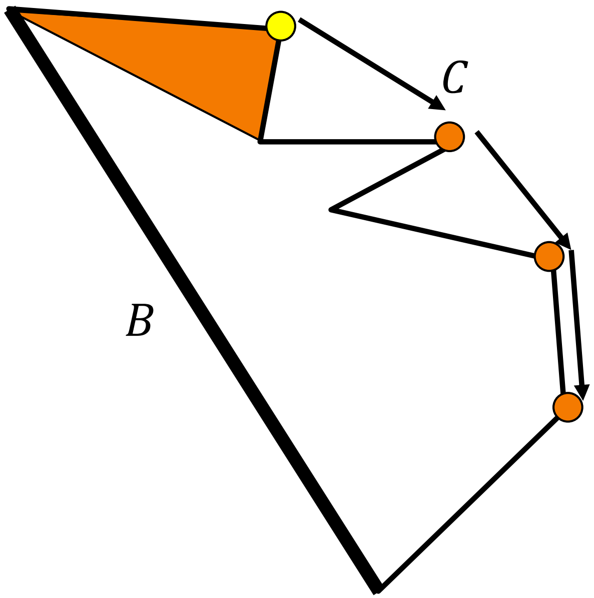
$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

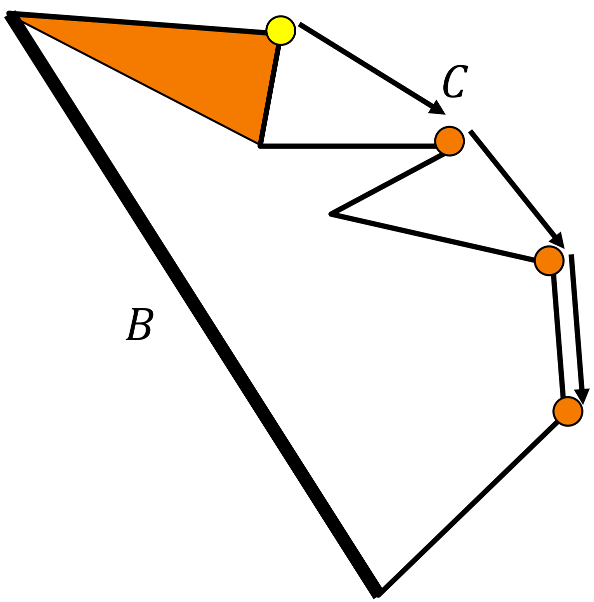
if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

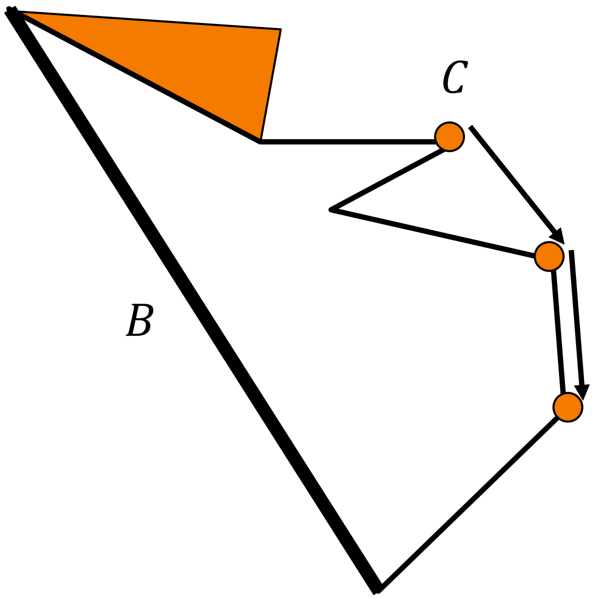
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$\underline{C \leftarrow C - \{v\}}$

$\underline{P \leftarrow P - \{v\}}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$\underline{v \leftarrow \text{First}(\ C\ )}$

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

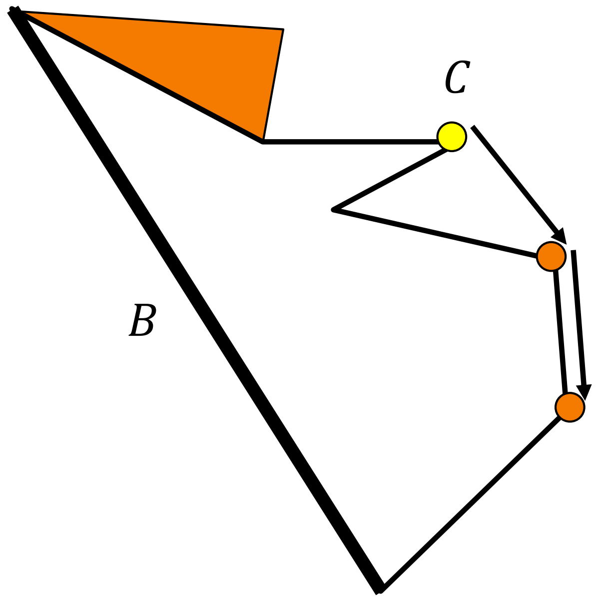
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

<u>output($\Delta v^- v v^+$)</u>

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

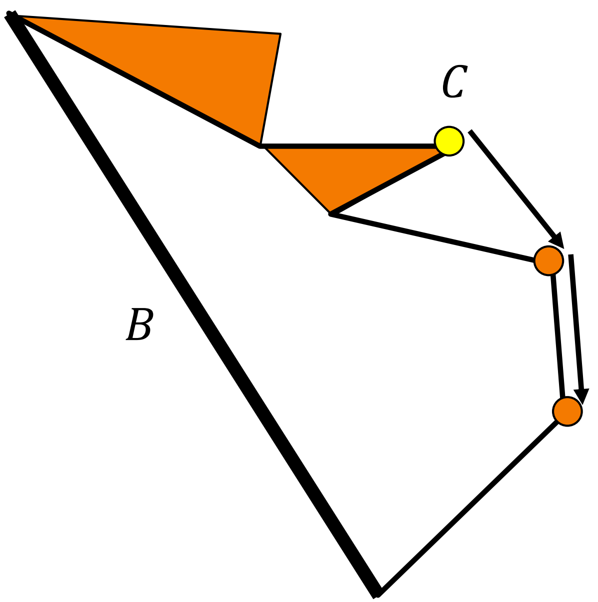
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$ FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

<u>if($v^+ \notin C$ AND $v^+ \notin B$)</u>

<u>if($\angle v^- v^+ v^{++} < \pi$)</u>

<u>v.addAfter(v^+)</u>

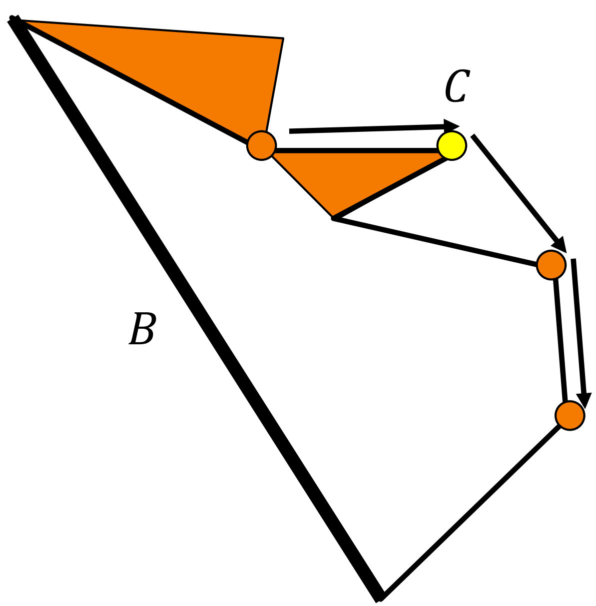
<u>if($v^- \notin C$ AND $v^- \notin B$)</u>

<u>if($\angle v^{--} v^- v^+ < \pi$)</u>

<u>v.addBefore(v^-)</u>

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$ FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

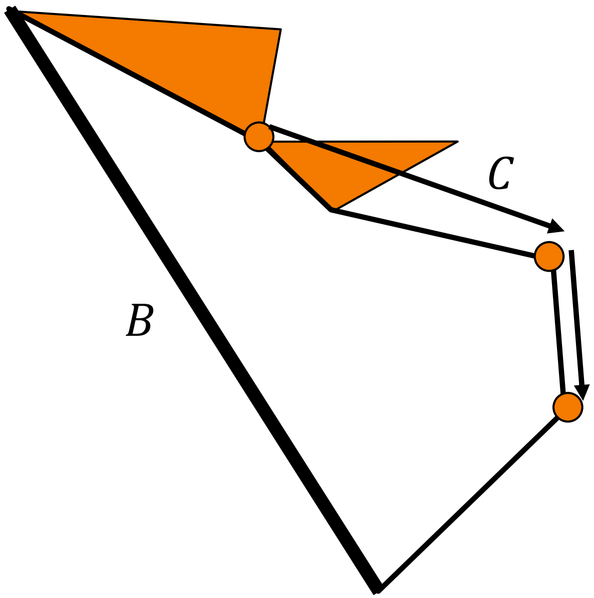
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$\underline{C \leftarrow C - \{v\}}$

$\underline{P \leftarrow P - \{v\}}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

<u>$v \leftarrow$ First(C)</u>

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

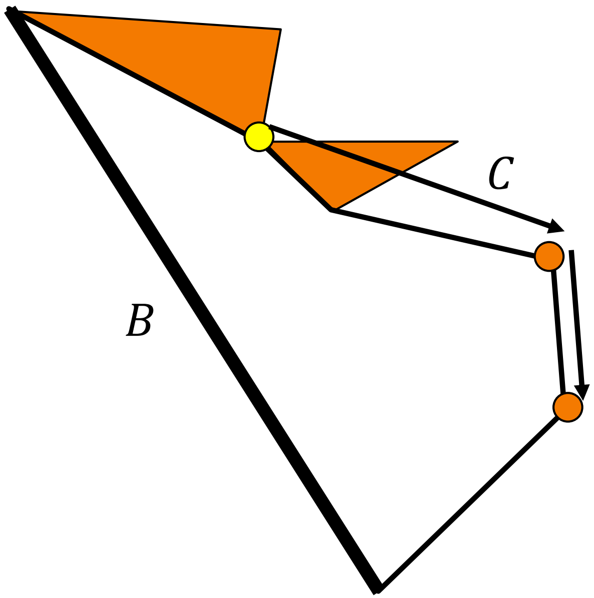
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

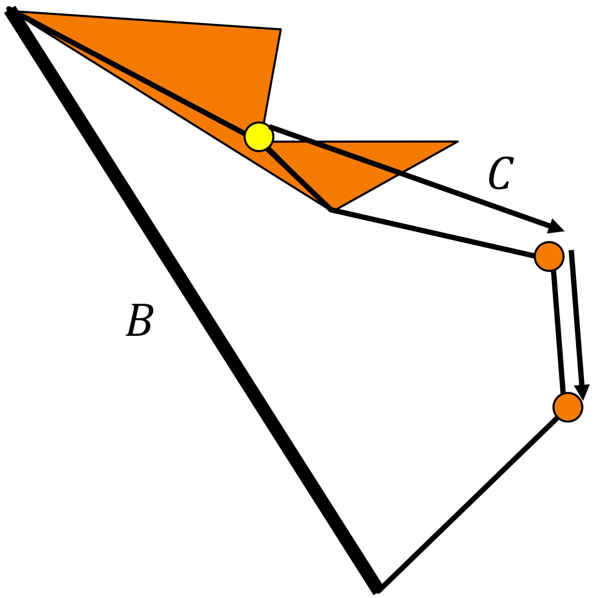
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

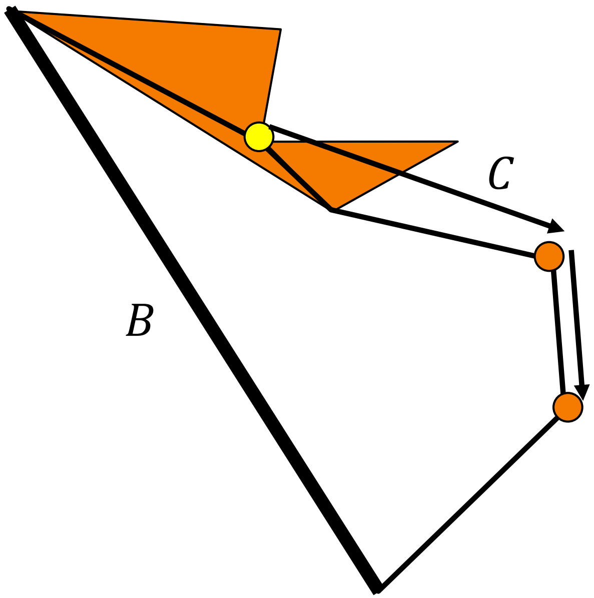
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$ FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

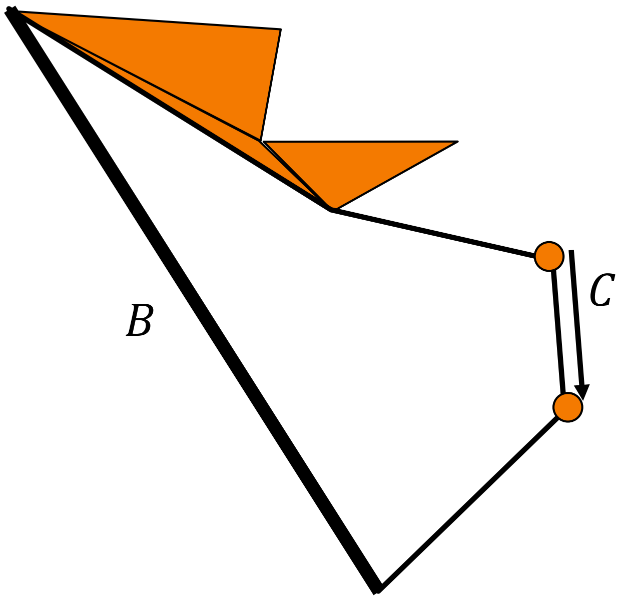
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$\underline{C \leftarrow C - \{v\}}$

$\underline{P \leftarrow P - \{v\}}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

<u>$v \leftarrow$ First(C)</u>

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

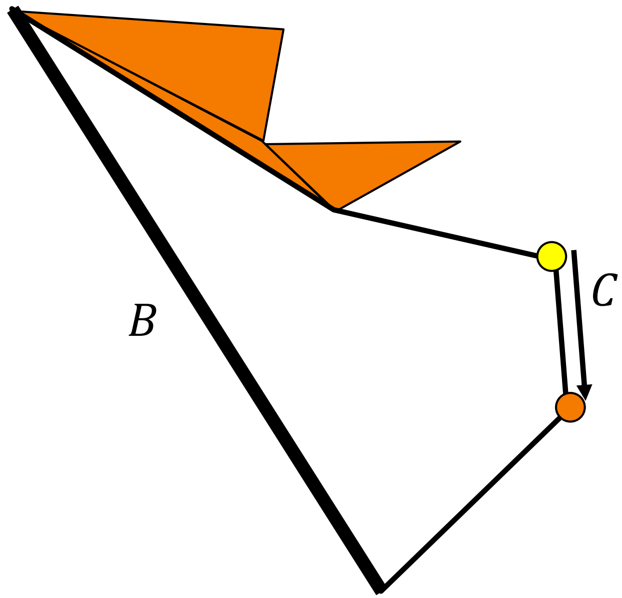
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^{-}vv^{+}$)

if($v^{+} \notin C$ AND $v^{+} \notin B$)

if($\angle v^{-}v^{+}v^{++} < \pi$)

v.addAfter(v^{+})

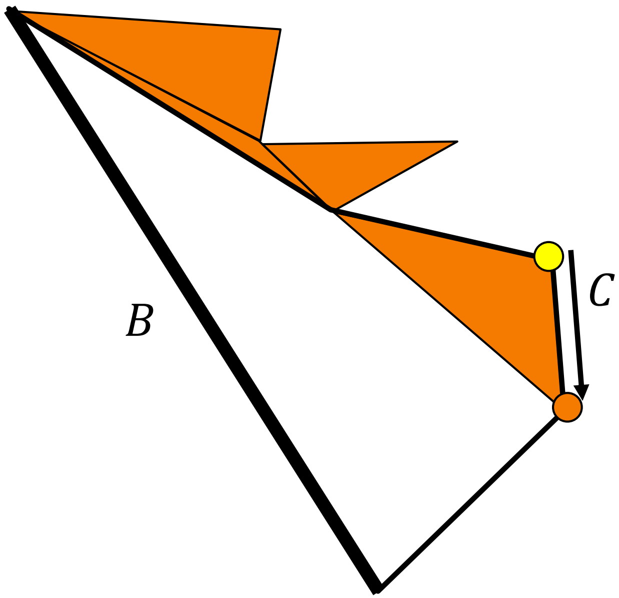
if($v^{-} \notin C$ AND $v^{-} \notin B$)

if($\angle v^{--}v^{-}v^{+} < \pi$)

v.addBefore(v^{-})

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

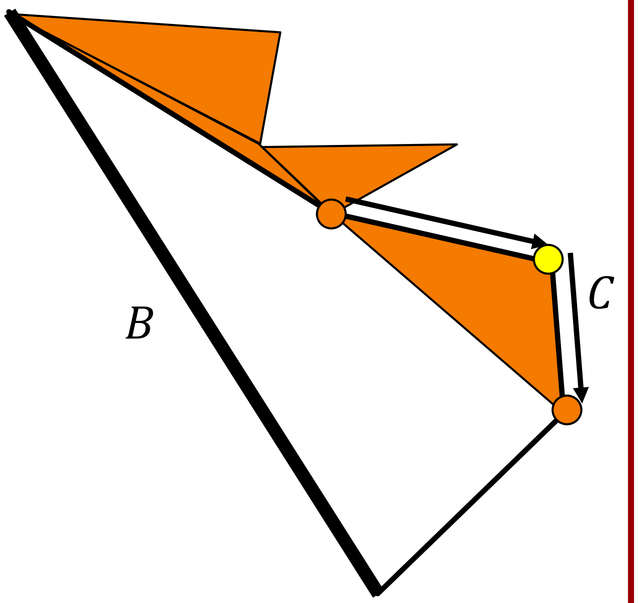
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

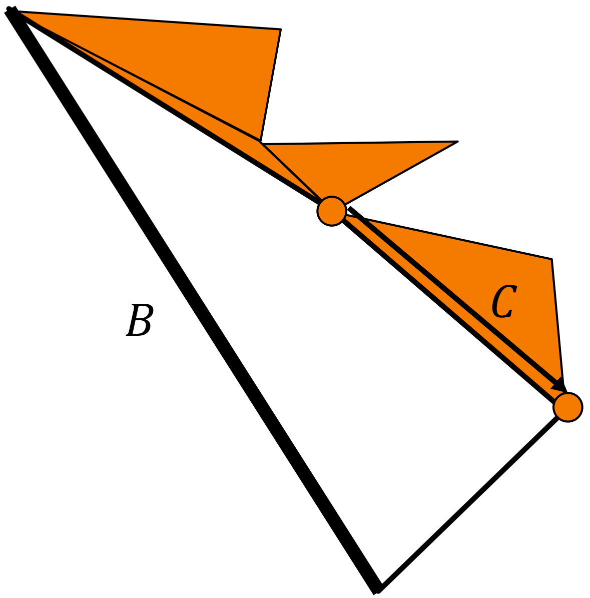
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$\underline{C \leftarrow C - \{v\}}$

$\underline{P \leftarrow P - \{v\}}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$\underline{v \leftarrow \text{First}(C)}$

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

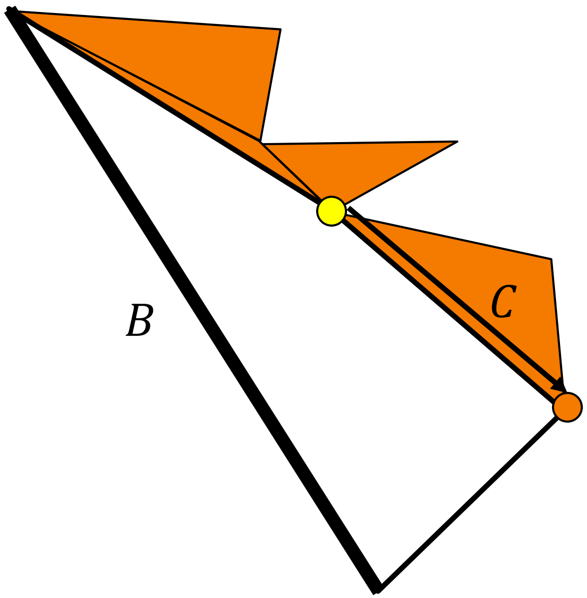
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$ FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

<u>output($\Delta v^- v v^+$)</u>

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

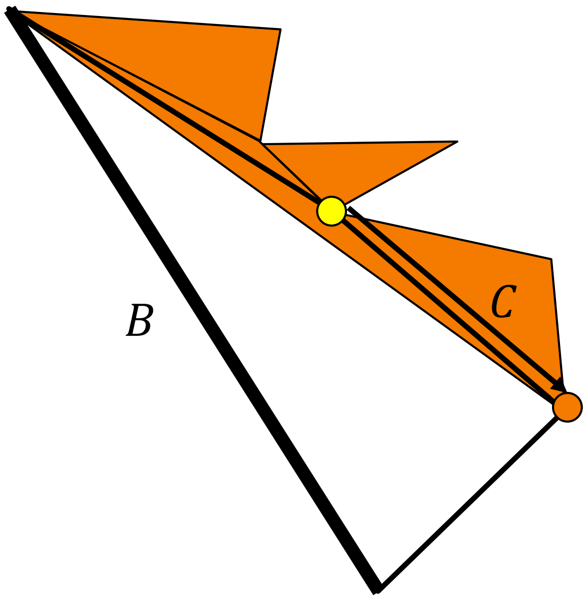
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$ FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

<u>if($v^+ \notin C$ AND $v^+ \notin B$)</u>

<u>if($\angle v^- v^+ v^{++} < \pi$)</u>

<u>v.addAfter(v^+)</u>

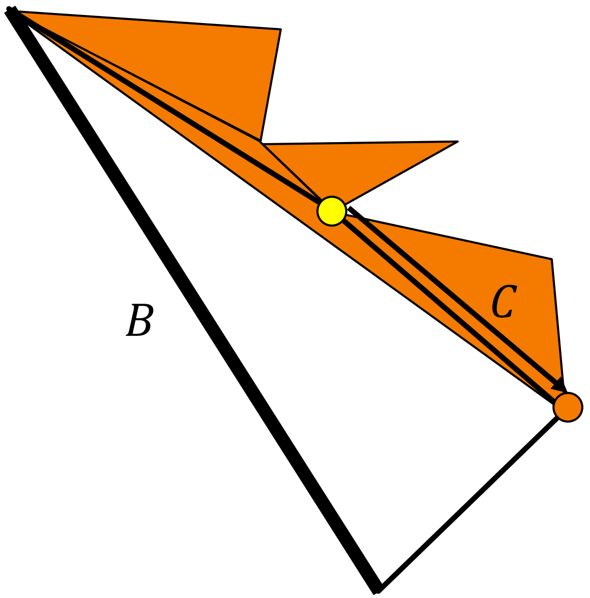
<u>if($v^- \notin C$ AND $v^- \notin B$)</u>

<u>if($\angle v^{--} v^- v^+ < \pi$)</u>

<u>v.addBefore(v^-)</u>

$C \leftarrow C - \{v\}$

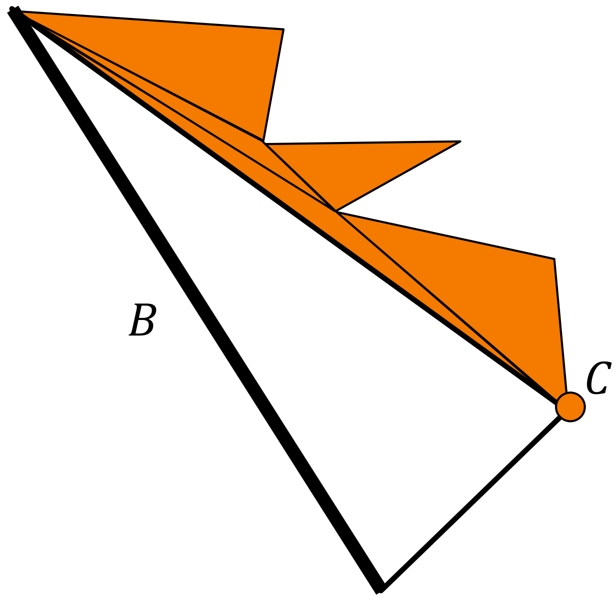
$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

- $B \leftarrow$ FindBase(P)
- $C \leftarrow$ LinkConvexVertices($P - B$)
- while $C \neq \emptyset$:
  - $v \leftarrow$ First(C)
  - output($\Delta v^- v v^+$)
  - if($v^+ \notin C$ AND $v^+ \notin B$)
    - if($\angle v^- v^+ v^{++} < \pi$)
      - v.addAfter(v^+)
  - if($v^- \notin C$ AND $v^- \notin B$)
    - if($\angle v^{--} v^- v^+ < \pi$)
      - v.addBefore(v^-)
  - $\underline{C \leftarrow C - \{v\}}$
  - $\underline{P \leftarrow P - \{v\}}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

<u>$v \leftarrow$ First(C)</u>

output($\Delta v^- v v^+$)

if($v^+ \notin C$ AND $v^+ \notin B$)

if($\angle v^- v^+ v^{++} < \pi$)

v.addAfter(v^+)

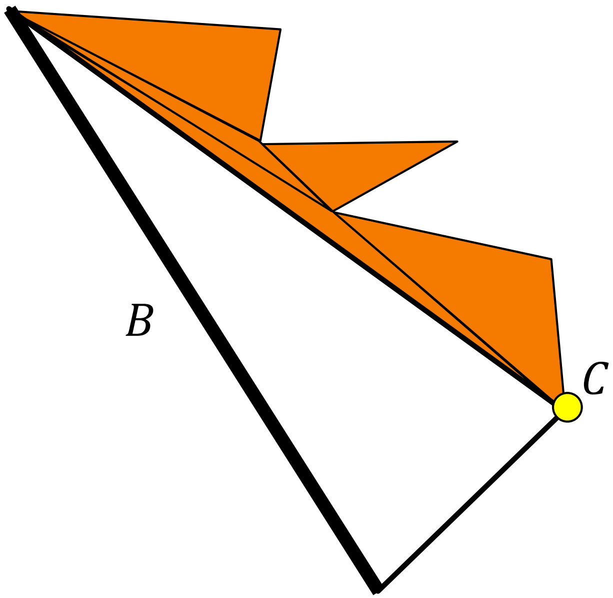
if($v^- \notin C$ AND $v^- \notin B$)

if($\angle v^{--} v^- v^+ < \pi$)

v.addBefore(v^-)

$C \leftarrow C - \{v\}$

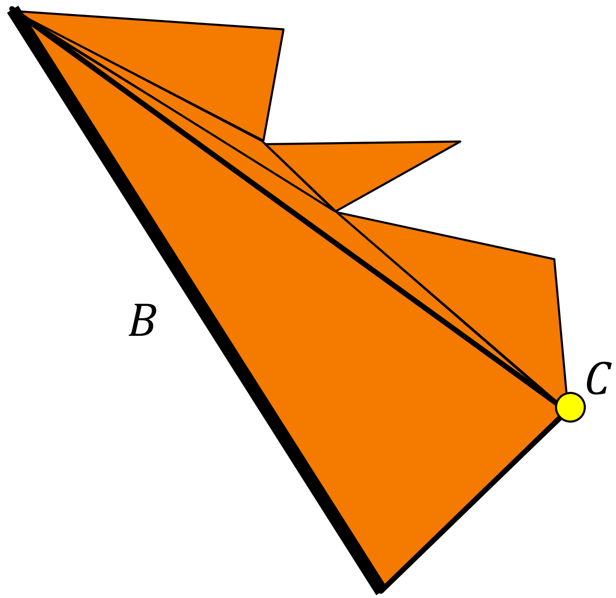
$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

- $B \leftarrow$ FindBase(P)
- $C \leftarrow$ LinkConvexVertices($P - B$)
- while $C \neq \emptyset$:
  - $v \leftarrow$ First(C)
  - <u>output($\Delta v^- v v^+$)</u>
  - if($v^+ \notin C$ AND $v^+ \notin B$)
    - if($\angle v^- v^+ v^{++} < \pi$)
      - v.addAfter(v^+)
  - if($v^- \notin C$ AND $v^- \notin B$)
    - if($\angle v^{--} v^- v^+ < \pi$)
      - v.addBefore(v^-)
  - $C \leftarrow C - \{v\}$
  - $P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

$B \leftarrow$FindBase(P)

$C \leftarrow$ LinkConvexVertices($P - B$)

while $C \neq \emptyset$:

$v \leftarrow$ First(C)

output($\Delta v^- v v^+$)

<u>if($v^+ \notin C$ AND $v^+ \notin B$)</u>

<u>if($\angle v^- v^+ v^{++} < \pi$)</u>

<u>v.addAfter(v^+)</u>

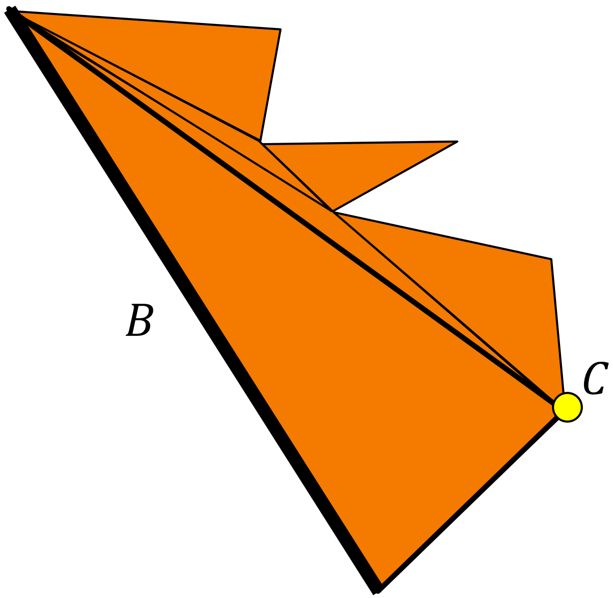
<u>if($v^- \notin C$ AND $v^- \notin B$)</u>

<u>if($\angle v^{--} v^- v^+ < \pi$)</u>

<u>v.addBefore(v^-)</u>

$C \leftarrow C - \{v\}$

$P \leftarrow P - \{v\}$



Monotone Mountain Triangulation

MonotoneMountainTriangulation(P):

  $B \leftarrow$FindBase(P)

  $C \leftarrow$ LinkConvexVertices($P - B$)

  while $C \neq \emptyset$:

    $v \leftarrow$ First(C)

    output($\Delta v^- v v^+$)

    if($v^+ \notin C$ AND $v^+ \notin B$)

      if($\angle v^- v^+ v^{++} < \pi$)

        v.addAfter(v^+)

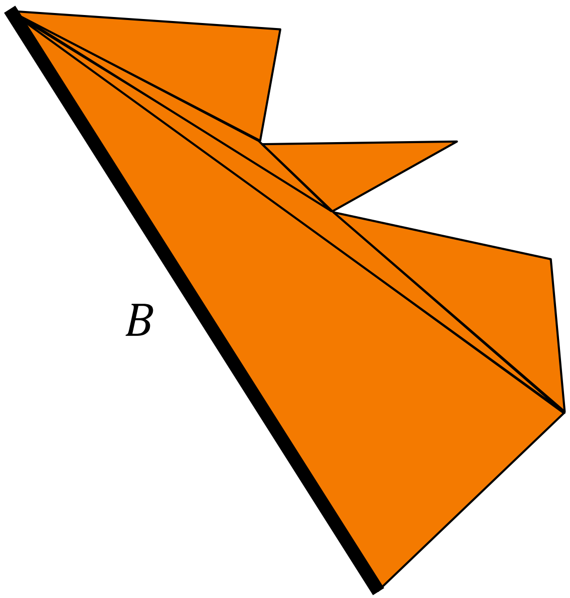
    if($v^- \notin C$ AND $v^- \notin B$)

      if($\angle v^{--} v^- v^+ < \pi$)

        v.addBefore(v^-)

    $\underline{C \leftarrow C - \{v\}}$

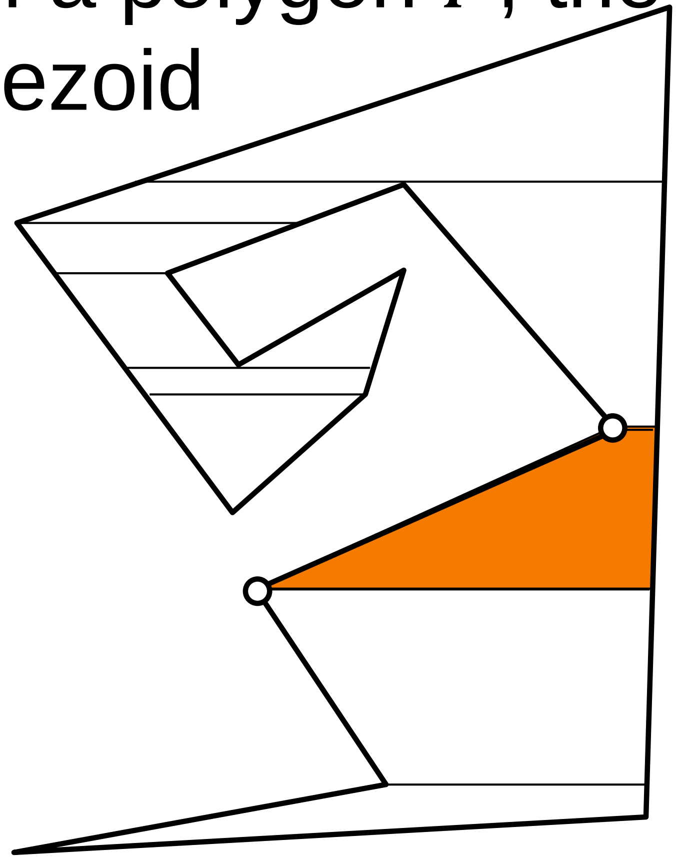
    $\underline{P \leftarrow P - \{v\}}$



Generating Monotone Mountains

Recall:

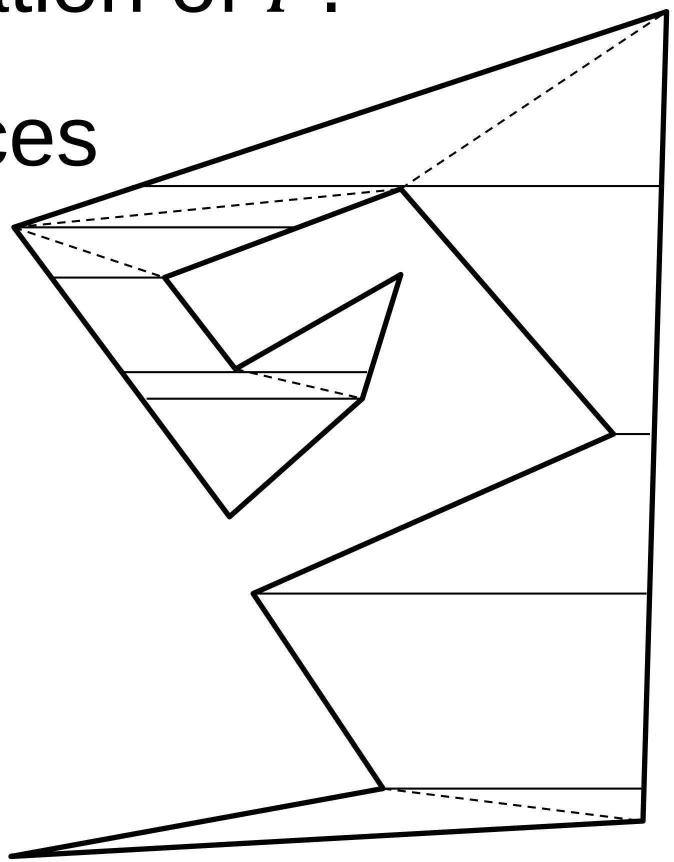
Given a trapezoidalization of a polygon P, the supporting vertices of a trapezoid are the vertices of P whose horizontals form the bases of the trapezoid.



Generating Monotone Mountains

Approach:

- Compute a trapezoidalization of P.
- Connect supporting vertices that don't come from the same side.*

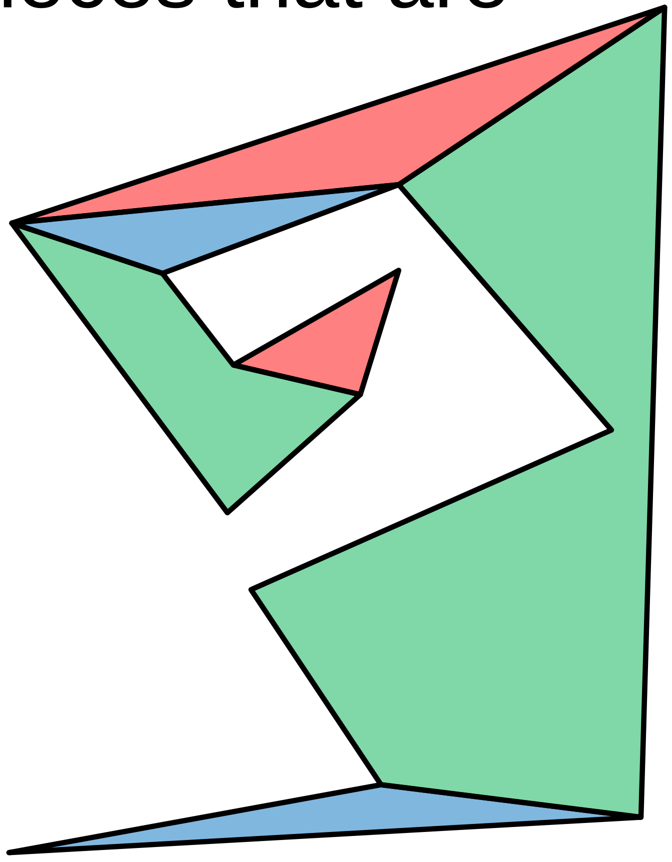


*If a supporting vertex is in the middle of a base it automatically does not come from the same side as the other supporting vertex.

Generating Monotone Mountains

Claim:

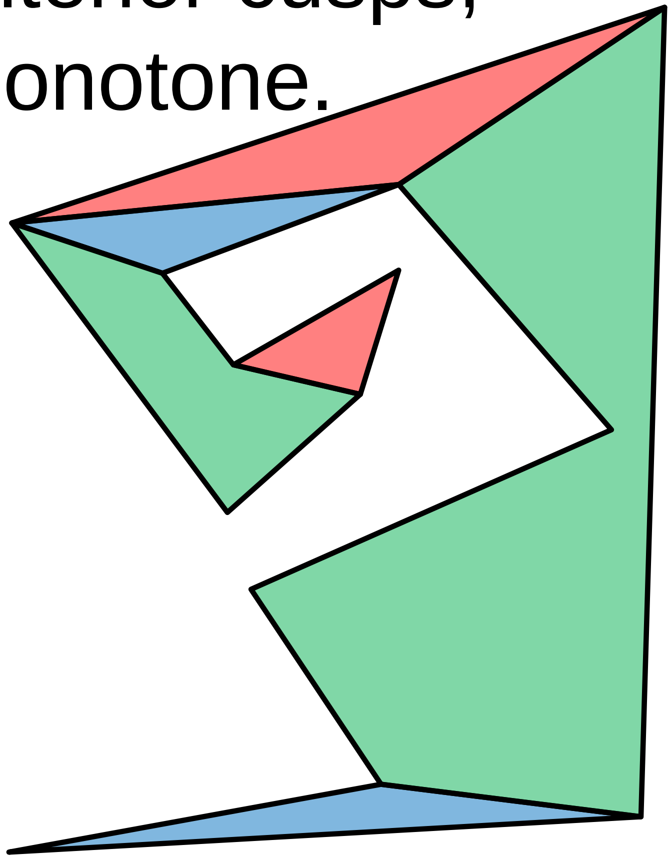
Such a partition generates pieces that are monotone mountains.



Generating Monotone Mountains

Proof:

This algorithm removes all interior cusps, so the partition pieces are monotone.



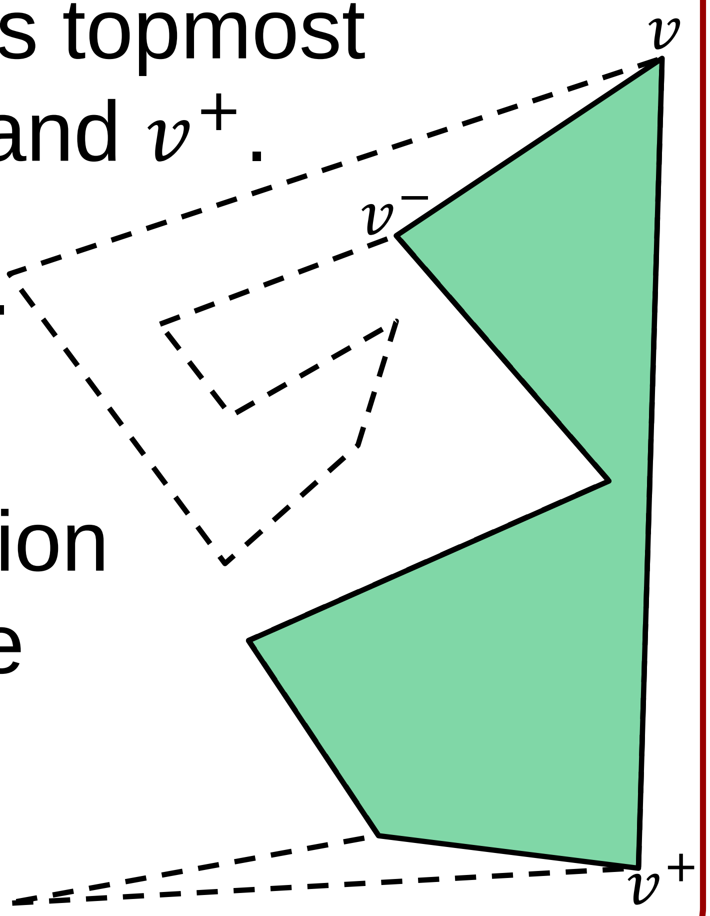
Generating Monotone Mountains

Proof:

Given a piece, we can find its topmost vertex v, with neighbors v^- and v^+.

Assume v^+ is lower than v^-.

We will show that v^+ is the bottom of the partitioned region by showing it connects to the chain coming from $\overline{vv^-}$.



Generating Monotone Mountains

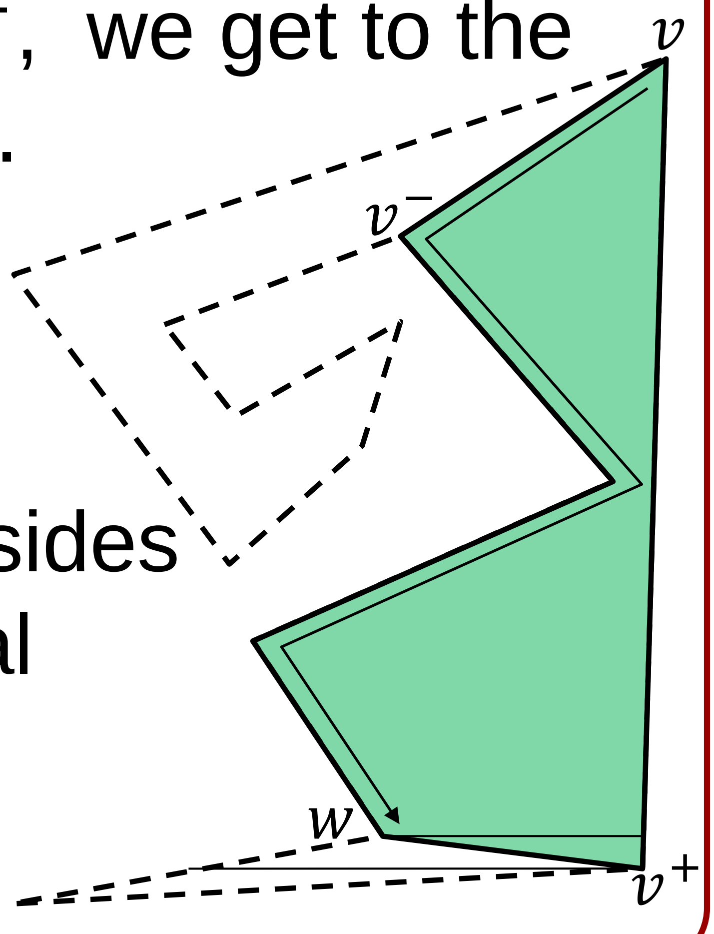
Proof:

Following the chain down v^-, we get to the last vertex, $w \in P$, above v^+.

There is a trapezoid supported by w and v^+.

If w and v^+ are on different sides of the trapezoid, the diagonal to v^+ must be added.

$\Rightarrow v^+$ is a partition endpoint



Generating Monotone Mountains

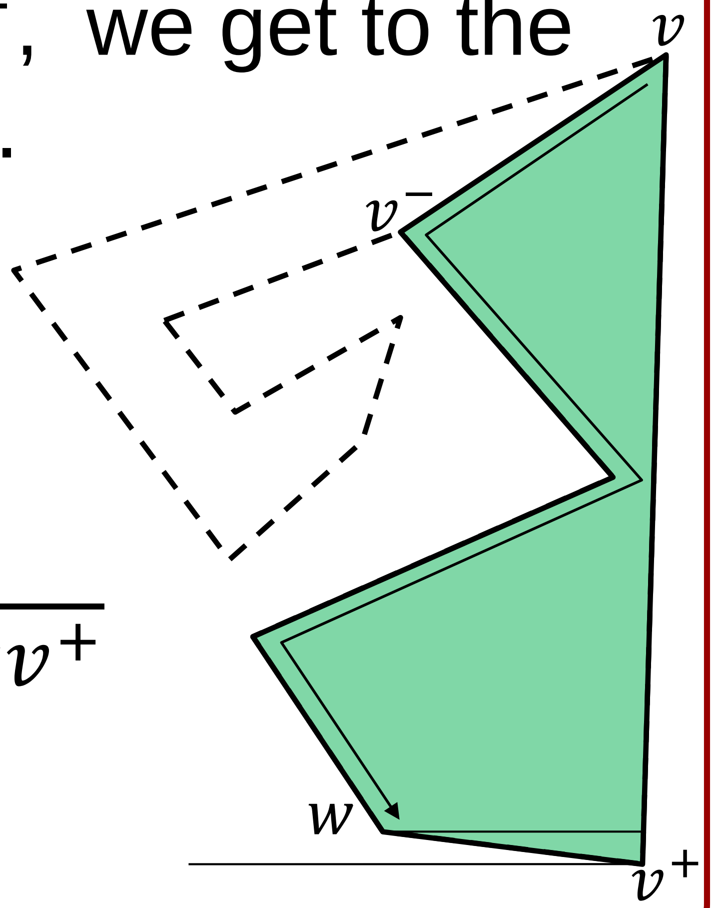
Proof:

Following the chain down v^-, we get to the last vertex, $w \in P$, above v^+.

There is a trapezoid supported by w and v^+.

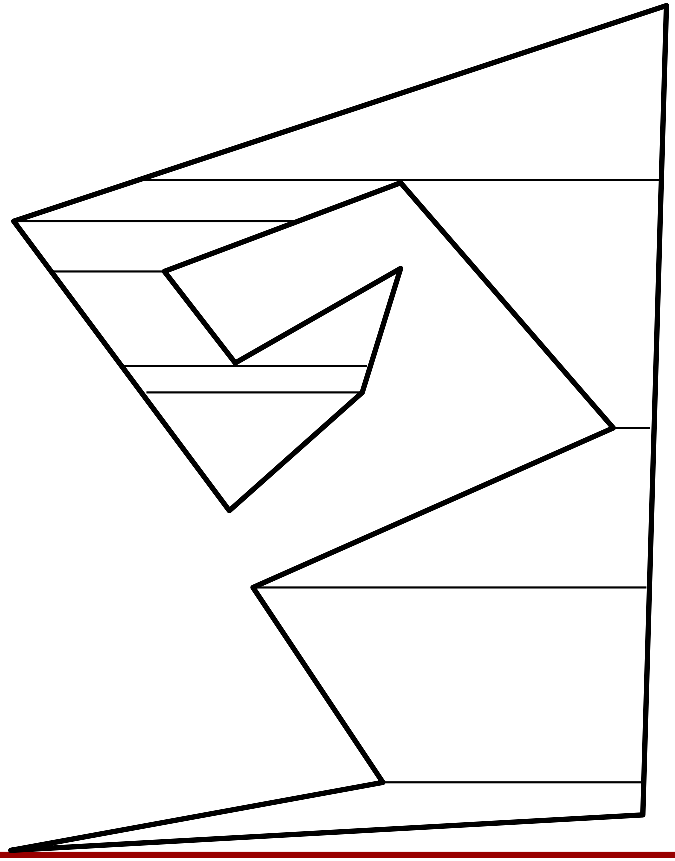
If w and v^+ are on the same side of the trapezoid, then $\overline{wv^+}$ is already on P.

$\Rightarrow v^+$ is a partition endpoint



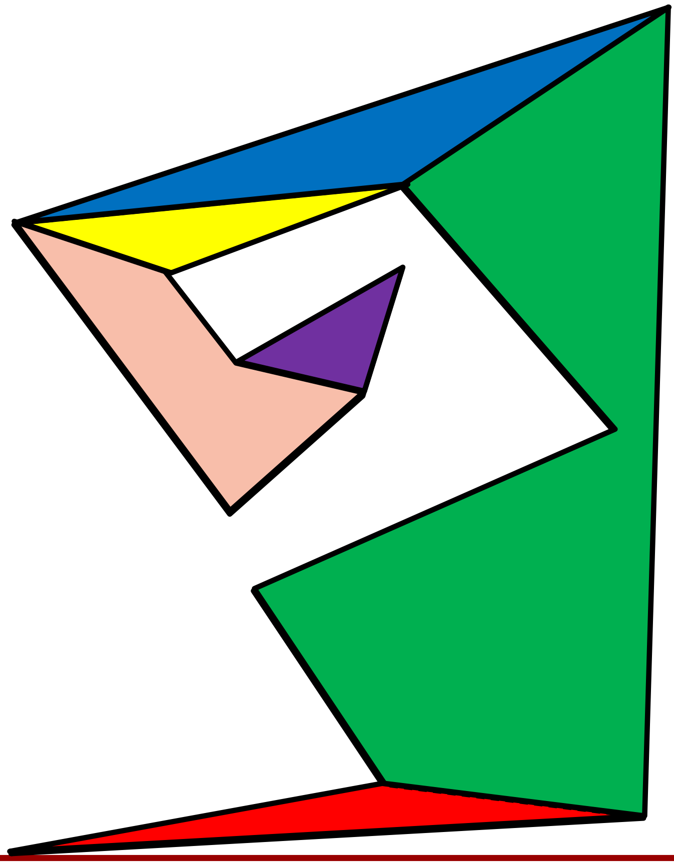
Triangulation

- Triangulate(P):
  - Construct a trapezoidalization



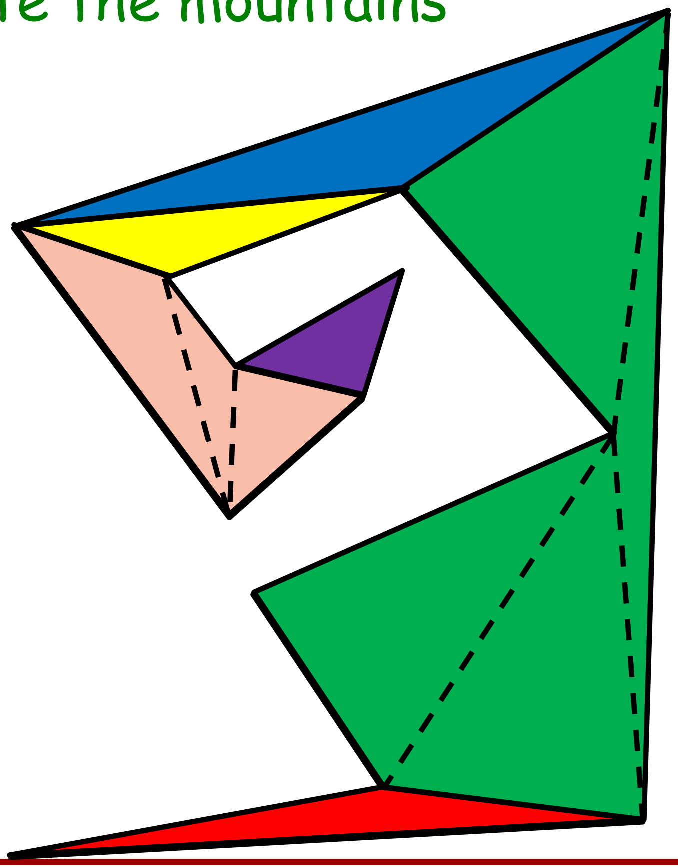
Triangulation

- Triangulate(P):
  - Construct a trapezoidalization
  - Connect supporting vertices from different sides.



Triangulation

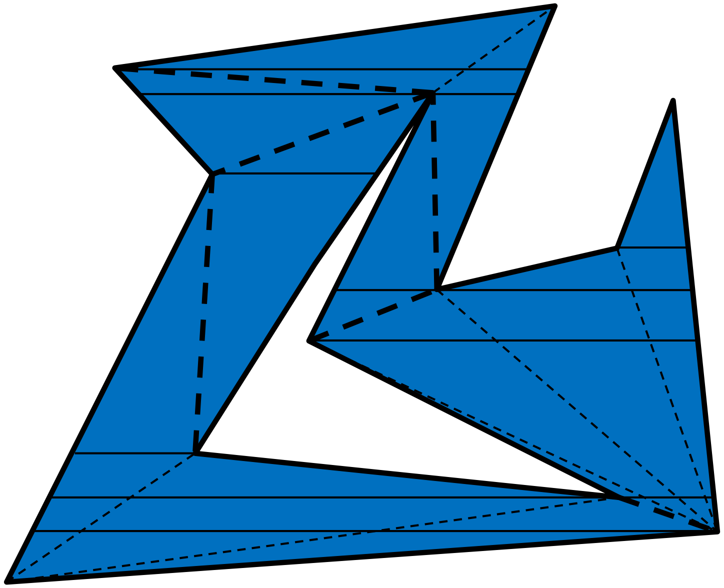
- Triangulate(P):
  - Construct a trapezoidalization
  - Connect supporting vertices from different sides.
  - Triangulate the mountains



Triangulation

Note:

The algorithms for triangulating via monotone polygons and monotone mountains works for polygons with disconnected boundaries.



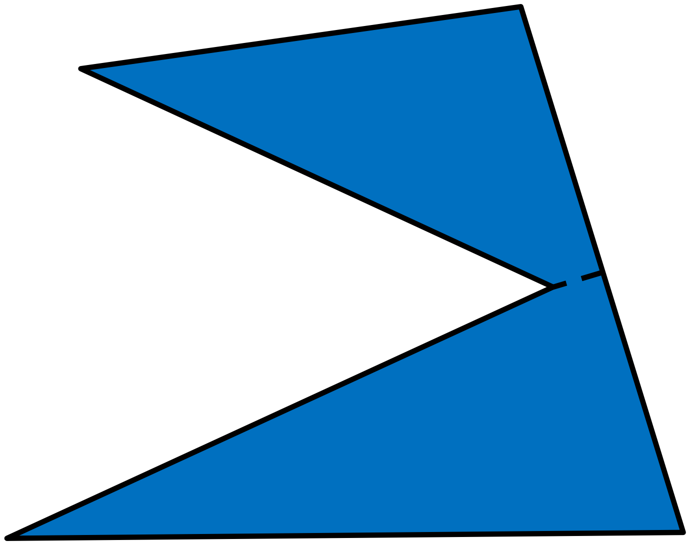
Outline

- Triangle Partitions
- Convex Partitions

Convex Partitions

Definition:

A *convex partition by segments* of a polygon P is a decomposition of P into convex polygons obtained by introducing arbitrary segments.

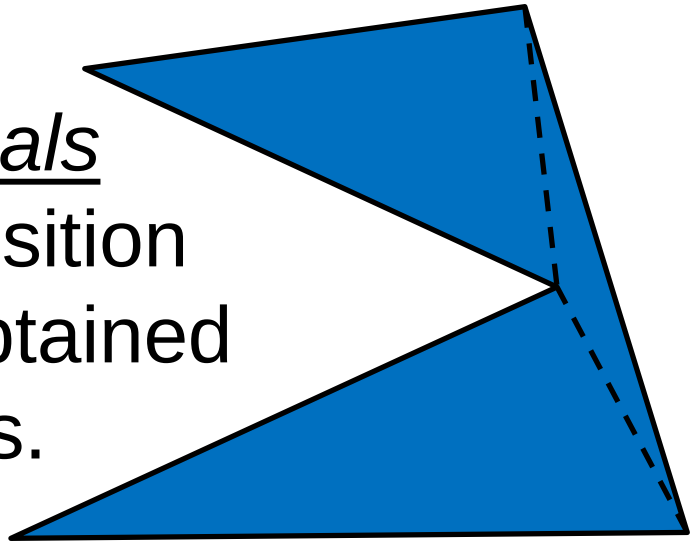


Convex Partitions

Definition:

A *convex partition* *by segments* of a polygon P is a decomposition of P into convex polygons obtained by introducing arbitrary segments.

A *convex partition* *by diagonals* of a polygon P is a decomposition of P into convex polygons obtained by only introducing diagonals.



Convex Partitions

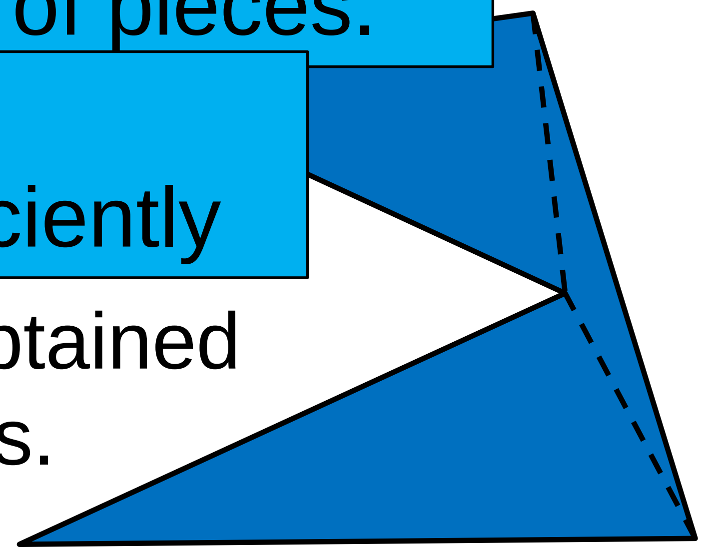
<u>Definition</u>:

A *convex partition <u>by segments</u>* of a polygon P is
a dec … s
obtai … s.

<u>Challenge</u>:
Compute a convex partition with the smallest number of pieces.

A *convex pa*
of a polygon
of P into convex polygons obtained
by only introducing diagonals.

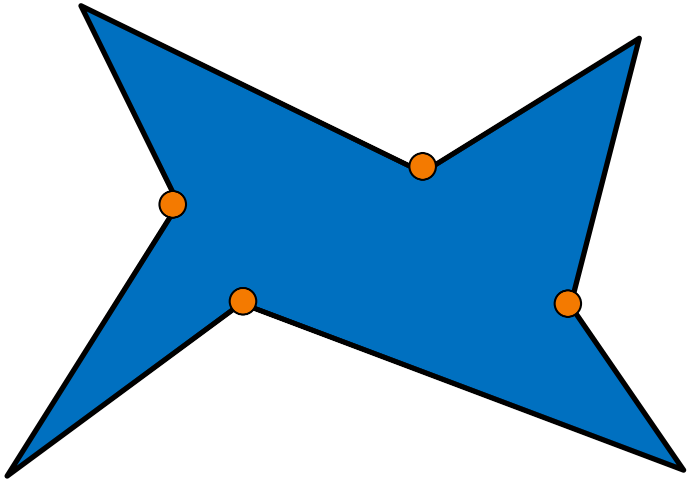
<u>Challenge^2</u>:
Compute it efficiently



Convex Partitions (by Segments)

Claim (Chazelle):
Assume the polygon P has r reflex vertices.
If Φ is the fewest number of polygons required for a convex partition by segments of P then:

$$\lceil r/2 \rceil + 1 \leq \Phi \leq r + 1$$



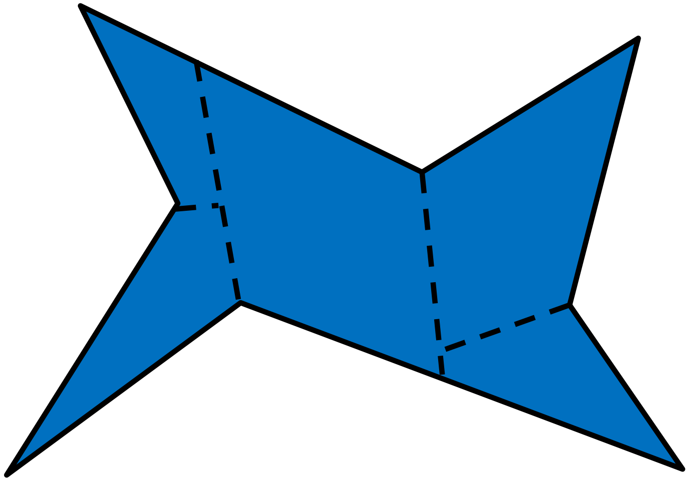
Convex Partitions (by Segments)

Proof ($\Phi \leq r + 1$):

For each reflex vertex, add the bisector.

Because the segment bisects, the reflex angle splits into two (equal) convex angles.

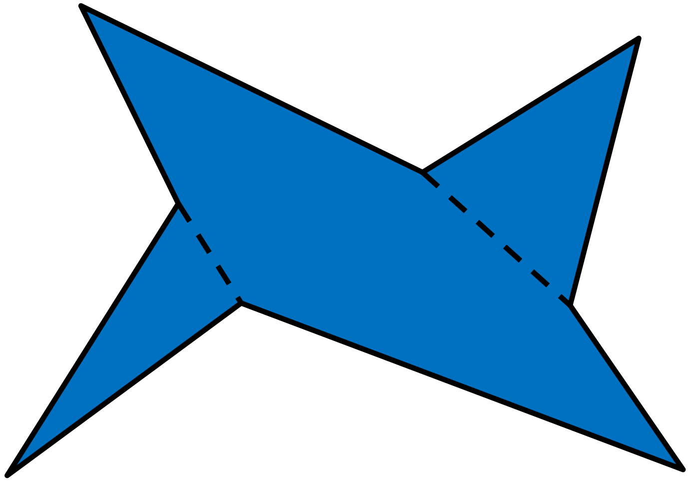
Doing this for each reflex vertices, gives a convex partition with $r + 1$ pieces.



Convex Partitions (by Segments)

Proof ($\lceil r/2 \rceil \leq \Phi$):

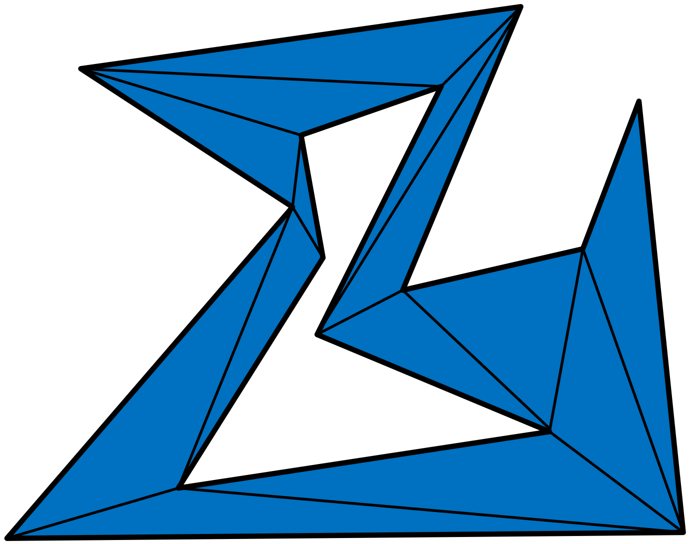
Each reflex vertex needs to be split and each introduced segment can split at most two reflex vertices.



Convex Partitions (by Diagonals)

Definition:

A diagonal in a convex partition is *essential for vertex* $v \in P$ if removing the diagonal creates a piece that is not convex at v.



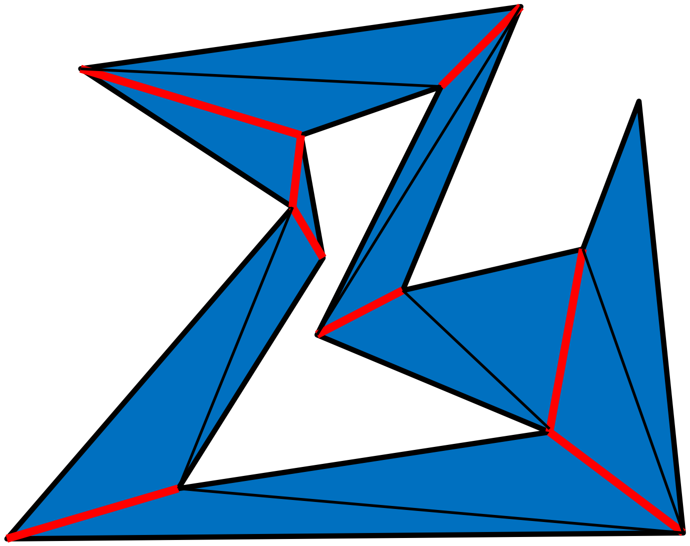
Convex Partitions (by Diagonals)

<u>Definition</u>:

A diagonal in a convex partition is *essential for vertex* $v \in P$ if removing the diagonal creates a piece that is not convex at v.

<u>Note</u>:

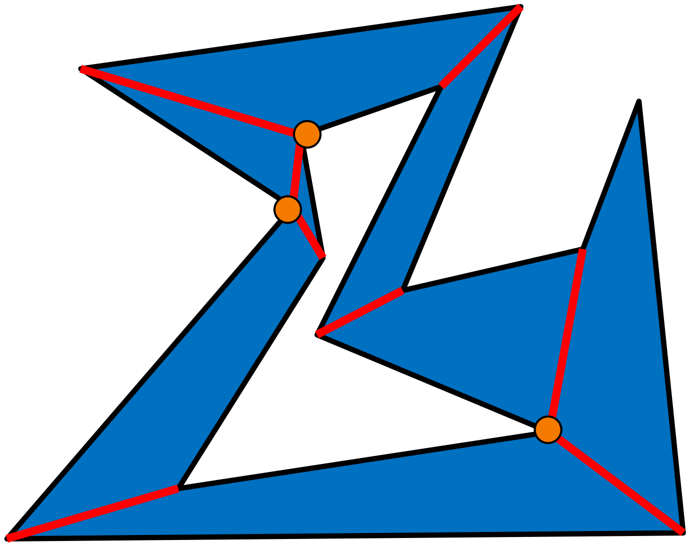
If v is convex, the diagonal cannot be essential for v.



Convex Partitions (by Diagonals)

Claim:

If v is a reflex vertex, it can have at most two essential diagonals.



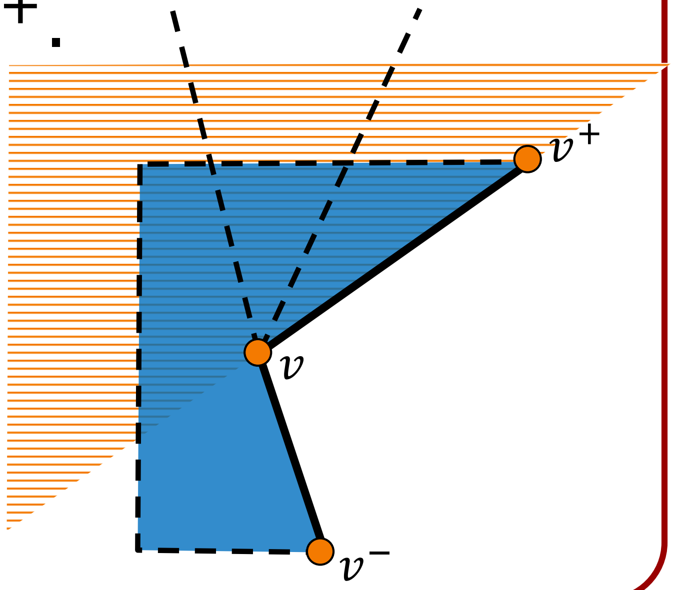
Convex Partitions (by Diagonals)

Proof:

Given a reflex vertex v, let v^- and v^+ be the vertices immediately before and after v in P.

There can be at most one essential segment in the half-space to the right of $\overline{vv^+}$.

(If there were two, we could remove the one closer to $\overline{vv^+}$.)

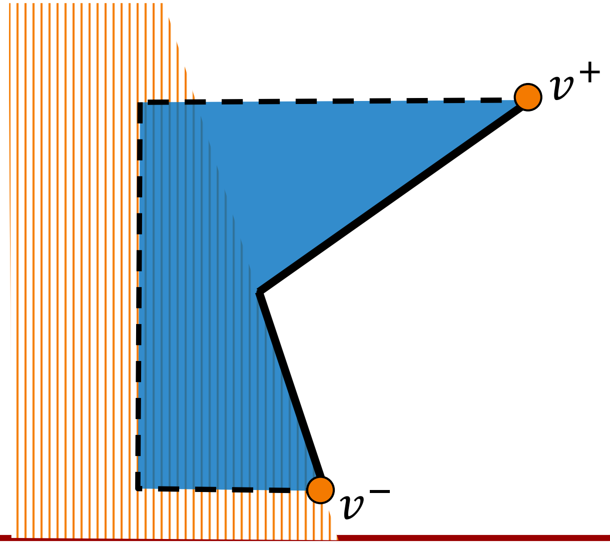


Convex Partitions (by Diagonals)

Proof:

Given a reflex vertex v, let v^- and v^+ be the vertices immediately before and after v in P.

Similarly, there can be at most one essential segment in the half-space to the right of $\overline{vv^-}$.



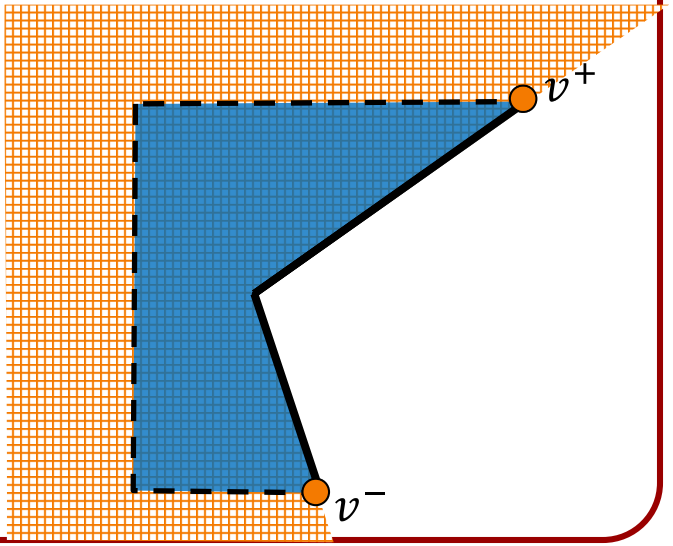
Convex Partitions (by Diagonals)

Proof:

Given a reflex vertex v, let v^- and v^+ be the vertices immediately before and after v in P.

Similarly, there can be at most one essential segment in the half-space to the right of $\overline{vv^-}$.

Since the two half-spaces cover the interior of the vertex there are at most two essential diagonals at v.



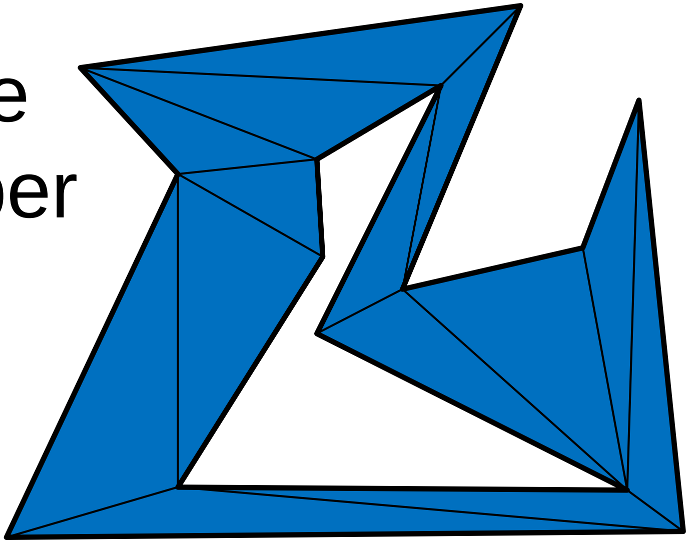
Convex Partitions (by Diagonals)

Algorithm (Hertl & Mehlhorn):

Start with a triangulation and remove diagonals that are not essential for either endpoint.

Claim:

This algorithm is never worse than $4\times$ optimal in the number of convex pieces.



Convex Partitions (by Diagonals)

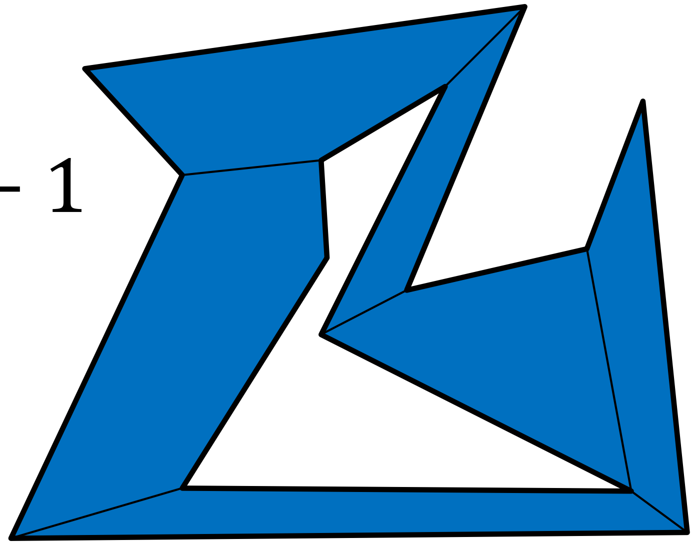
Proof:

When the algorithm terminates, every remaining diagonal is essential for some (reflex) vertex.

Each reflex vertex can have at most two essential diagonals.

$\Rightarrow$ There can be at most $2r + 1$ pieces in the partition.

Since at least $\lceil r/2 \rceil + 1$ are required, the result is within $4 \times$ optimal.



Convex Partitions

Why do we care?

Convex polygons are easier to intersect against.

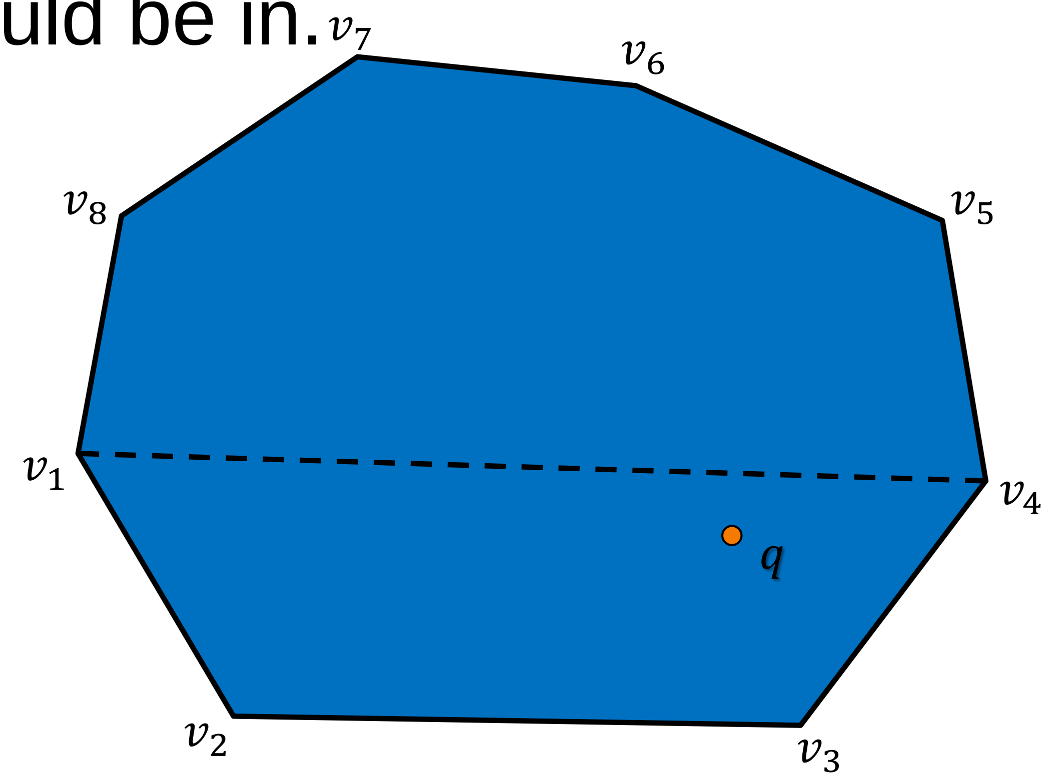
For a polygon with n vertices:

- Testing if a point is inside is $O(\log n)$.
- Testing if a line intersects is $O(\log n)$.
- Testing if two polygons intersect is $O(\log n)$.

Point/Convex-Polygon Intersection

Algorithm:

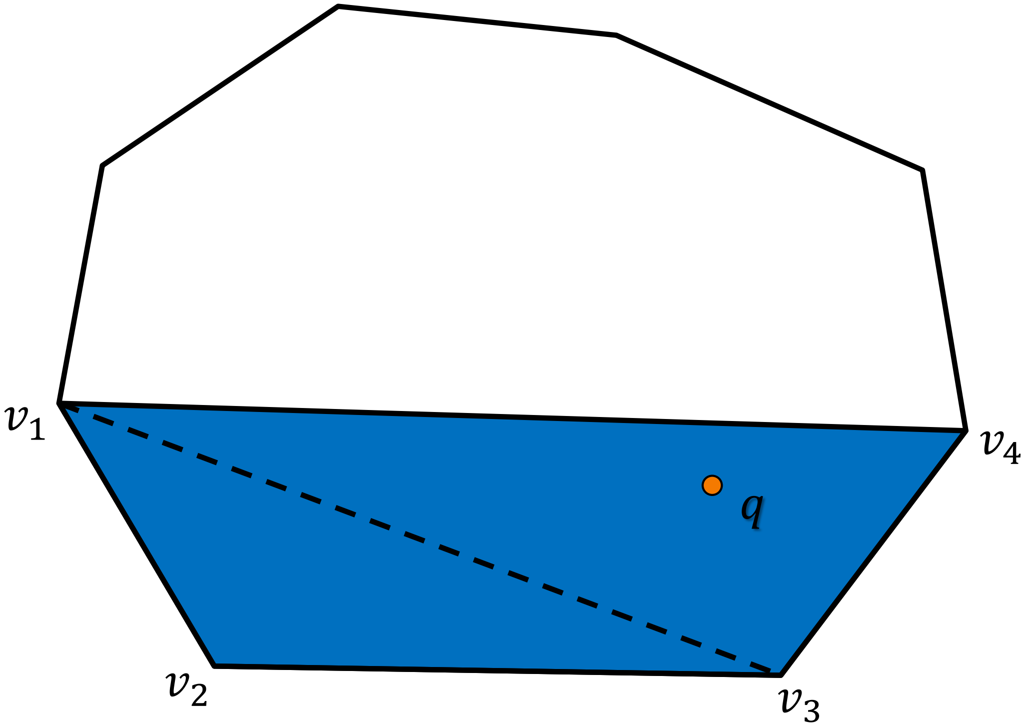
Recursively split the polygon in half and test the half the point could be in.



Point/Convex-Polygon Intersection

Algorithm:

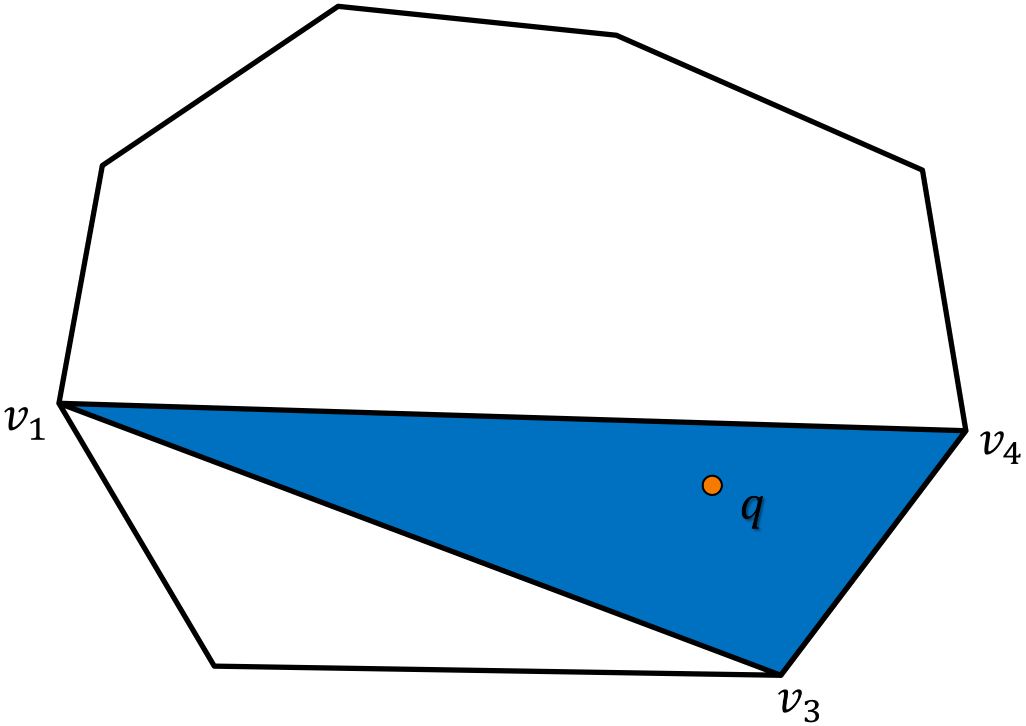
Recursively split the polygon in half and test the half the point could be in.



Point/Convex-Polygon Intersection

Algorithm:

Recursively split the polygon in half and test the half the point could be in.



Point/Convex-Polygon Intersection

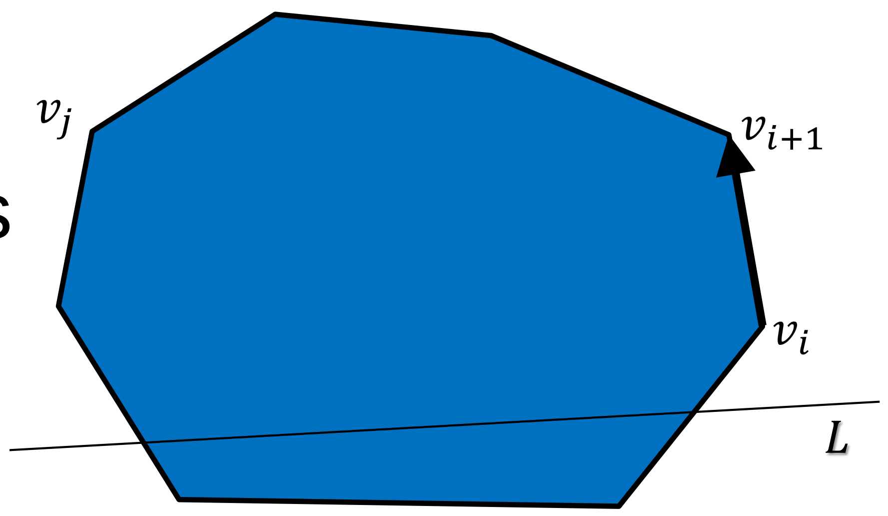
InConvexPolygon(q , $\{v_1, \ldots, v_n\}$)

- if($n == 3$)
  - » return InTriangle(q , $\{v_1, v_2, v_3\}$);
- if(Left($v_1, v_{n/2}$, q))
  - » return InConvexPolygon(q , $\{v_{n/2}, \ldots, v_1\}$);
- else
  - » return InConvexPolygon(q , $\{v_1, \ldots, v_{n/2}\}$);

Line/Convex-Polygon Intersection

Note:

Given a convex polygon P, a line segment L, and vertices $v_i, v_j \in P$ on the same side of L. Assume v_i is closer to L than v_j:



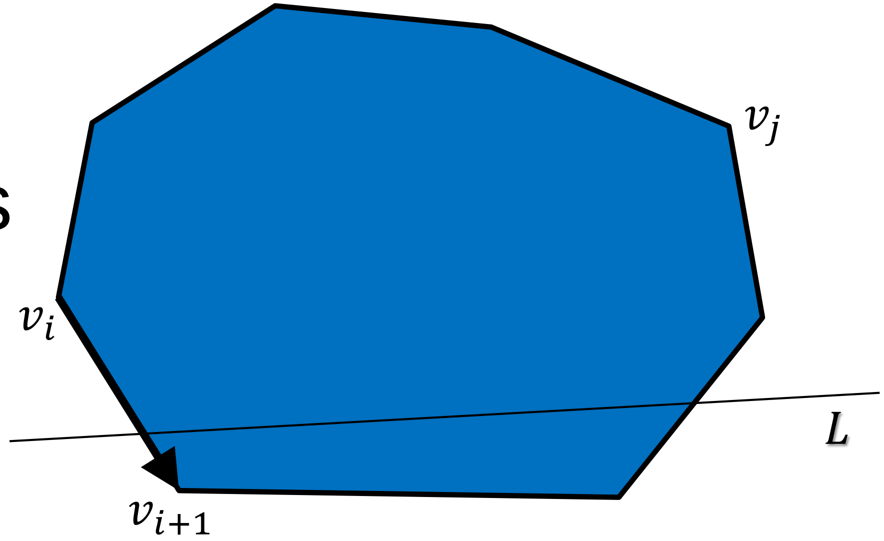
If the vector $\overrightarrow{v_i v_{i+1}}$ points away from L, then L can only intersect P along the chain $\{v_j, v_{j+1}, \dots, v_i\}$.

Line/Convex-Polygon Intersection

Note:

Given a convex polygon P, a line segment L, and vertices $v_i, v_j \in P$ on the same side of L. Assume v_i is closer to L than v_j:

If the vector $\overrightarrow{v_i v_{i+1}}$ points away from L, then L can only intersect P along the chain $\{v_j, v_{j+1}, \dots, v_i\}$.



Otherwise, L can only intersect P along the chain $\{v_i, v_{i+1}, \dots, v_j\}$.

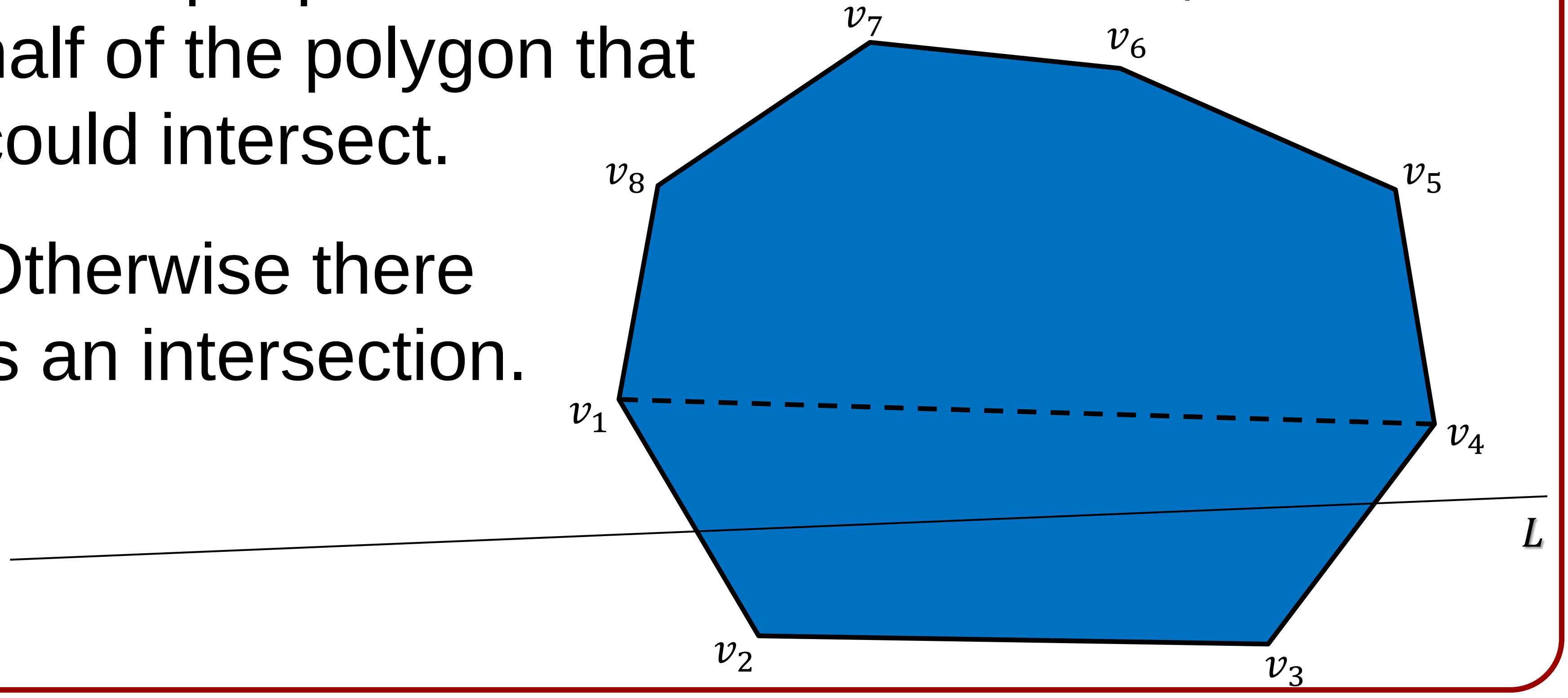
Line/Convex-Polygon Intersection

Algorithm:

Recursively split the polygon in half.

If the split points are on the same side, test the half of the polygon that could intersect.

Otherwise there is an intersection.



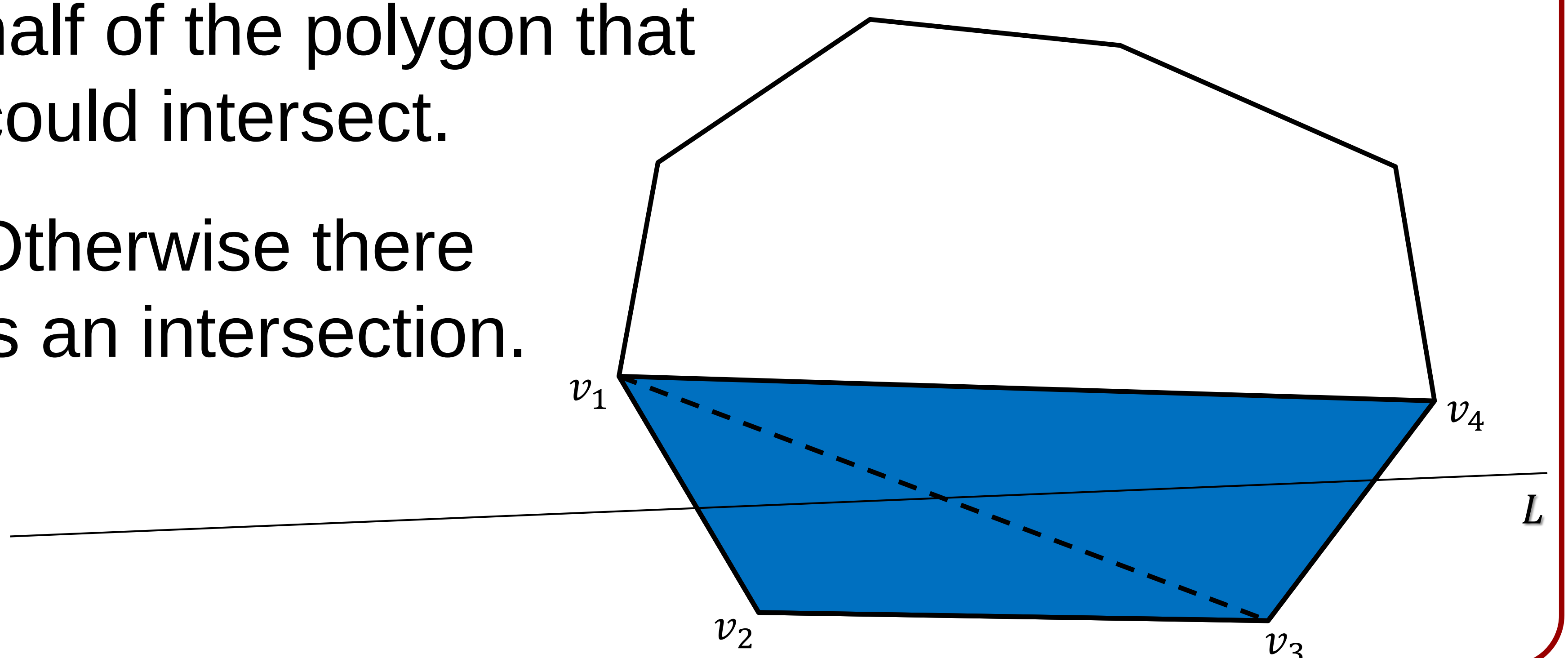
Line/Convex-Polygon Intersection

Algorithm:

Recursively split the polygon in half.

If the split points are on the same side, test the half of the polygon that could intersect.

Otherwise there is an intersection.



Line/Convex-Polygon Intersection

IsectConvexPolygon($\{l_1, l_2\}$, $\{v_1, \dots, v_n\}$)

- if($n == 3$)
  - » return IsectTriangle($\{l_1, l_2\}$, $\{v_1, v_2, v_3$);
- if(Left(l_1 , l_2 , v_1)!=Left(l_1 , l_2 , $v_{n/2}$))
  - » return true;
- else
  - » if(Dist($\{l_1, l_2\}$, v_1)<Dist($\{l_1, l_2\}$, $v_{n/2}$)
    - – if(Dist($\{l_1, l_2\}$, v_1)<Dist($\{l_1, l_2\}$, v_2)
      - • return IsectConvexPolygon($\{l_1, l_2\}$, $\{v_{n/2}, \dots, v_1\}$);
    - – else
      - • return IsectConvexPolygon($\{l_1, l_2\}$, $\{v_1, \dots, v_{n/2}\}$);
  - » else…