Convex Hulls (3D)

O'Rourke, Chapter 4

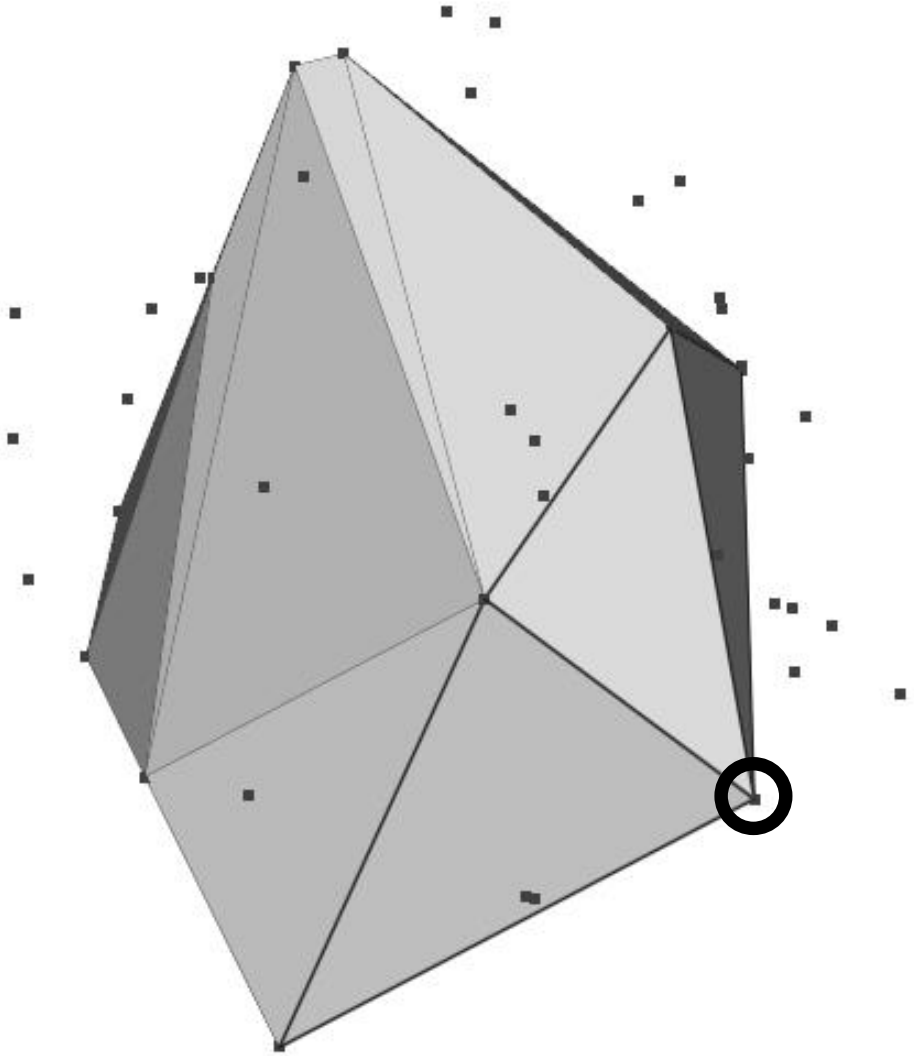
Outline

- Incremental
- Divide-and-Conquer
- Higher Dimensions

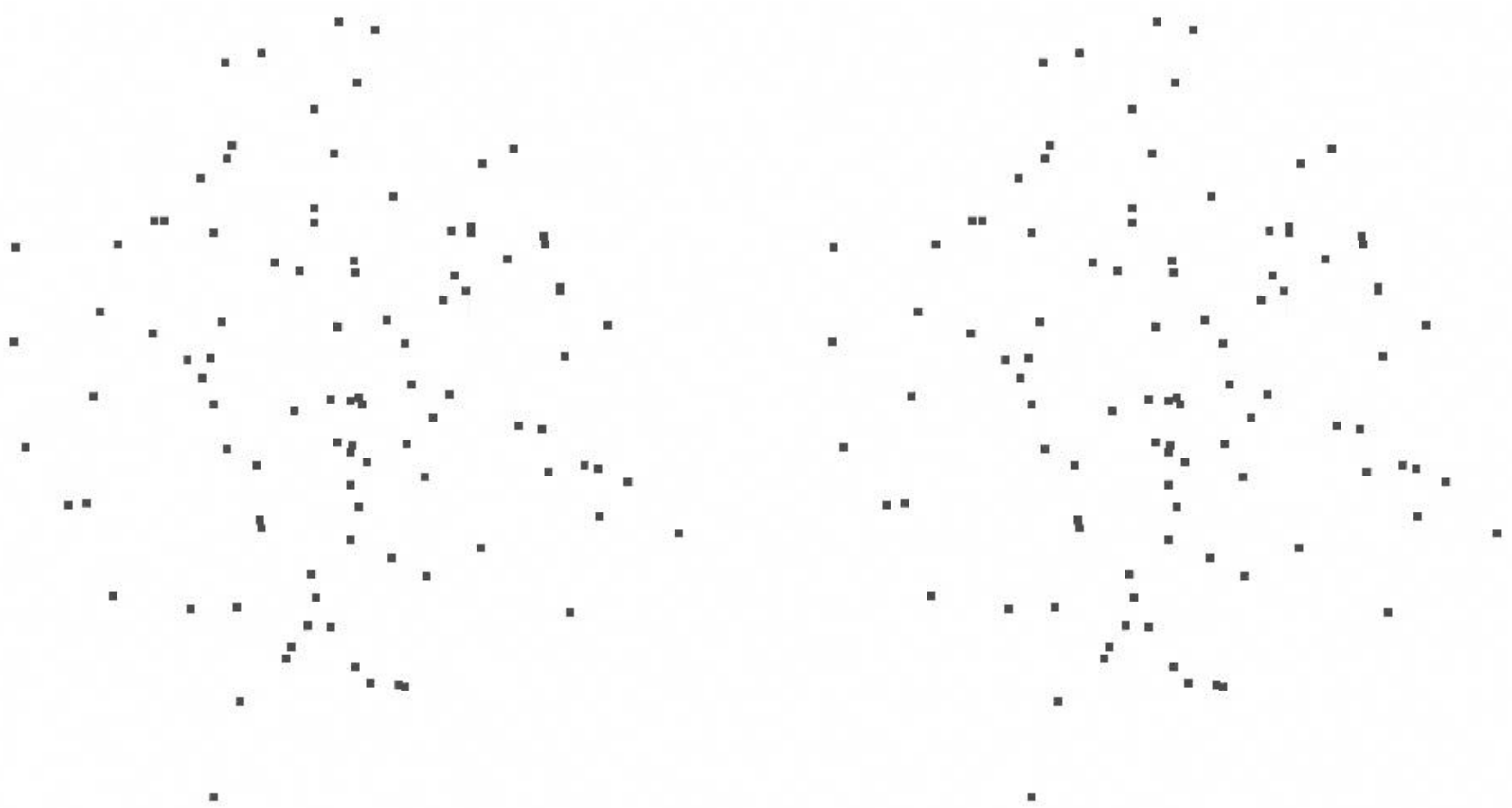
Incremental Algorithm

- Start with a small hull.
- Iteratively add the rest of the points:
  - Connect the new point to the old hull along a cone
  - Remove the old faces

Incremental Algorithm



Incremental Algorithm



Incremental Algorithm

IncrmentalHull(P):

- H ← InitializeHull(P)
- for p ∈ P
  - » AddPointToHull(p , H)
  - » P.remove(p)

Incremental Algorithm

InitializeHull(P):

- for i ∈ [0, |P|-3)
  - » if(!IsCollinear(p_i , p_{i+1} , p_{i+2})
    - return { { p_i , p_{i+1} , p_{i+2} } , { p_i , p_{i+2} , p_{i+1} } }
- die("Points are collinear")

Incremental Algorithm

AddPointToHull(p , H):

- for f ∈ Faces(H)
  - » f.visible = Right(f.v[0] , f.v[1] , f.v[2] , p)
- for e ∈ Edges(H)
  - » if (e.face[0].visible && !e.face[1].visible)
    - H.insert(Face(e.v[0] , e.v[1] , p))
    - H.remove(e.face[0])
  - » else if(!e.face[0].visible && e.face[1].visible)
    - H.insert(Face(e.v[1] , e.v[0] , p))
    - H.remove(e.face[1])
- for f ∈ Faces(H)
  - » f.visible = false

Complexity: $O(|H|)$

Incremental Algorithm

AddPointToHull(p , H):

- for f ∈ Faces(H)
  - » f.visible = Right(f.v[0] , f.v[1] , f.v[2] , p)
- for e ∈ Edges(H)
  - »
- for
  - » f.visible = false

We need a data-structure (e.g. half-edge) that allows for efficient traversal of the edges.

We can't add/remove faces while we are traversing since this may break the correctness of the mesh representation.

Store temporary information:

- Vertices store the new cone edge
- Edges store the new cone face

Complexity: $O(|H|)$

Incremental Algorithm

IncrmentalHull(P):

- H ← InitializeHull(P)
- for p ∈ P
  - » AddPointToHull(p , H)
  - » P.remove(p)

Note:
Since we use the **Right** test, the resulting hull may contain non-extremal points.

Complexity: $O(|P|^2)$

Outline

- Incremental
- Divide-and-Conquer
- Higher Dimensions

Divide-and-Conquer [Chan, 2003]

Recall:

Given a point $p = (p_x, p_y, p_z) \in \mathbb{R}^3$ and an angle α, the projection of the point p onto the xy-plane after being rotated by α degrees about the x-axis is:

$$\pi_\alpha(p) = \left(p_x, \left\langle \left(p_y, p_z\right), (\cos\alpha\,, \sin\alpha) \right\rangle\right)$$

Divide-and-Conquer [Chan, 2003]

Recall:

If a region $\Omega \subset \mathbb{R}^d$ is convex then $T(\Omega)$ will also be convex for any affine transformation:

$$T(p) = Ap + q$$

Affine transformations take lines to lines:

- If ℓ is a line segment with end-points in $T(\Omega)$ then $T^{-1}(\ell)$ is a line segment with end-points in Ω.
- If ℓ is a line segment and there exists $p \in \ell$ s.t. $p \notin T(\Omega)$, then $T^{-1}(p) \in T^{-1}(\ell)$ and $T^{-1}(p) \notin \Omega$.

$\Rightarrow$ If $T(\Omega)$ is not convex, then Ω isn't convex either.

Divide-and-Conquer [Chan, 2003]

Recall:

If a region $\Omega \subset \mathbb{R}^d$ is convex then $T(\Omega)$ will also be convex for any affine transformation:

$$T(p) = Ap + q$$

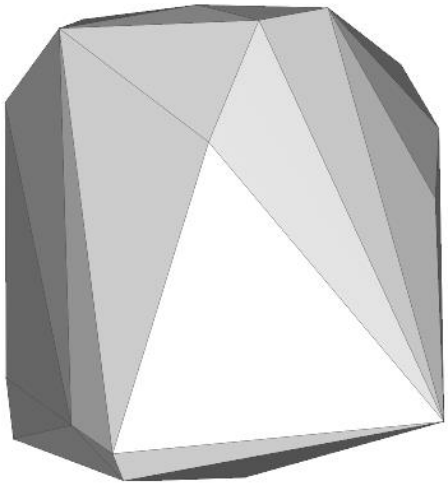
If the affine transformation is orientation-preserving ($\det(A) > 0$) then Left, LeftOn, Right, and RightOn relations are also preserved.

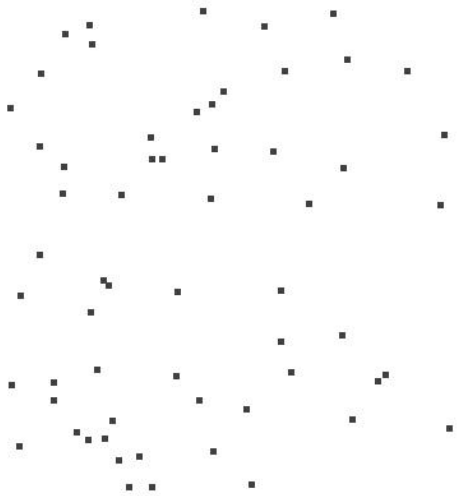
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

- Compute a “movie” of the lower-hull of the projection of the point set onto the xy-plane as the points are rotated about the x-axis.



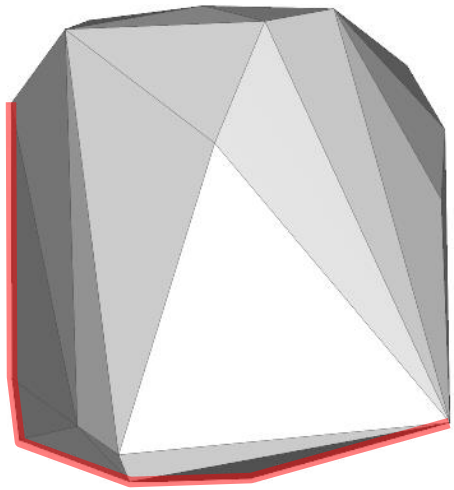


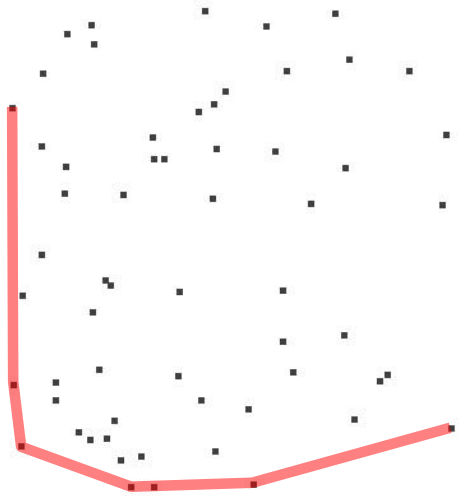
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

- Compute a “movie” of the lower-hull of the projection of the point set onto the xy-plane as the points are rotated about the x-axis.



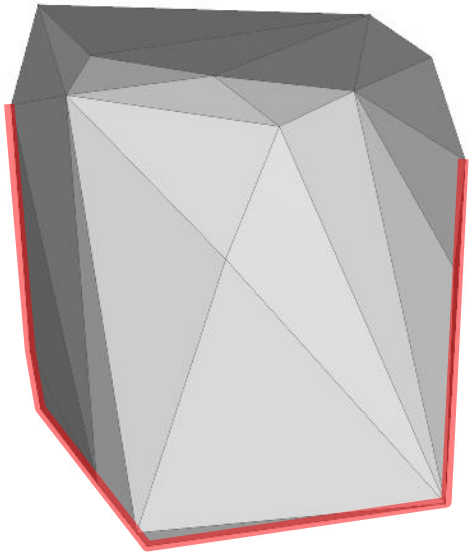


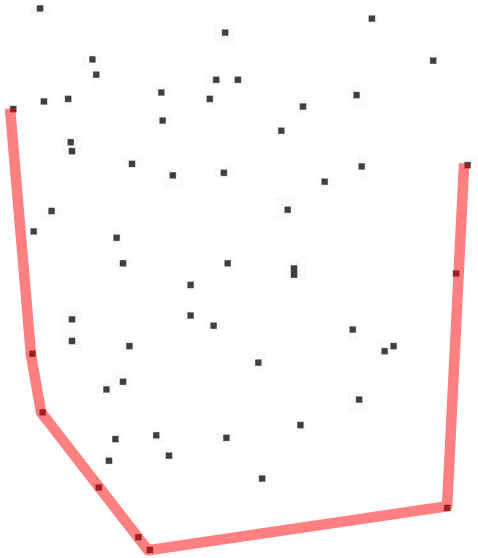
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

- Compute a “movie” of the lower-hull of the projection of the point set onto the xy-plane as the points are rotated about the x-axis.



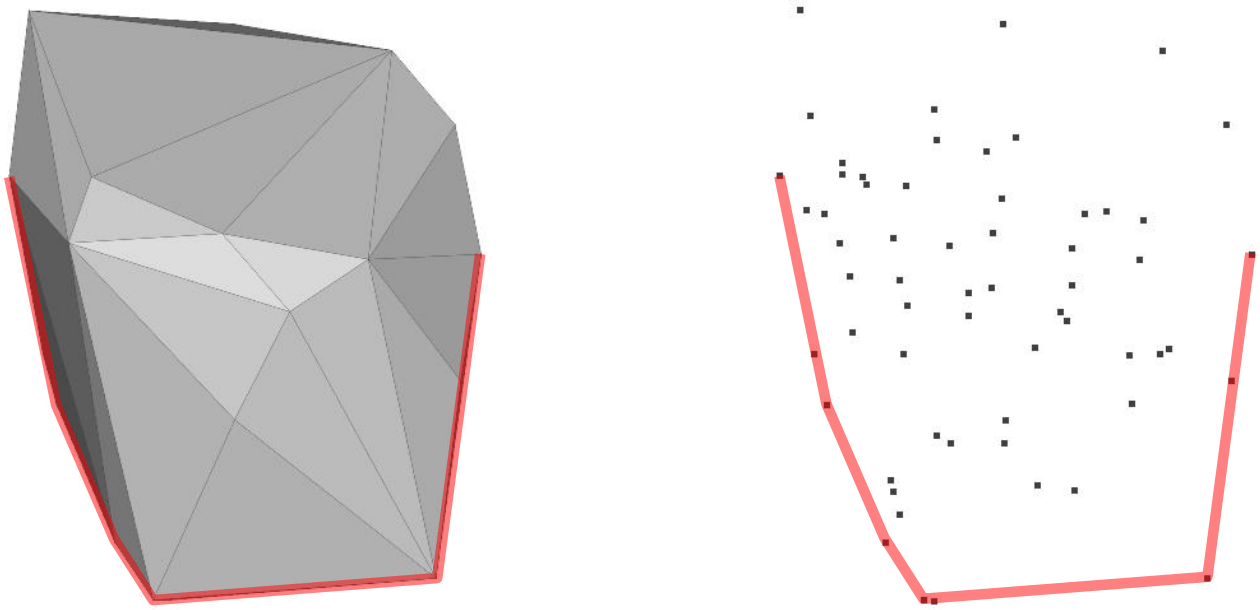


Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

- Compute a “movie” of the lower-hull of the projection of the point set onto the xy-plane as the points are rotated about the x-axis.

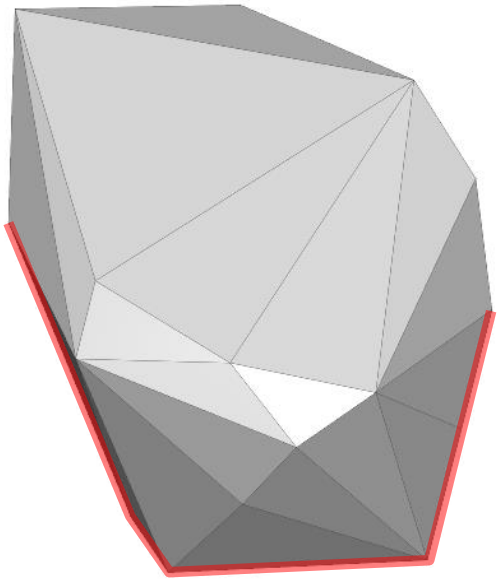


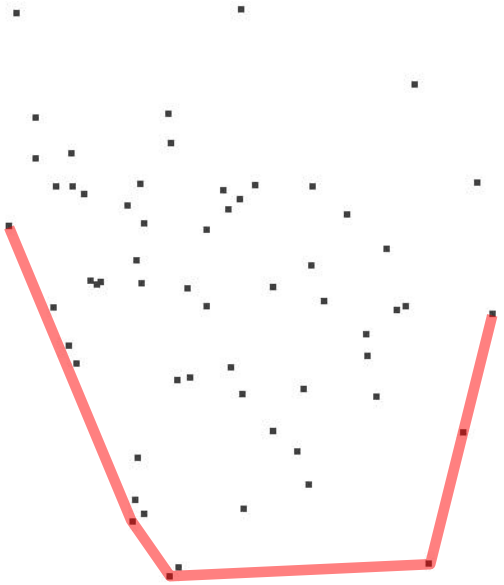
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

- Compute a “movie” of the lower-hull of the projection of the point set onto the xy-plane as the points are rotated about the x-axis.





Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

- Store the points sorted by x-coordinate
- Store previous/next pointers with the points to represent the 2D hull connectivity
- Have a sorted-by-angle list of events at which the 2D hull connectivity changes

Note:
The first and last points are extremal so they will always be on the hull.

Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

Key Idea:

As we rotate the points, the hull changes by:

- A point being inserted between end-points of an edge:
  one edge → two edges
- A point being deleted:
  two edges → one edge

At both events, a hull triangle is generated.

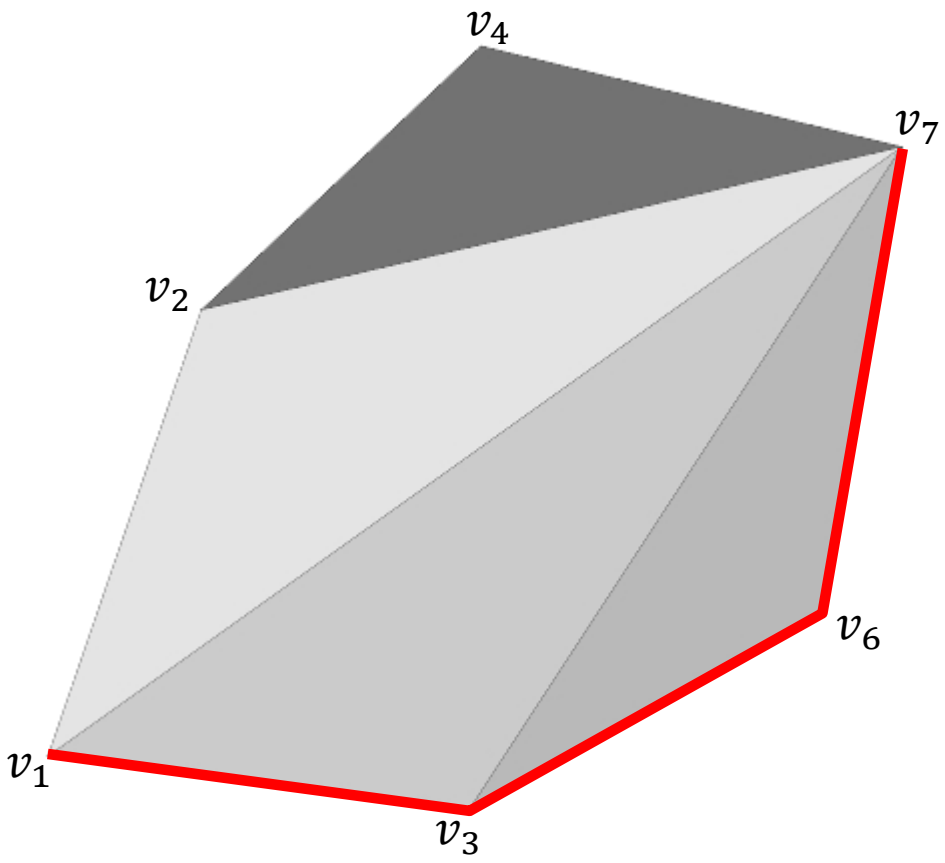
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$$E = \left\{ \begin{array}{c} \boxed{(I, 15^\circ, v_3 v_5 v_6)} \\ (D, 39^\circ, v_1 v_3 v_5) \\ (D, 75^\circ, v_5 v_6 v_7) \\ (I, 90^\circ, v_1 v_4 v_5) \\ (I, 130^\circ, v_1 v_2 v_4) \\ (D, 145^\circ, v_4 v_5 v_7) \end{array} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$



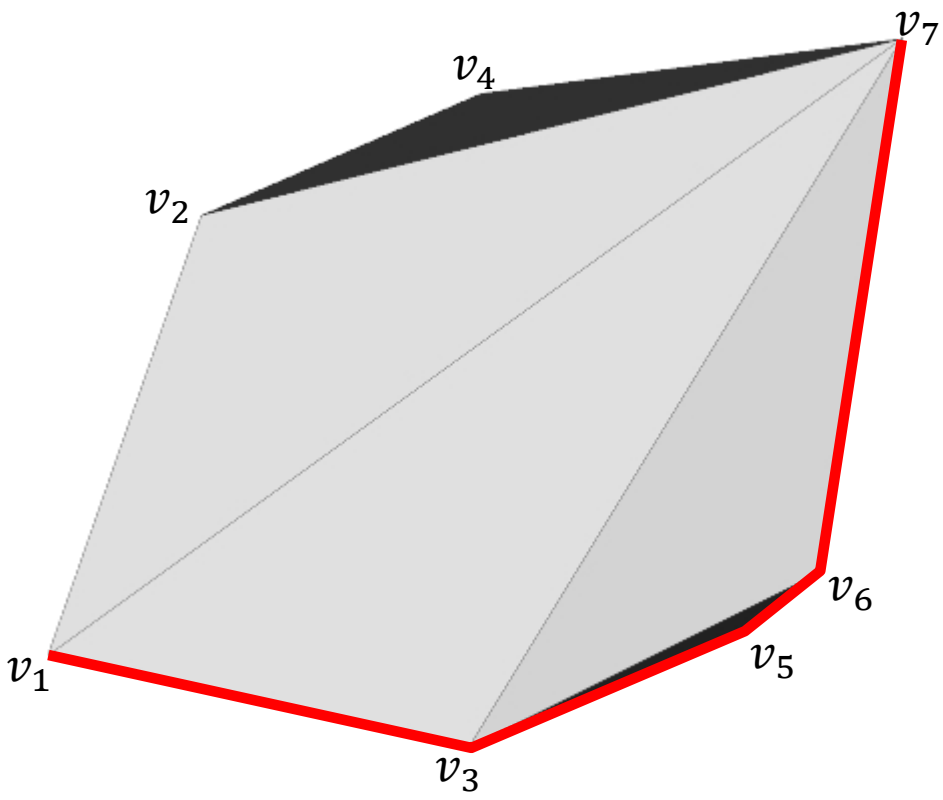
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 20°$

$$E = \left\{ \begin{array}{c} (I, 15°, v_3 v_5 v_6) \\ (D, 39°, v_1 v_3 v_5) \\ (D, 75°, v_5 v_6 v_7) \\ (I, 90°, v_1 v_4 v_5) \\ (I, 130°, v_1 v_2 v_4) \\ (D, 145°, v_4 v_5 v_7) \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

Divide-and-Conquer [Chan, 2003]

Approach:

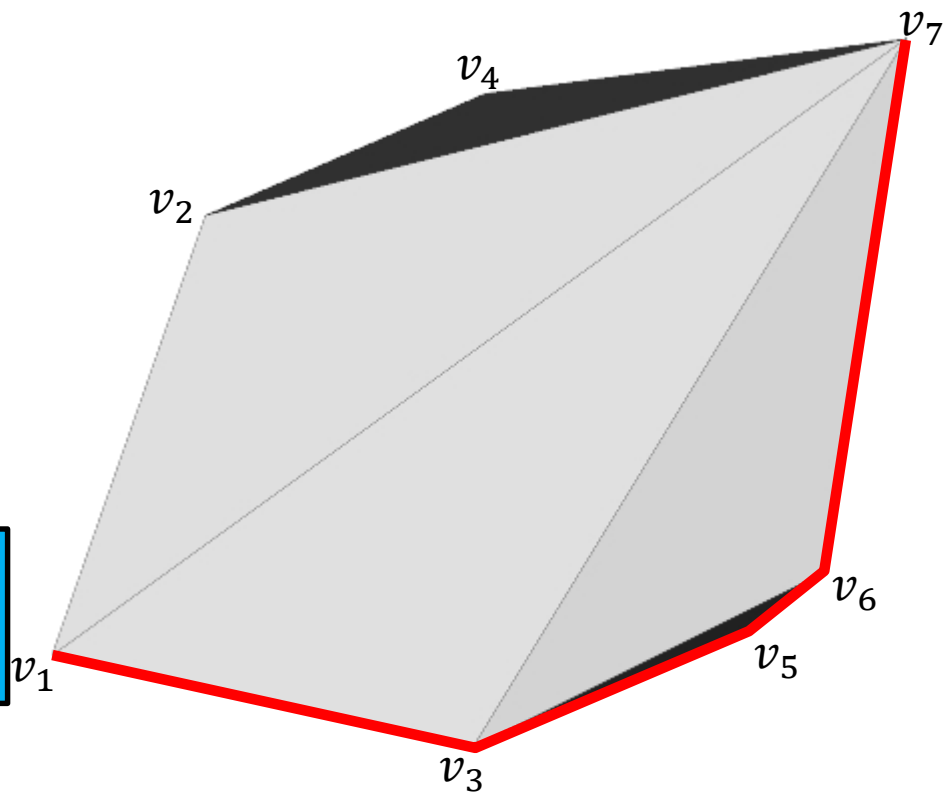
Use a *kinetic* representation.

$\alpha = 20°$

$$E = \left\{ \begin{array}{l} (I, 15°, v_3 v_5 v_6) \\ (D, 39°, v_1 v_3 v_5) \\ (D, 75°, v_5 v_6 v_7) \\ (I, 90°, v_1 v_4 v_5) \\ (I, 130°, v_1 v_2 v_4) \\ (D, 145°, v_4 v_5 v_7) \end{array} \right\}$$

The insert triplet is $\{v_3, v_5, v_6\}$ is a face on the hull.

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$



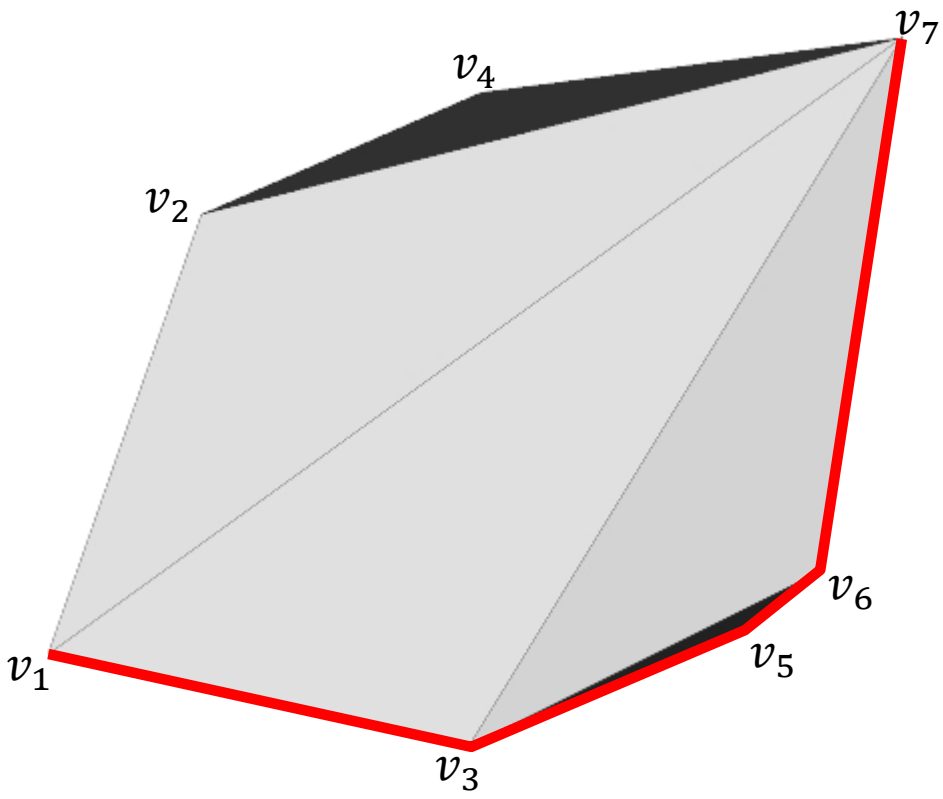
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 20°$

$$E = \begin{Bmatrix} (I, 15°, v_3v_5v_6) \\ \boxed{(D, 39°, v_1v_3v_5)} \\ (D, 75°, v_5v_6v_7) \\ (I, 90°, v_1v_4v_5) \\ (I, 130°, v_1v_2v_4) \\ (D, 145°, v_4v_5v_7) \end{Bmatrix}$$



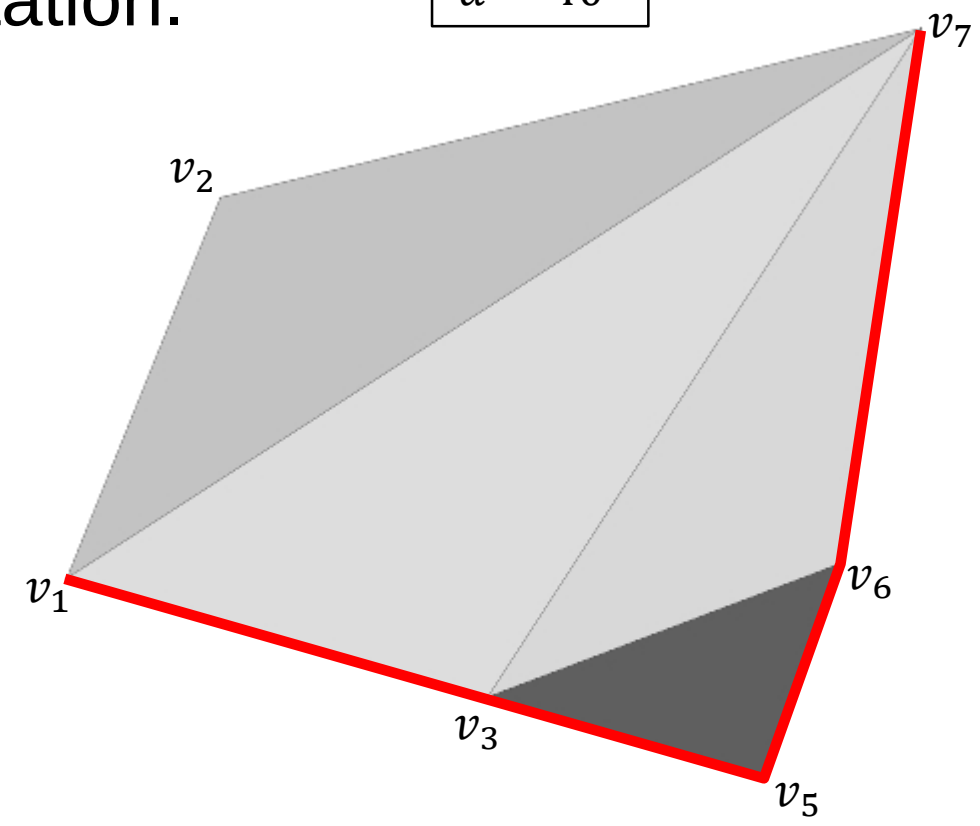
$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$$E = \left\{ \begin{array}{c} (I, 15°, v_3v_5v_6) \\ \boxed{(D, 39°, v_1v_3v_5)} \\ (D, 75°, v_5v_6v_7) \\ (I, 90°, v_1v_4v_5) \\ (I, 130°, v_1v_2v_4) \\ (D, 145°, v_4v_5v_7) \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

Divide-and-Conquer [Chan, 2003]

Approach:

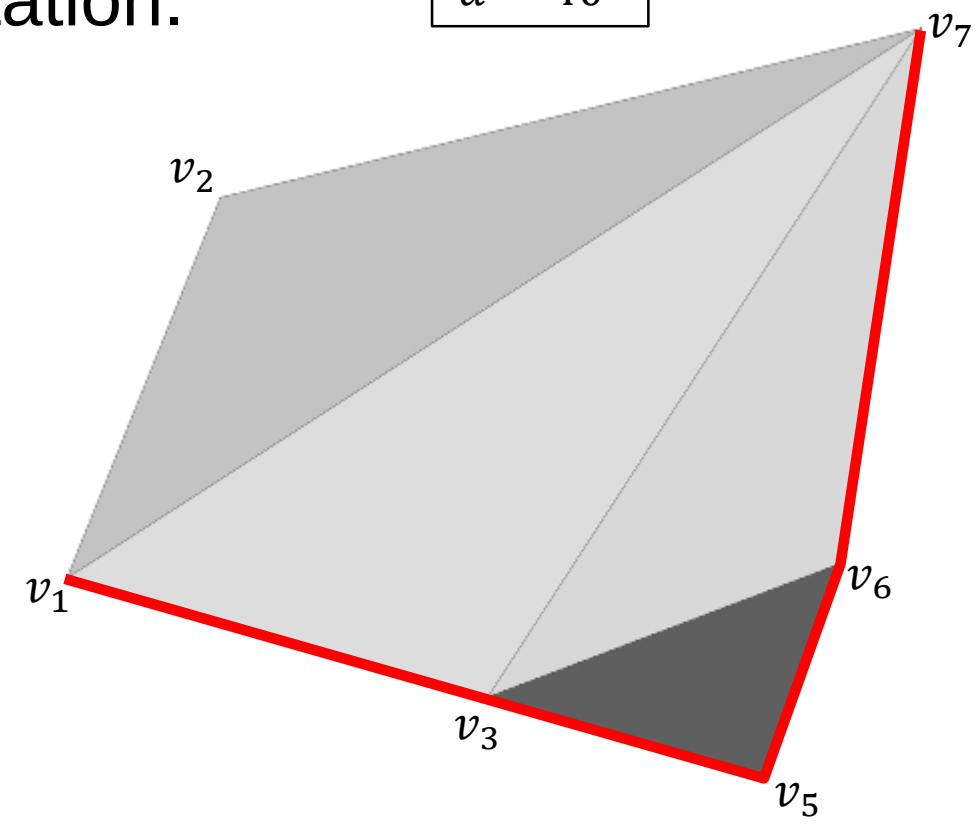
Use a *kinetic* representation.

$$E = \left\{ \begin{array}{c} (I, 15^\circ, v_3 v_5 v_6) \\ \boxed{(D, 39^\circ, v_1 v_3 v_5)} \\ (D, 75^\circ, v_5 v_6 v_7) \\ (I, 90^\circ, v_1 v_4 v_5) \\ (I, 130^\circ, v_1 v_2 v_4) \\ (D, 145^\circ, v_4 v_5 v_7) \end{array} \right\}$$

The delete triplet $\{v_1, v_5, v_3\}$ is a face on the hull.

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

$\alpha = 40^\circ$

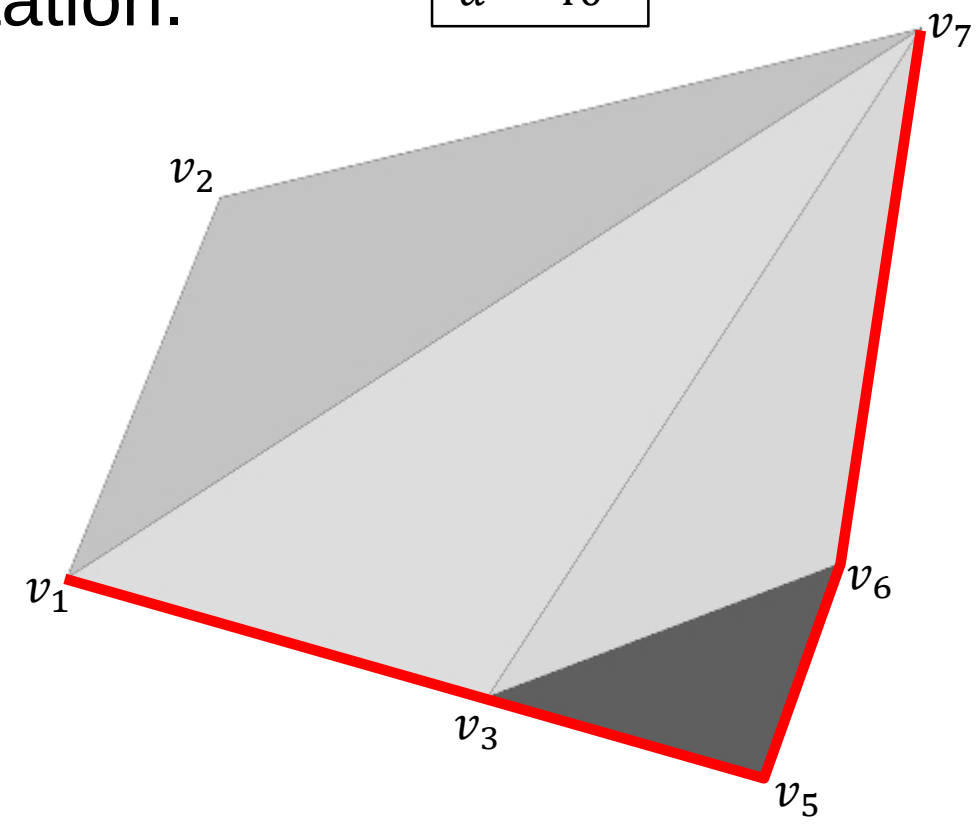


Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$$E = \left\{ \begin{array}{c} (I, 15^\circ, v_3 v_5 v_6) \\ (D, 39^\circ, v_1 v_3 v_5) \\ \boxed{(D, 75^\circ, v_5 v_6 v_7)} \\ (I, 90^\circ, v_1 v_4 v_5) \\ (I, 130^\circ, v_1 v_2 v_4) \\ (D, 145^\circ, v_4 v_5 v_7) \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

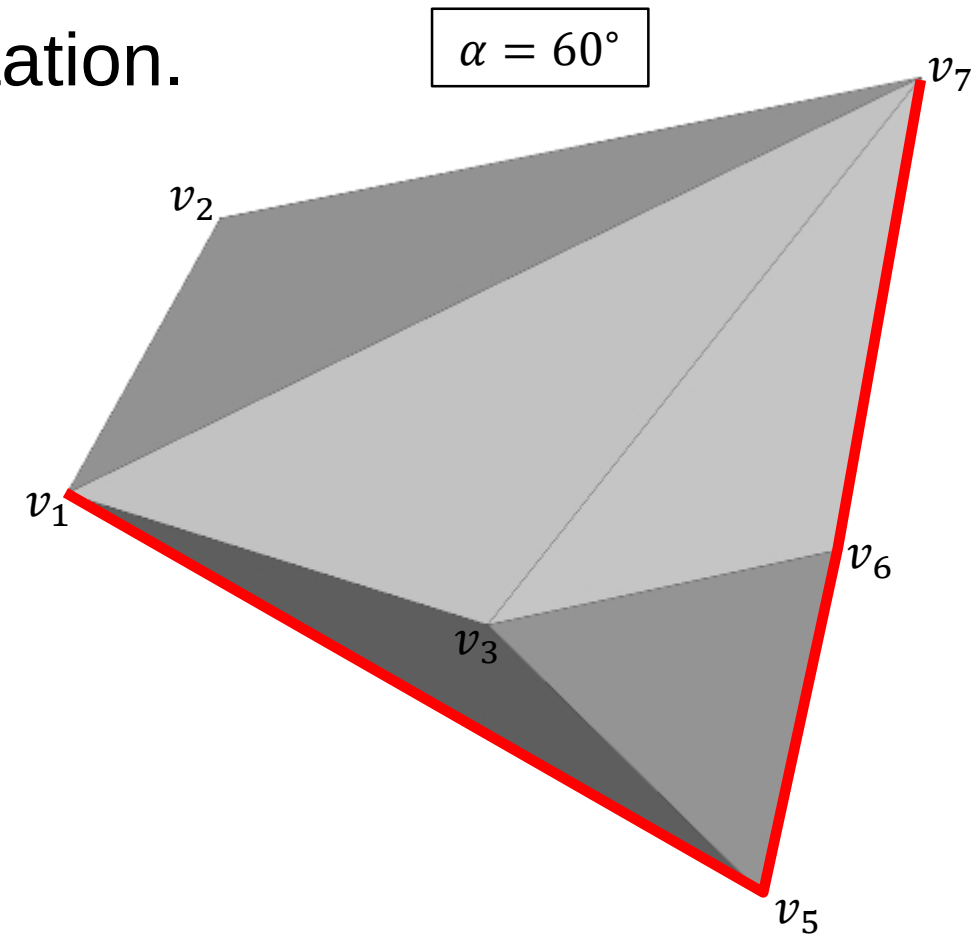
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$$E = \left\{ \begin{array}{c} (I, 15^\circ, v_3 v_5 v_6) \\ (D, 39^\circ, v_1 v_3 v_5) \\ \boxed{(D, 75^\circ, v_5 v_6 v_7)} \\ (I, 90^\circ, v_1 v_4 v_5) \\ (I, 130^\circ, v_1 v_2 v_4) \\ (D, 145^\circ, v_4 v_5 v_7) \end{array} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$



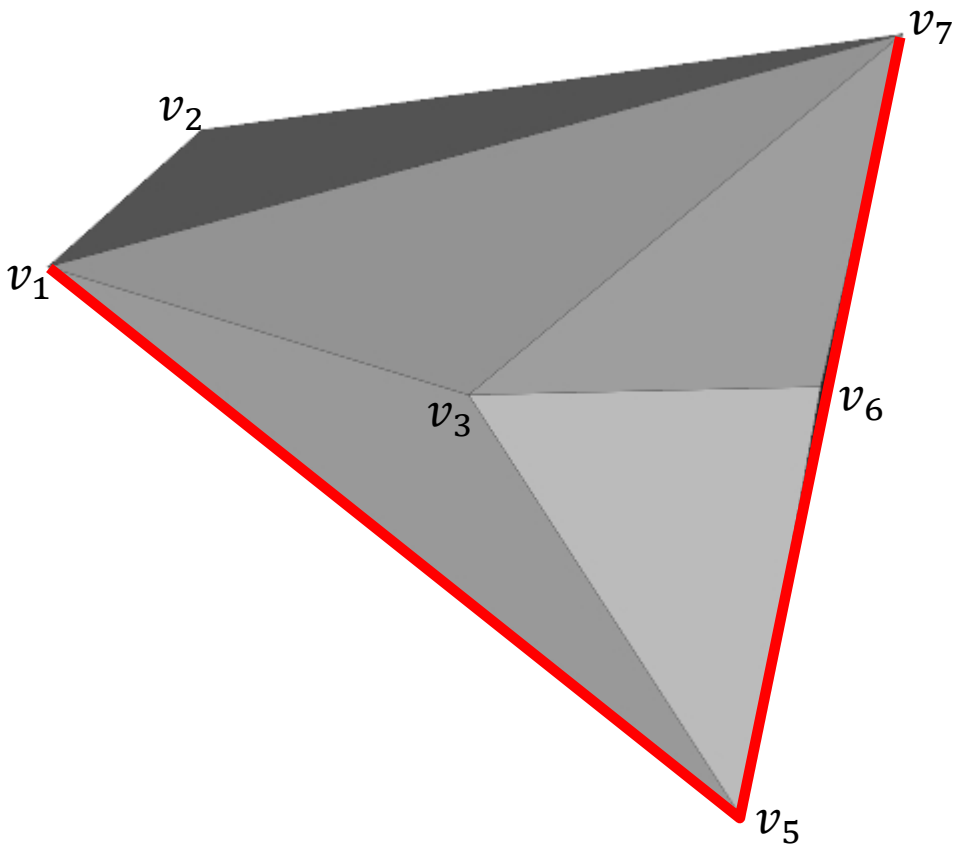
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$$E = \left\{ \begin{array}{c} (I, 15^\circ, v_3 v_5 v_6) \\ (D, 39^\circ, v_1 v_3 v_5) \\ \boxed{(D, 75^\circ, v_5 v_6 v_7)} \\ (I, 90^\circ, v_1 v_4 v_5) \\ (I, 130^\circ, v_1 v_2 v_4) \\ (D, 145^\circ, v_4 v_5 v_7) \end{array} \right\}$$

$\alpha = 80^\circ$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

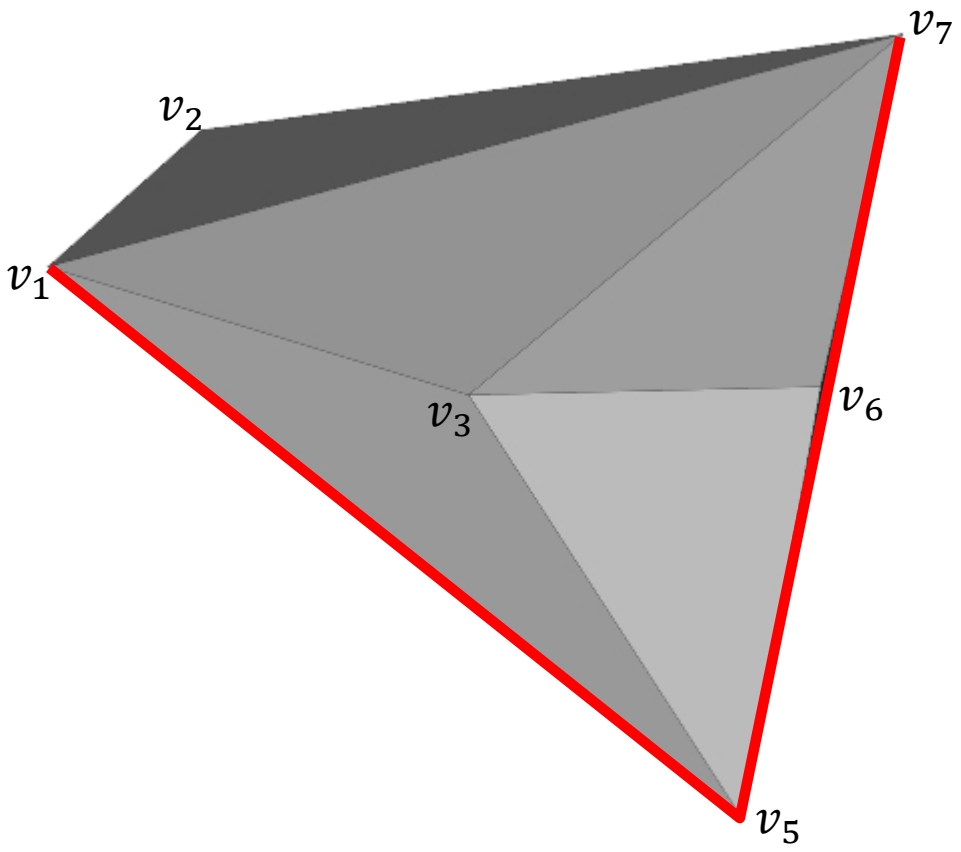
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 80°$

$$E = \left\{ \begin{array}{l} (I, 15°, v_3 v_5 v_6) \\ (D, 39°, v_1 v_3 v_5) \\ (D, 75°, v_5 v_6 v_7) \\ \boxed{(I, 90°, v_1 v_4 v_5)} \\ (I, 130°, v_1 v_2 v_4) \\ (D, 145°, v_4 v_5 v_7) \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

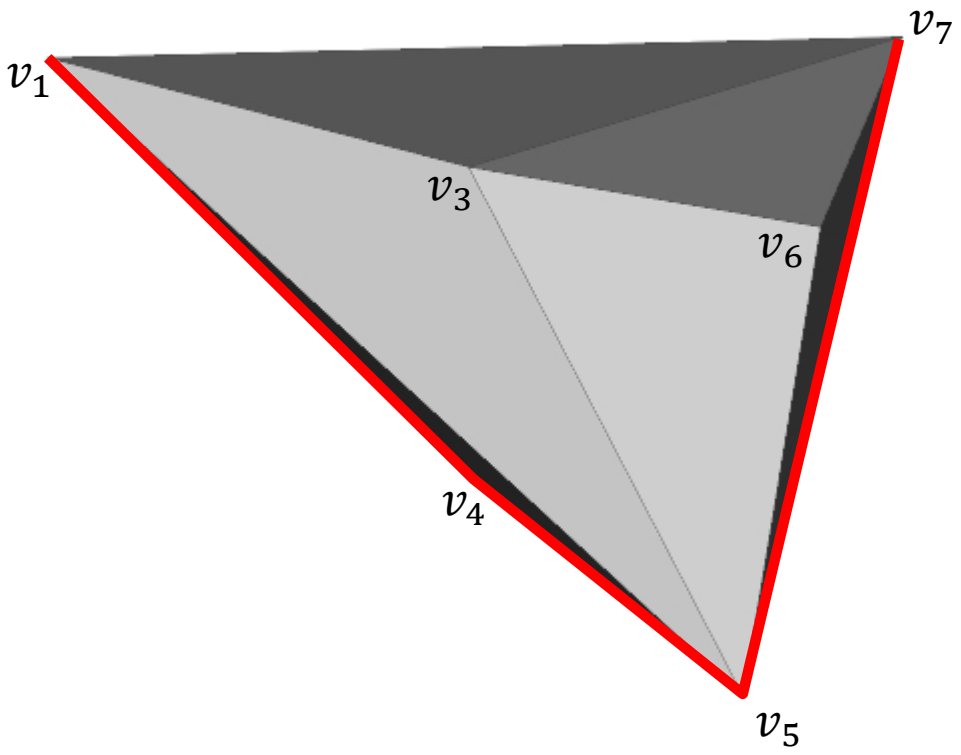
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 100°$

$$E = \left\{ \begin{array}{l} (I, 15°, v_3 v_5 v_6) \\ (D, 39°, v_1 v_3 v_5) \\ (D, 75°, v_5 v_6 v_7) \\ \boxed{(I, 90°, v_1 v_4 v_5)} \\ (I, 130°, v_1 v_2 v_4) \\ (D, 145°, v_4 v_5 v_7) \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

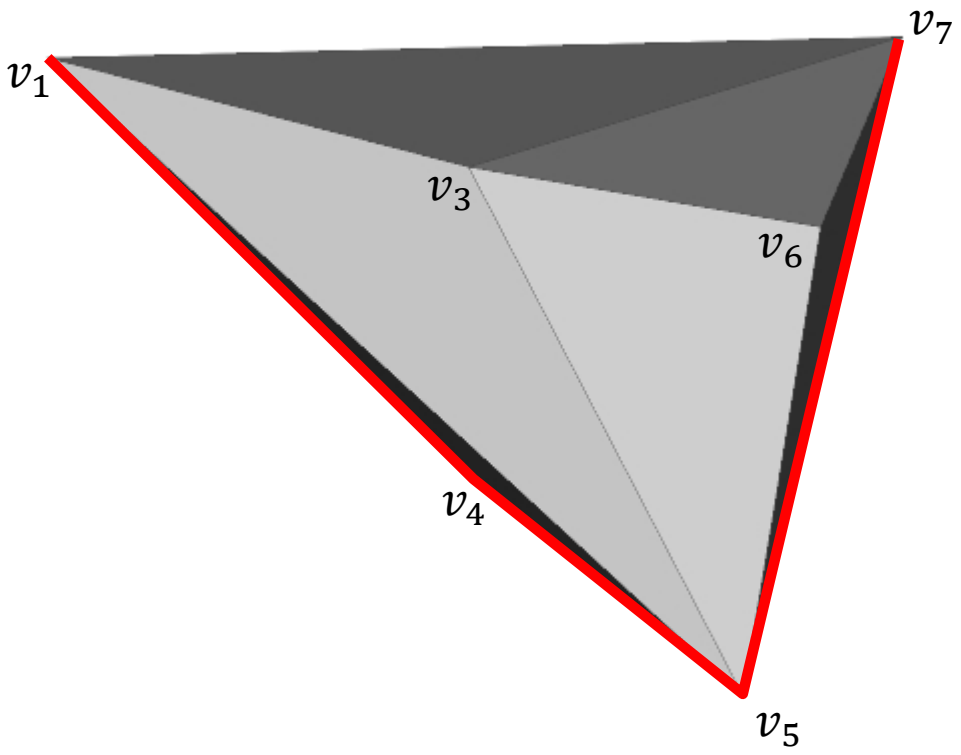
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 100°$

$$E = \left\{ \begin{array}{l} (I, 15°, v_3 v_5 v_6) \\ (D, 39°, v_1 v_3 v_5) \\ (D, 75°, v_5 v_6 v_7) \\ (I, 90°, v_1 v_4 v_5) \\ \boxed{(I, 130°, v_1 v_2 v_4)} \\ (D, 145°, v_4 v_5 v_7) \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

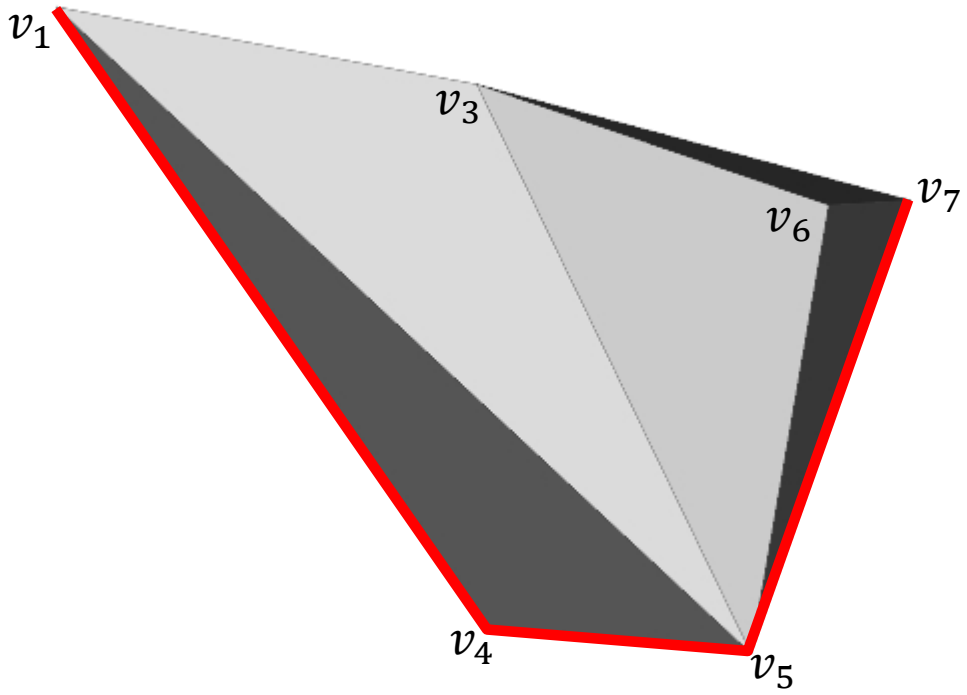
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 120°$

$$E = \left\{ \begin{array}{c} (I, 15°, v_3 v_5 v_6) \\ (D, 39°, v_1 v_3 v_5) \\ (D, 75°, v_5 v_6 v_7) \\ (I, 90°, v_1 v_4 v_5) \\ \boxed{(I, 130°, v_1 v_2 v_4)} \\ (D, 145°, v_4 v_5 v_7) \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

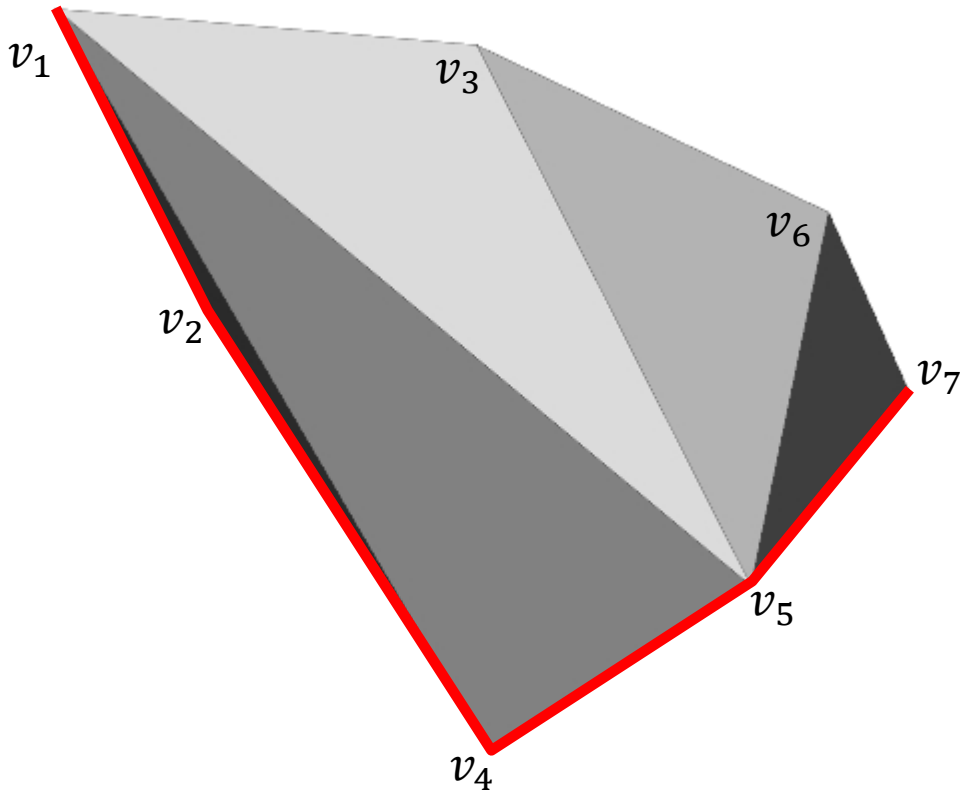
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 140°$

$$E = \left\{ \begin{array}{l} (I, 15°, v_3v_5v_6) \\ (D, 39°, v_1v_3v_5) \\ (D, 75°, v_5v_6v_7) \\ (I, 90°, v_1v_4v_5) \\ \boxed{(I, 130°, v_1v_2v_4)} \\ (D, 145°, v_4v_5v_7) \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

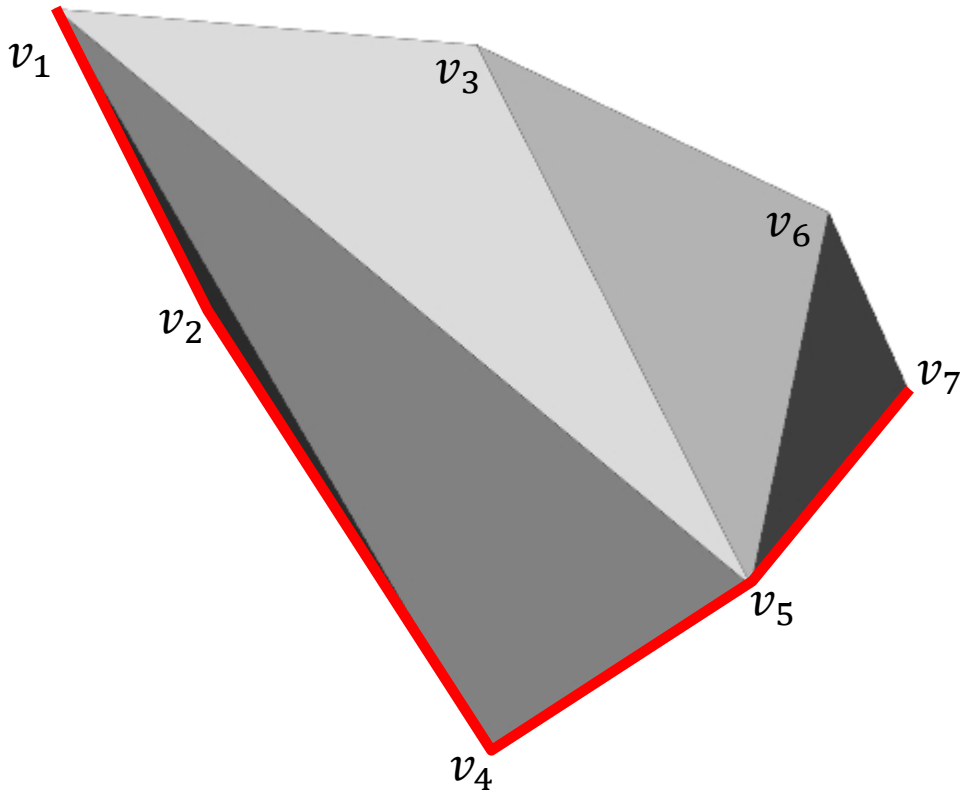
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 140°$

$$E = \left\{ \begin{array}{c} (I, 15°, v_3v_5v_6) \\ (D, 39°, v_1v_3v_5) \\ (D, 75°, v_5v_6v_7) \\ (I, 90°, v_1v_4v_5) \\ (I, 130°, v_1v_2v_4) \\ \boxed{(D, 145°, v_4v_5v_7)} \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

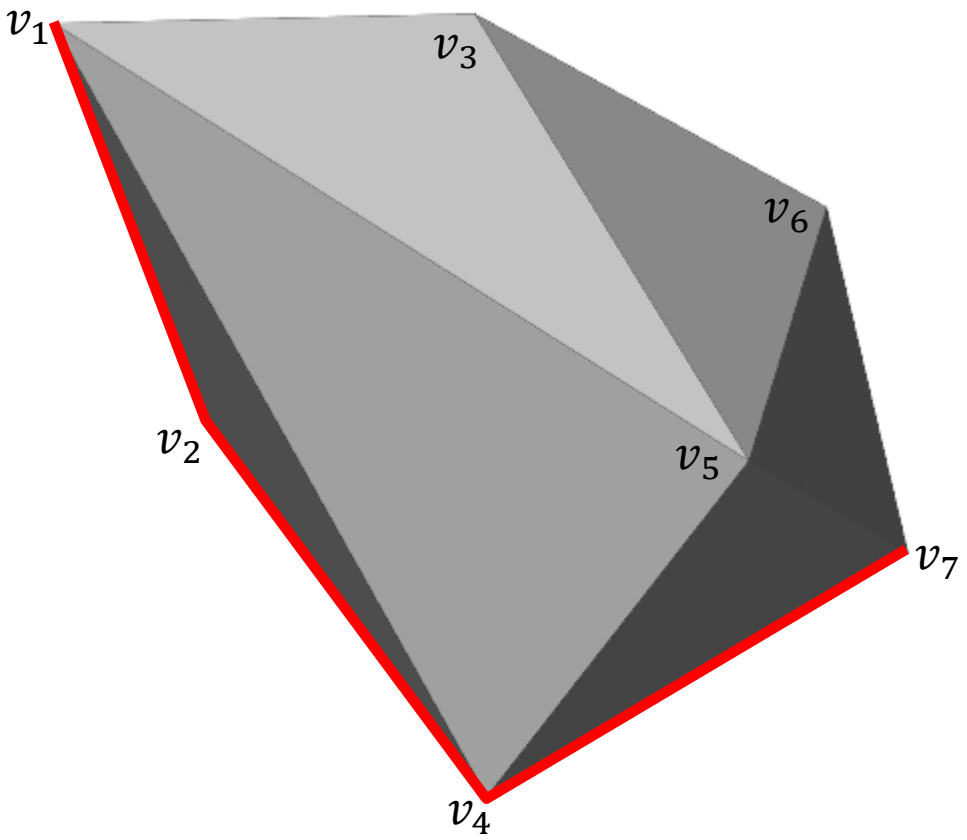
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 160°$

$$E = \left\{ \begin{array}{c} (I, 15°, v_3v_5v_6) \\ (D, 39°, v_1v_3v_5) \\ (D, 75°, v_5v_6v_7) \\ (I, 90°, v_1v_4v_5) \\ (I, 130°, v_1v_2v_4) \\ \boxed{(D, 145°, v_4v_5v_7)} \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

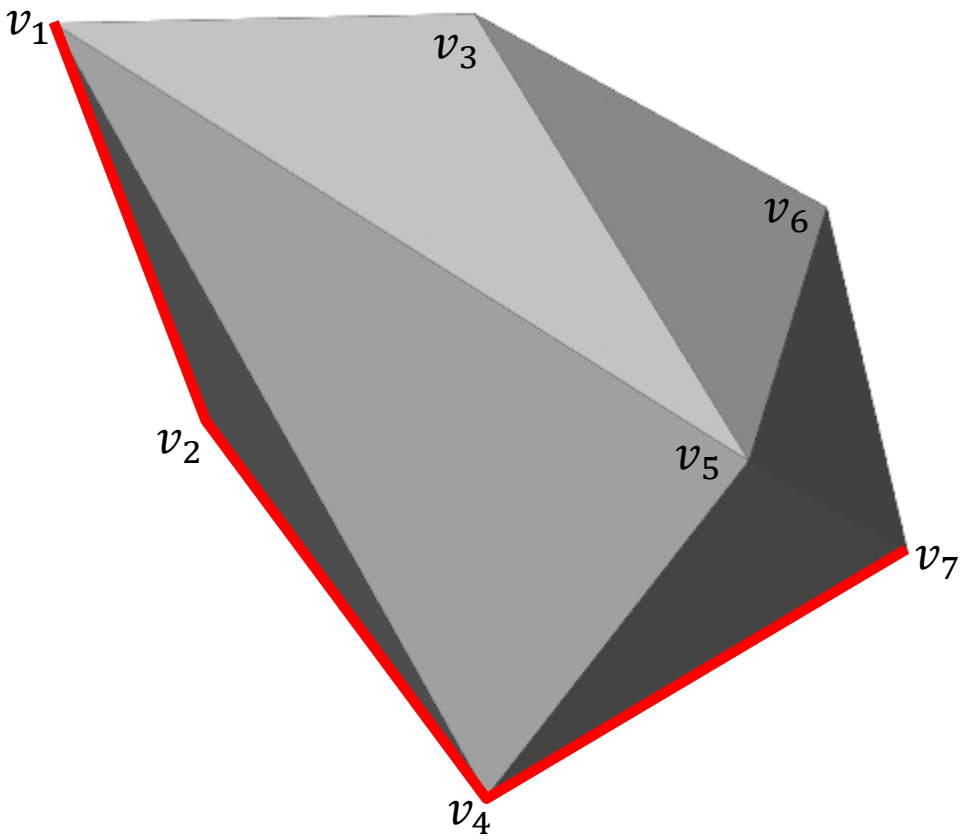
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 160°$

$$E = \left\{ \begin{array}{c} (I, 15°, v_3 v_5 v_6) \\ (D, 39°, v_1 v_3 v_5) \\ (D, 75°, v_5 v_6 v_7) \\ (I, 90°, v_1 v_4 v_5) \\ (I, 130°, v_1 v_2 v_4) \\ (D, 145°, v_4 v_5 v_7) \end{array} \right\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

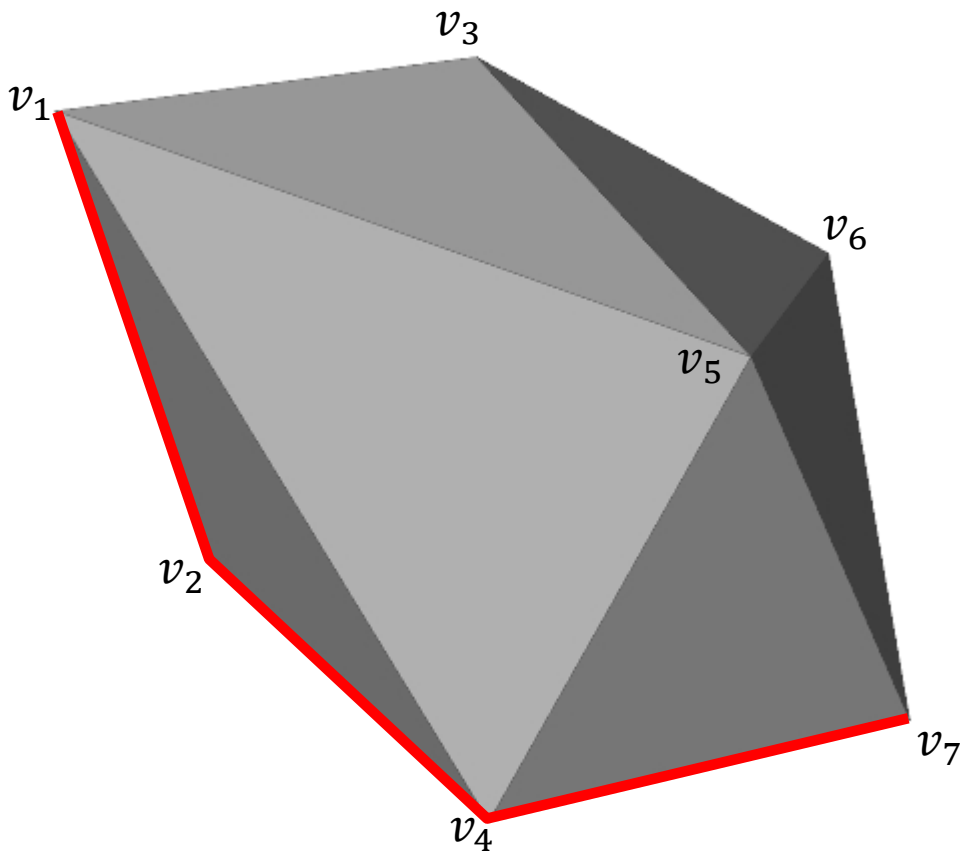
Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$\alpha = 180°$

$$E = \begin{Bmatrix} (I, 15°, v_3v_5v_6) \\ (D, 39°, v_1v_3v_5) \\ (D, 75°, v_5v_6v_7) \\ (I, 90°, v_1v_4v_5) \\ (I, 130°, v_1v_2v_4) \\ (D, 145°, v_4v_5v_7) \end{Bmatrix}$$



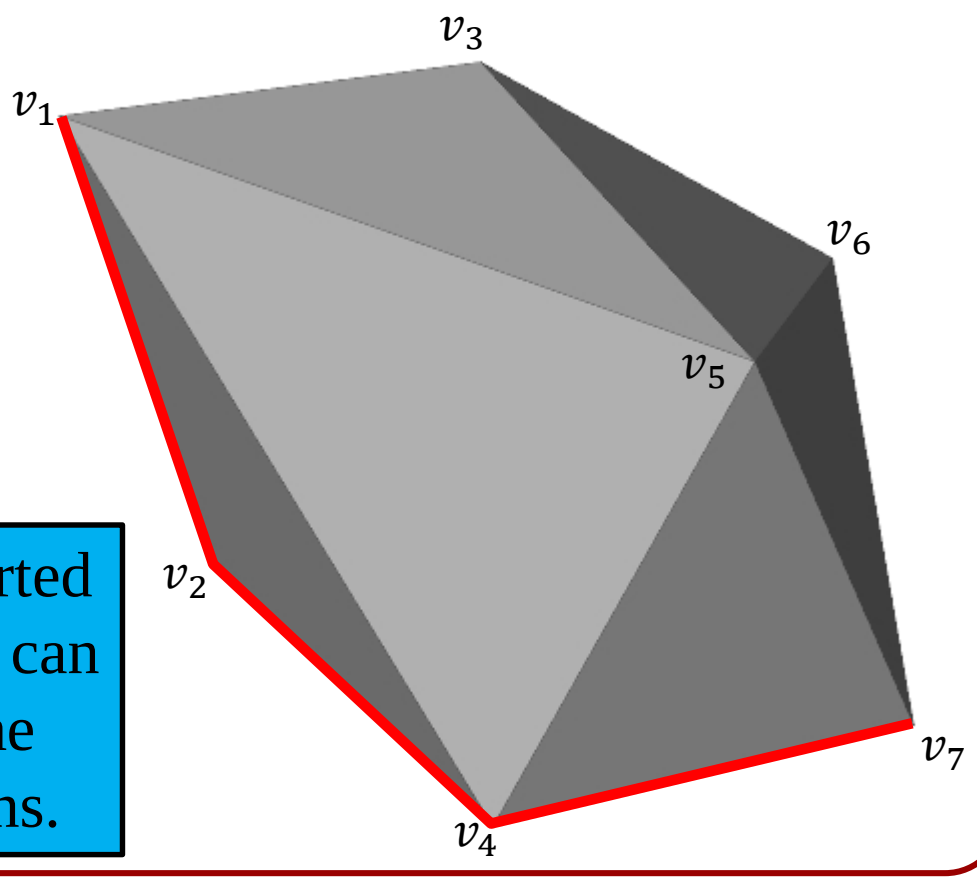
$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

Divide-and-Conquer [Chan, 2003]

Approach:

Use a *kinetic* representation.

$$E = \left\{ \begin{array}{c} (I, 15^\circ, v_3 v_5 v_6) \\ (D, 39^\circ, v_1 v_3 v_5) \\ (D, 75^\circ, v_5 v_6 v_7) \\ (I, 90^\circ, v_1 v_4 v_5) \\ (I, 130^\circ, v_1 v_2 v_4) \\ (D, 145^\circ, v_4 v_5 v_7) \end{array} \right\}$$



Since a vertex can only be inserted and deleted once the lower hull can be reconstructed in linear time from the kinetic representations.

Divide-and-Conquer [Chan, 2003]

Merge:

Use a *kinetic* representation, tracking the lower tangent that joins the hulls.

Key Idea:

As we rotate the points, the merged hull changes:

- The individual hulls change
- The lower tangent no longer supports the neighbors of its endpoints from below.

Divide-and-Conquer [Chan, 2003]

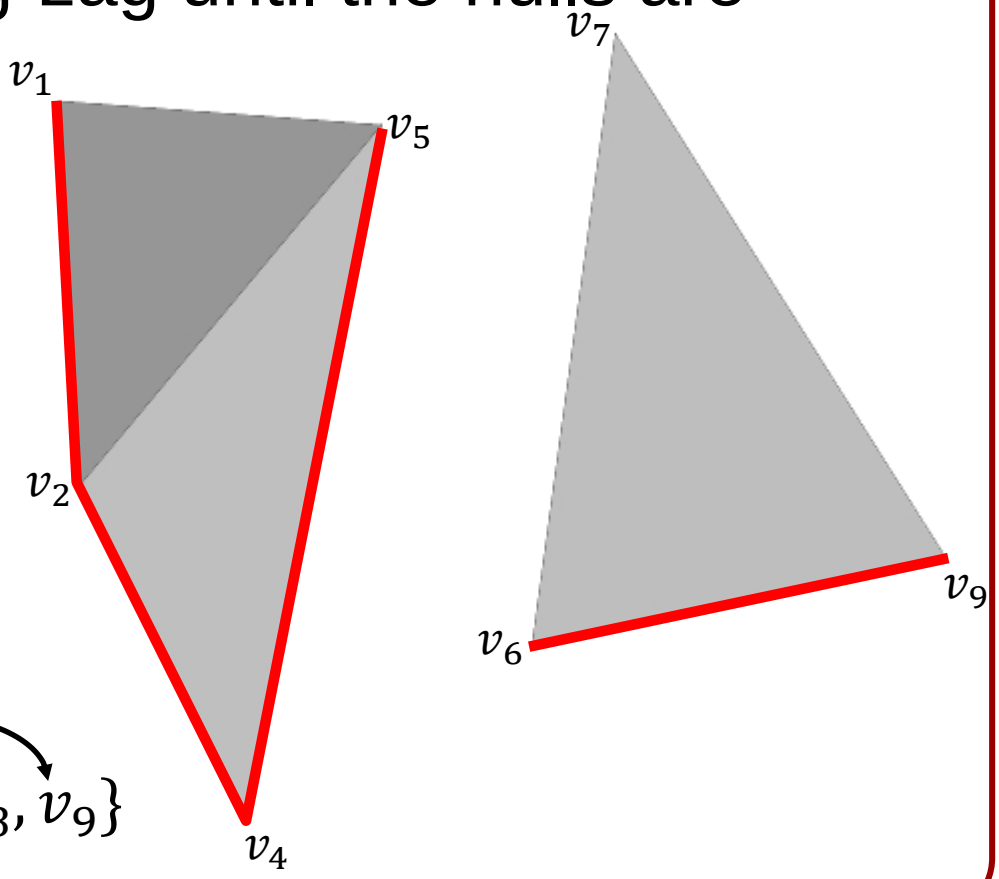
Merge:

- Initialize the lower tangent.
- Compute the minimum of the angles:
  - For the next left hull event
  - For the next right hull event
  - When the lower tangent stops supporting:
    - » the left neighbor of the left end-point
    - » the right neighbor of the left end-point
    - » the left neighbor of the right end-point
    - » the right neighbor of the right end-point
- Update the event list for the merged hull

Divide-and-Conquer [Chan, 2003]

Initialize the Lower Tangent:

- As in the 2D case, zig-zag until the hulls are supported.

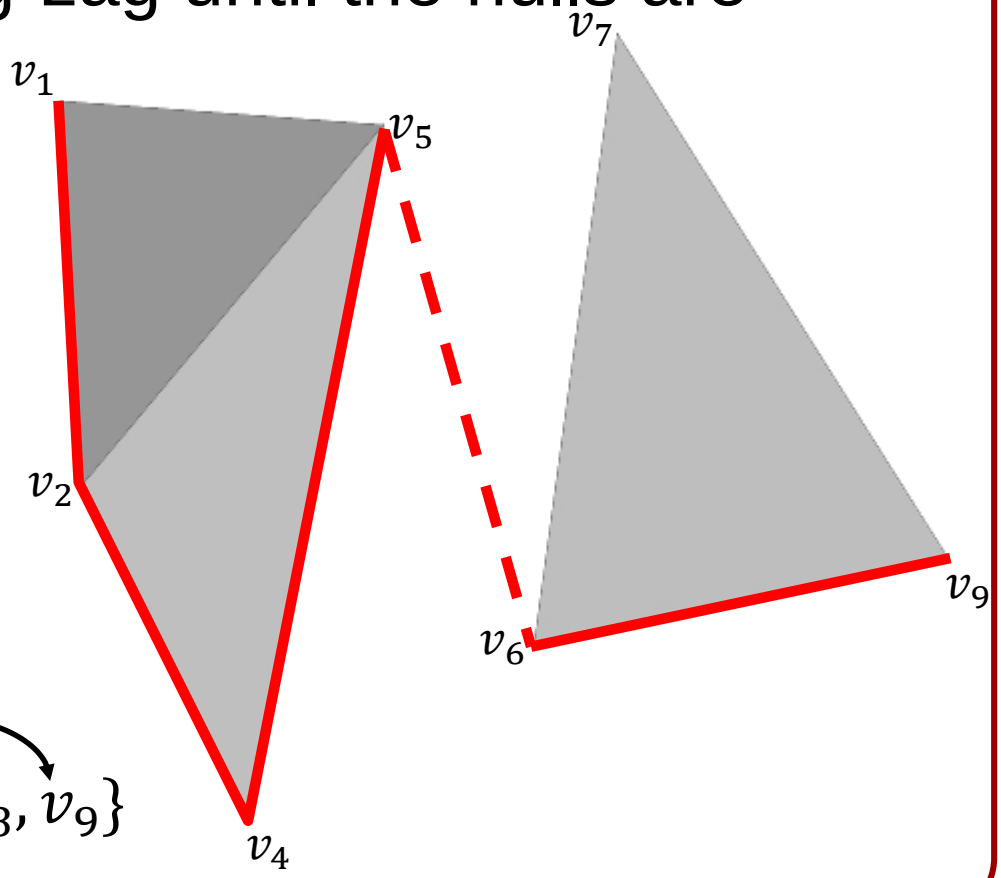


$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

Divide-and-Conquer [Chan, 2003]

Initialize the Lower Tangent:

- As in the 2D case, zig-zag until the hulls are supported.

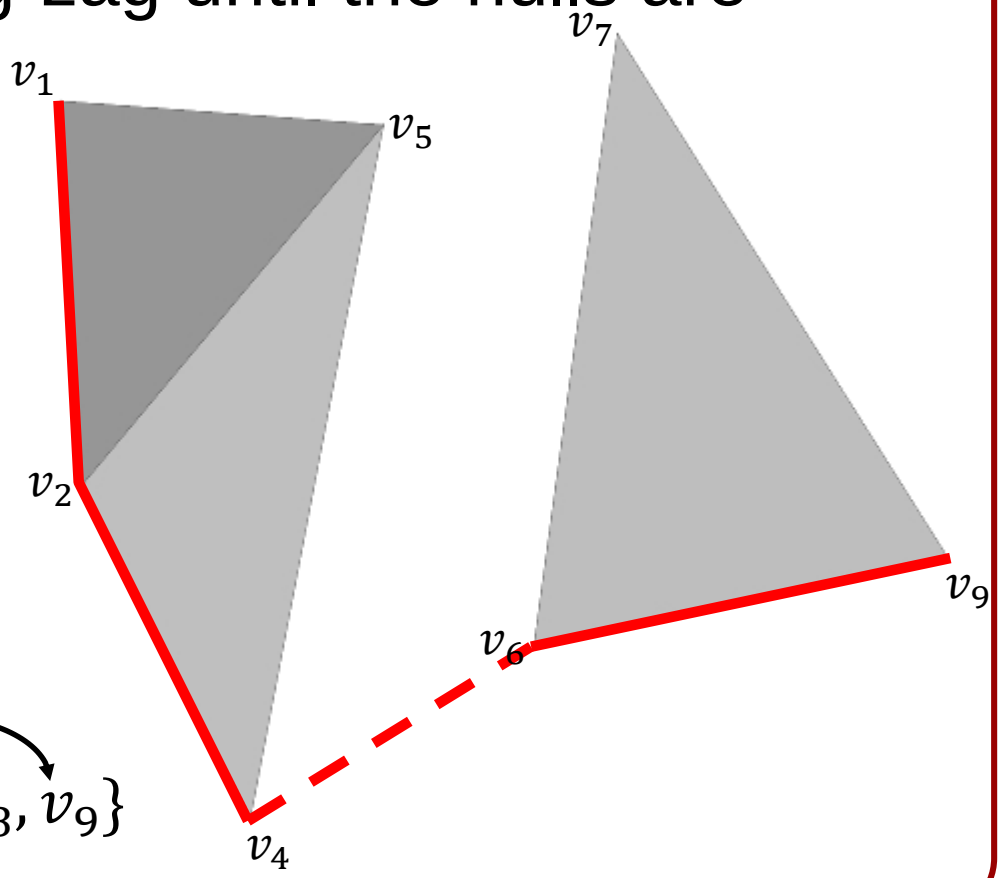


$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

Divide-and-Conquer [Chan, 2003]

Initialize the Lower Tangent:

- As in the 2D case, zig-zag until the hulls are supported.

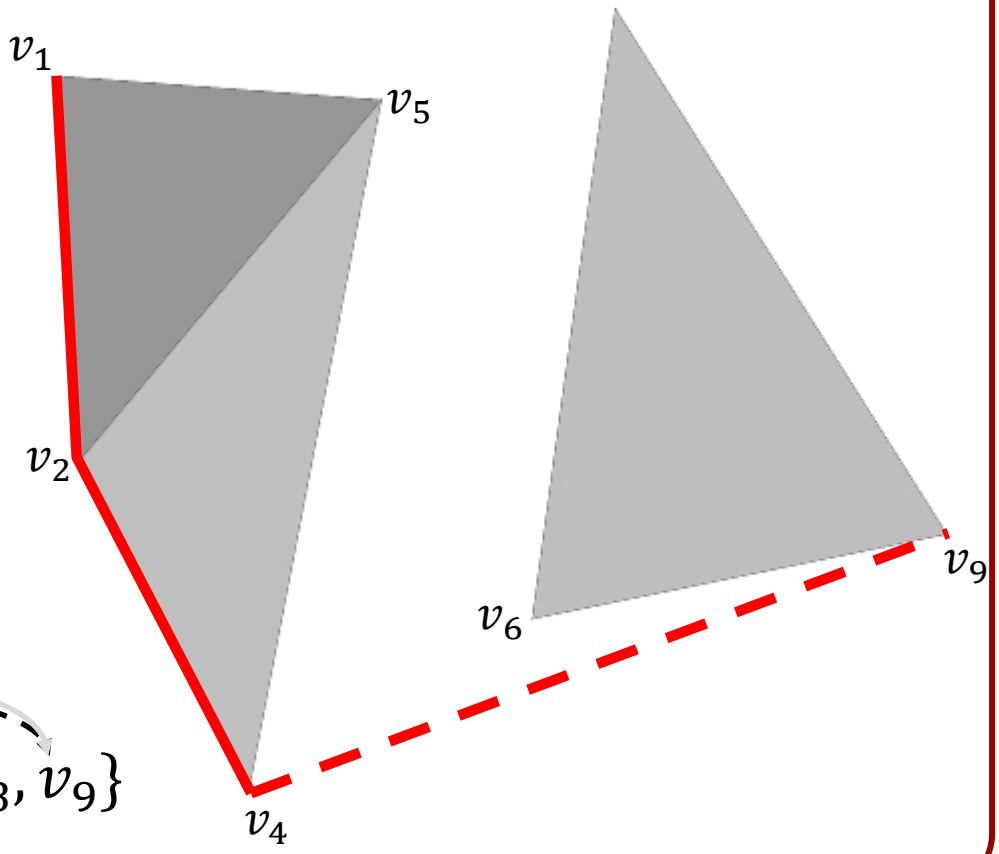


$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

Divide-and-Conquer [Chan, 2003]

Initialize the Lower Tangent:

- As in the 2D case, zig-zag until the hulls are supported.



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

Divide-and-Conquer [Chan, 2003]

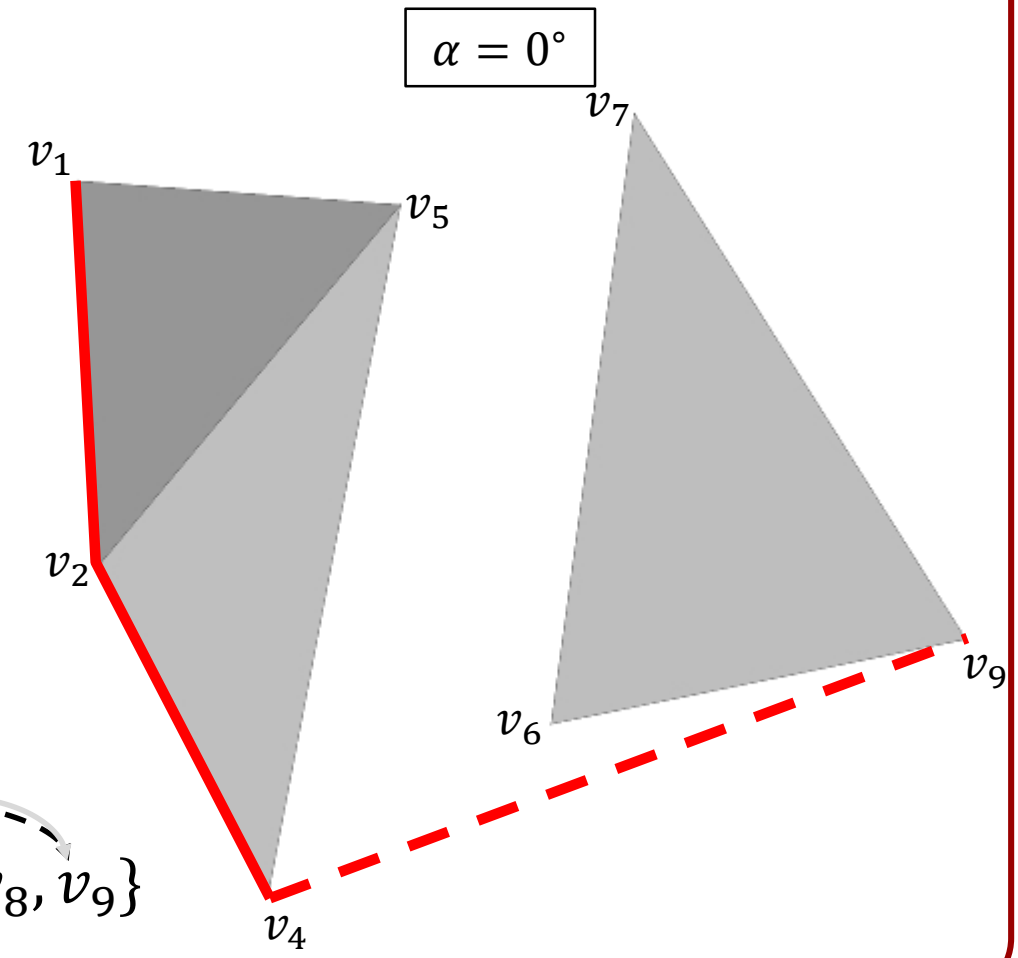
Advance to the next event:

$$E_L = \begin{Bmatrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{Bmatrix}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (45°, 55^0)$$

$$R = (\boxed{10°} \quad \cdot \quad)$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

Divide-and-Conquer [Chan, 2003]

Advance to the next event:

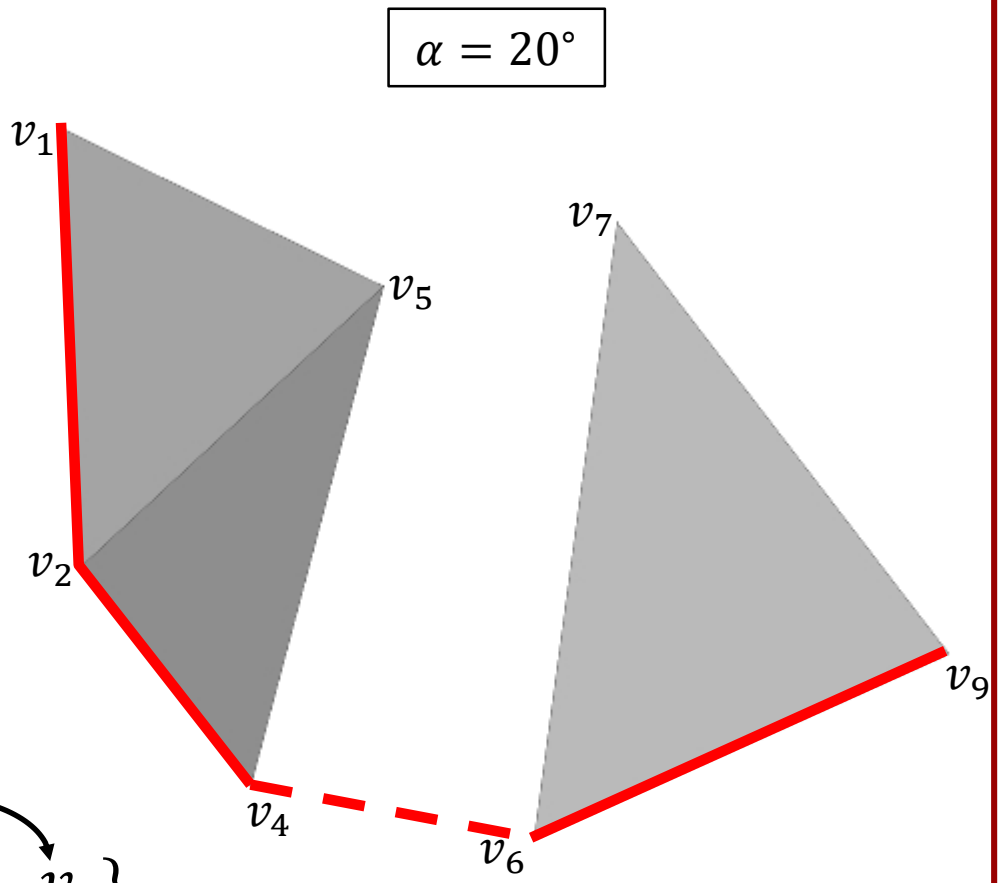
$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (45°, 55^0)$$

$$R = (\boxed{10°} \quad \cdot \quad)$$

$$E = \{(I, 10°, v_4, v_6, v_9)\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

Divide-and-Conquer [Chan, 2003]

Advance to the next event:

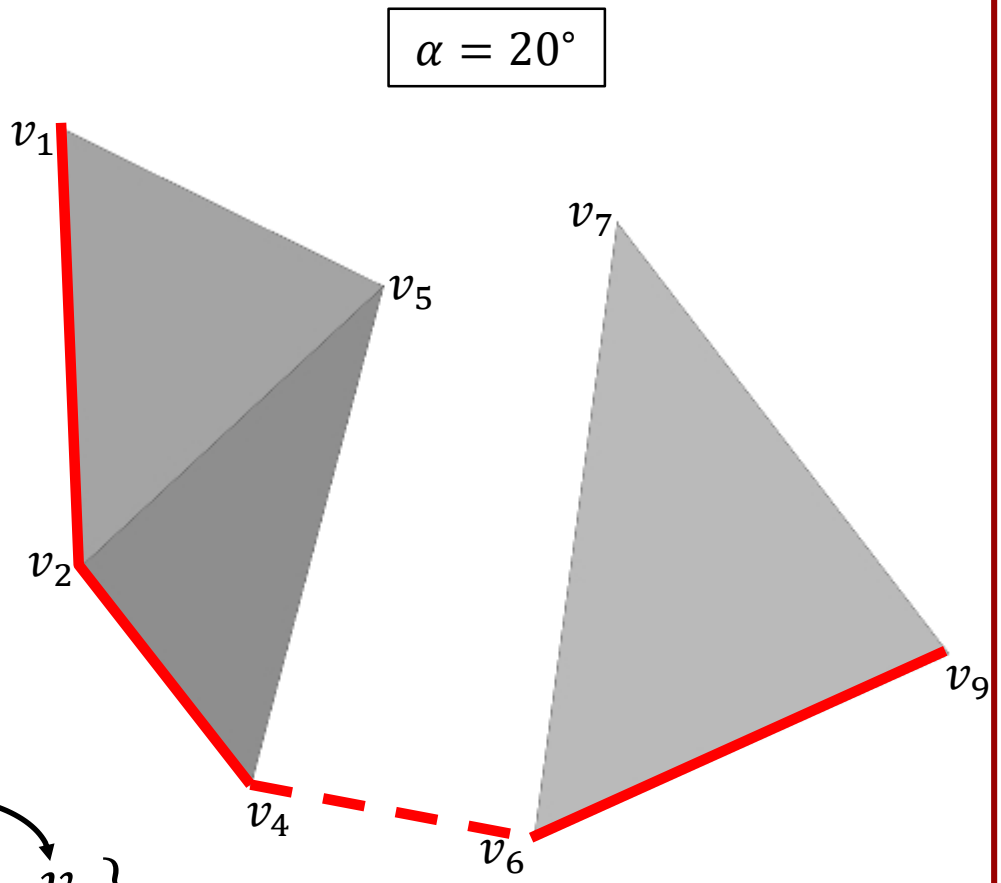
$$E_L = \begin{Bmatrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{Bmatrix}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (35°, 70°)$$

$$R = (\ \cdot\ , 170°)$$

$$E = \{(I, 10°, v_4, v_6, v_9)\}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \begin{Bmatrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{Bmatrix}$$

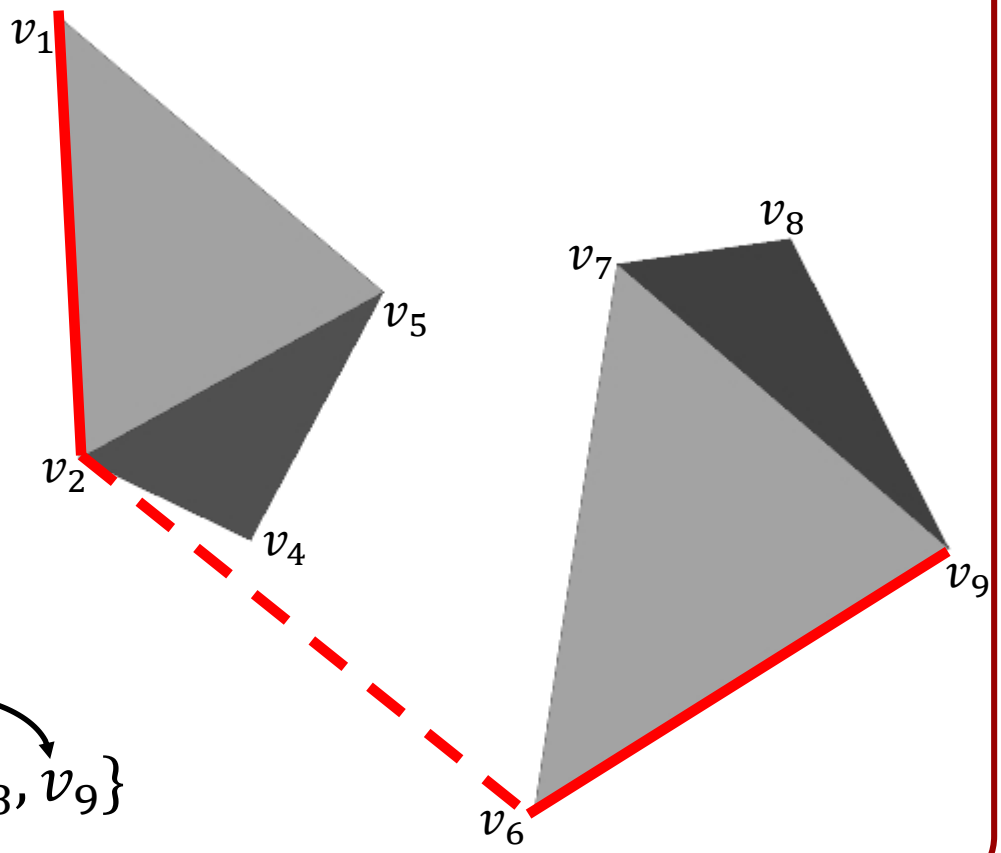
$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (35°, 70°)$$
$$R = (110°, \cdot)$$

$$E = \begin{Bmatrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \end{Bmatrix}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

$\alpha = 40°$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

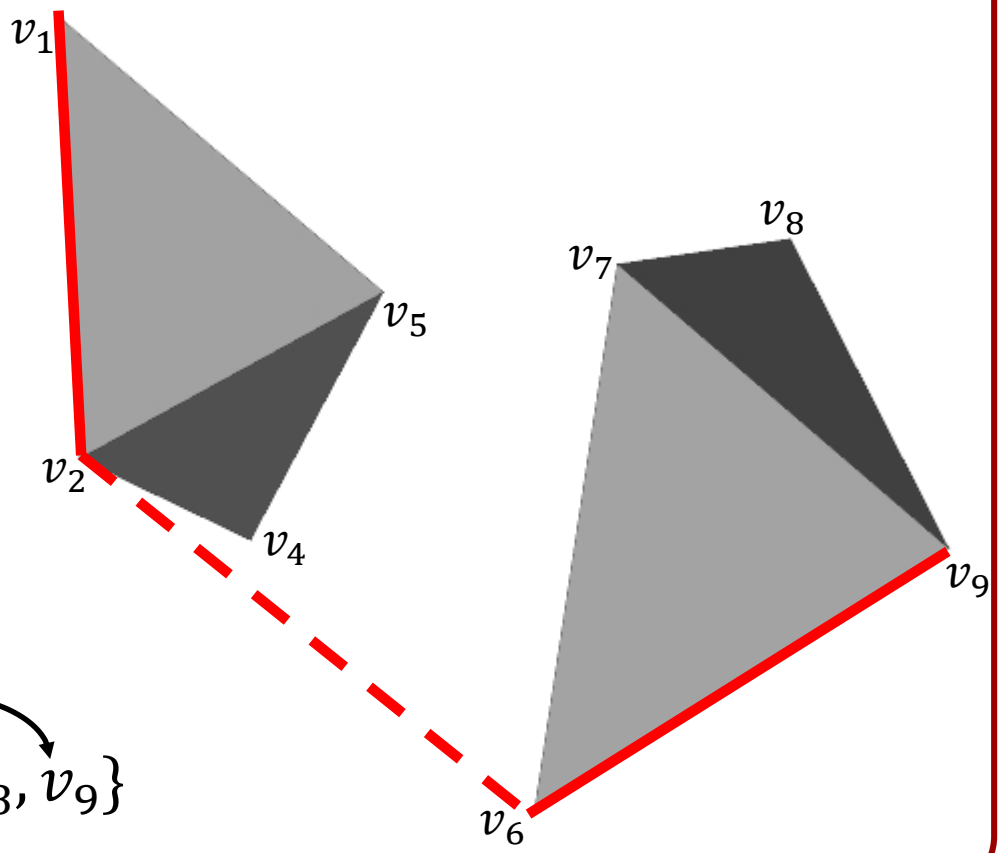
$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (120°, 100°)$$
$$R = (\ \cdot\ , 150°)$$

$$E = \left\{ \begin{matrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

$\alpha = 40°$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

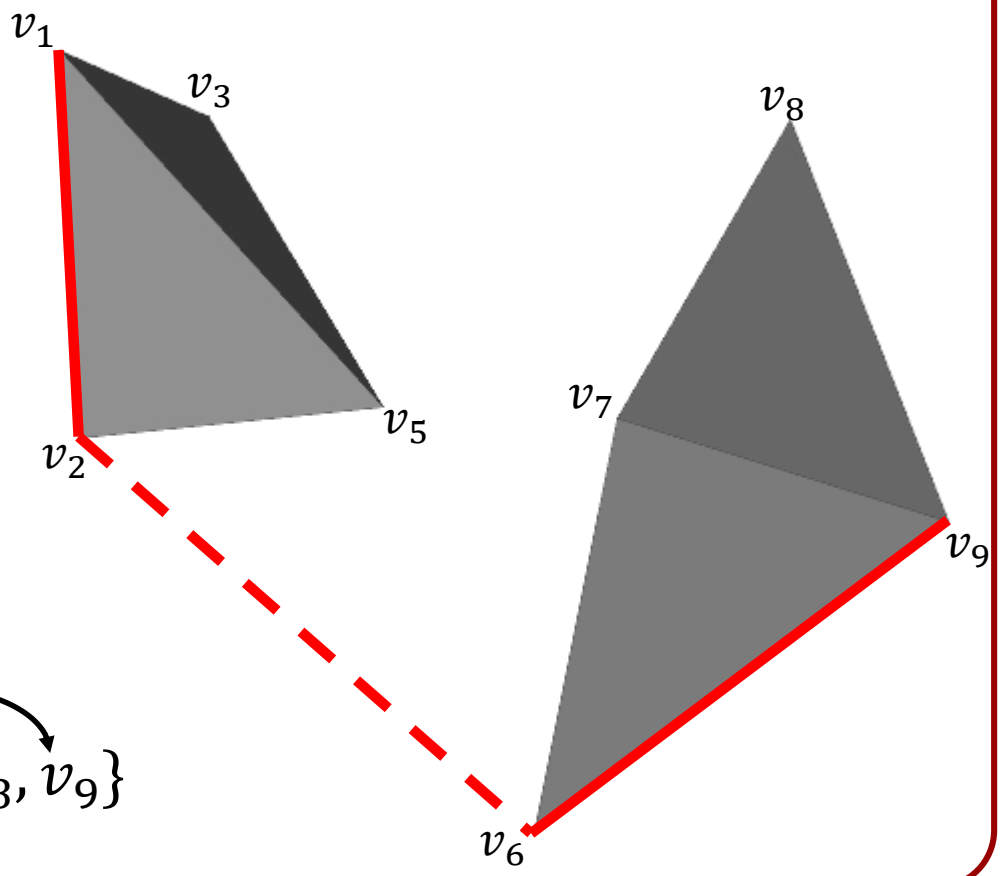
$$L = (120°, 100°)$$

$$R = (\ \cdot\ , 150°)$$

$$E = \left\{ \begin{matrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

$\alpha = 60°$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

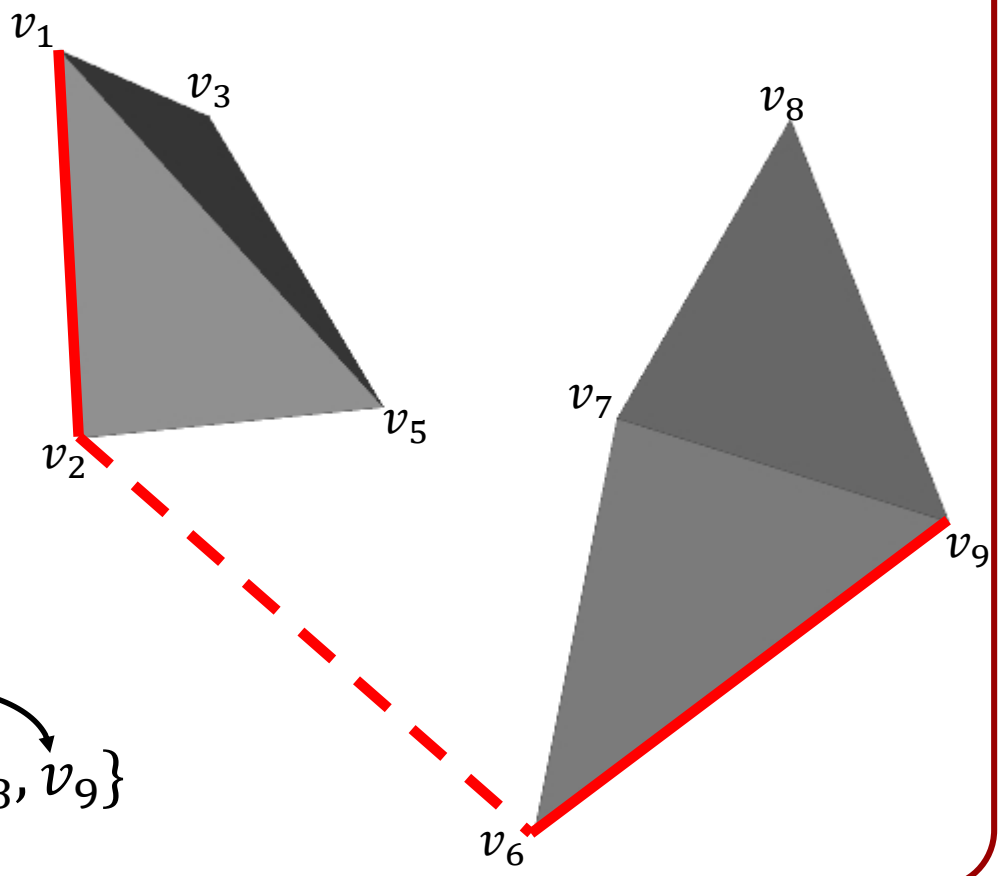
$$L = (120°, 100°)$$

$$R = (\ \cdot\ , 150°)$$

$$E = \left\{ \begin{matrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

$\alpha = 60°$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

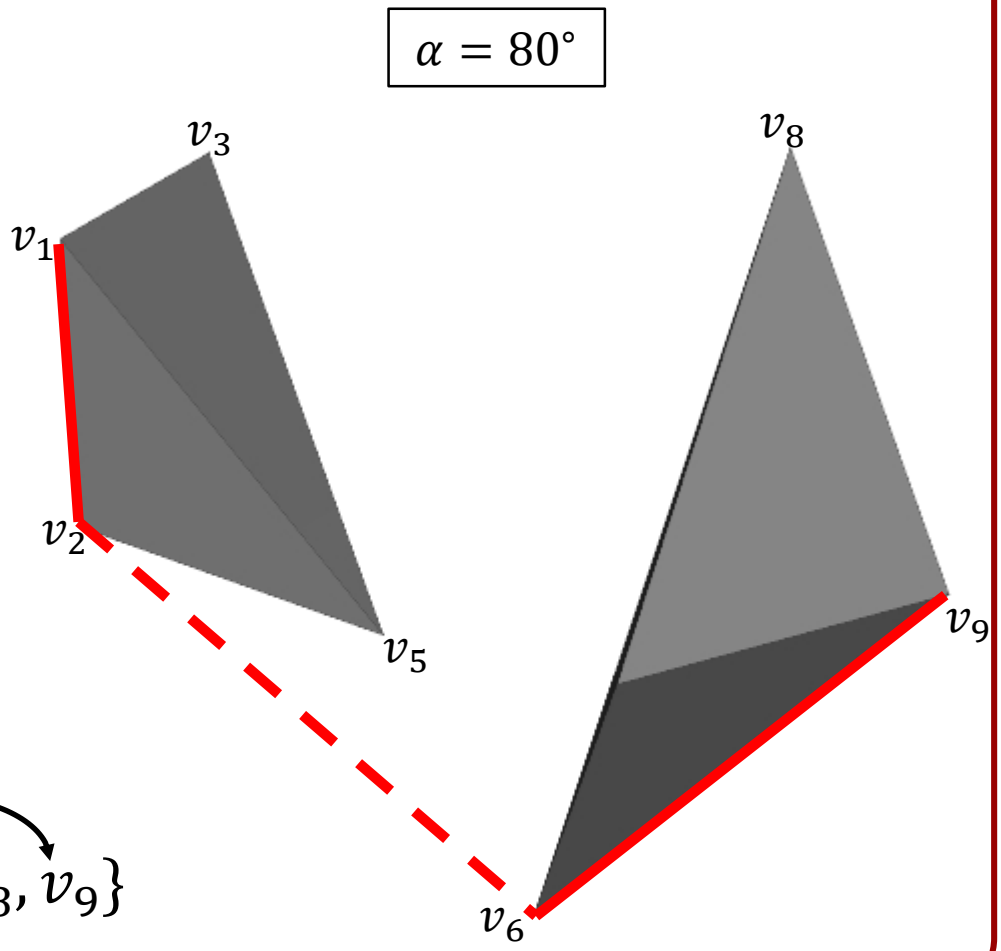
$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (120°, 100°)$$

$$R = (\ \cdot\ , 150°)$$

$$E = \left\{ \begin{matrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

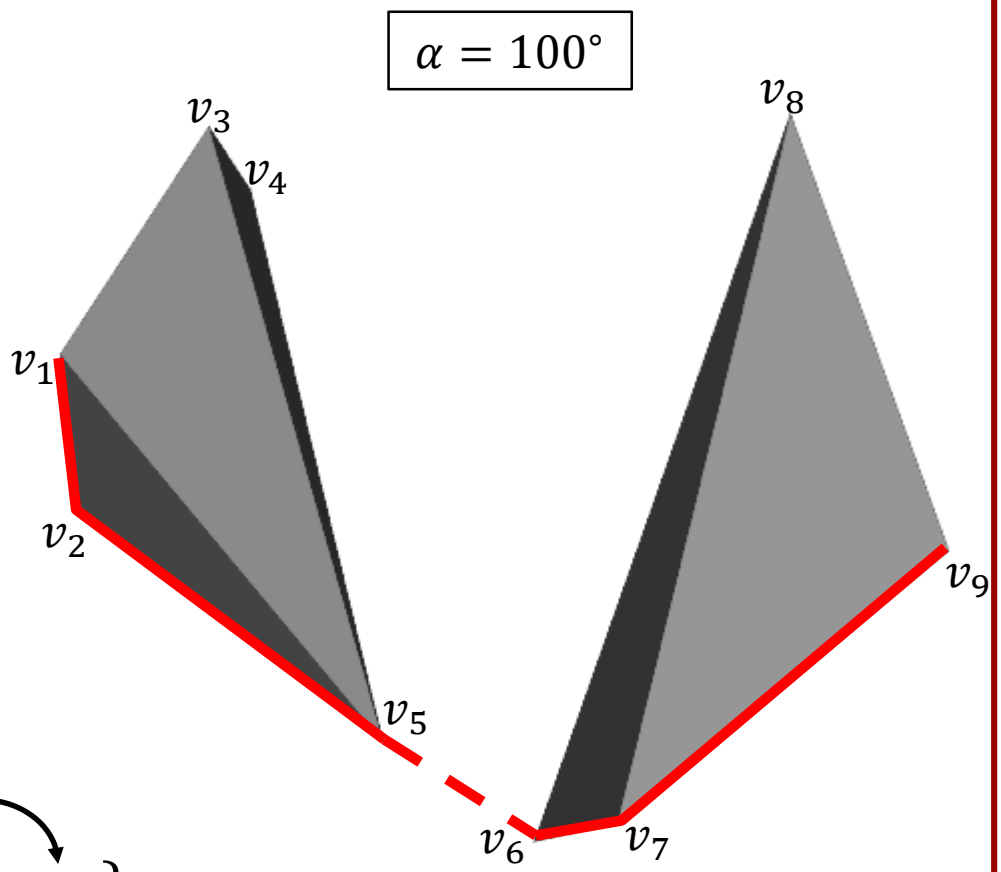
$$E_L = \begin{Bmatrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{Bmatrix}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (120°, 100°)$$

$$R = (\ \cdot\ , 150°)$$

$$E = \begin{Bmatrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \end{Bmatrix}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

Divide-and-Conquer [Chan, 2003]

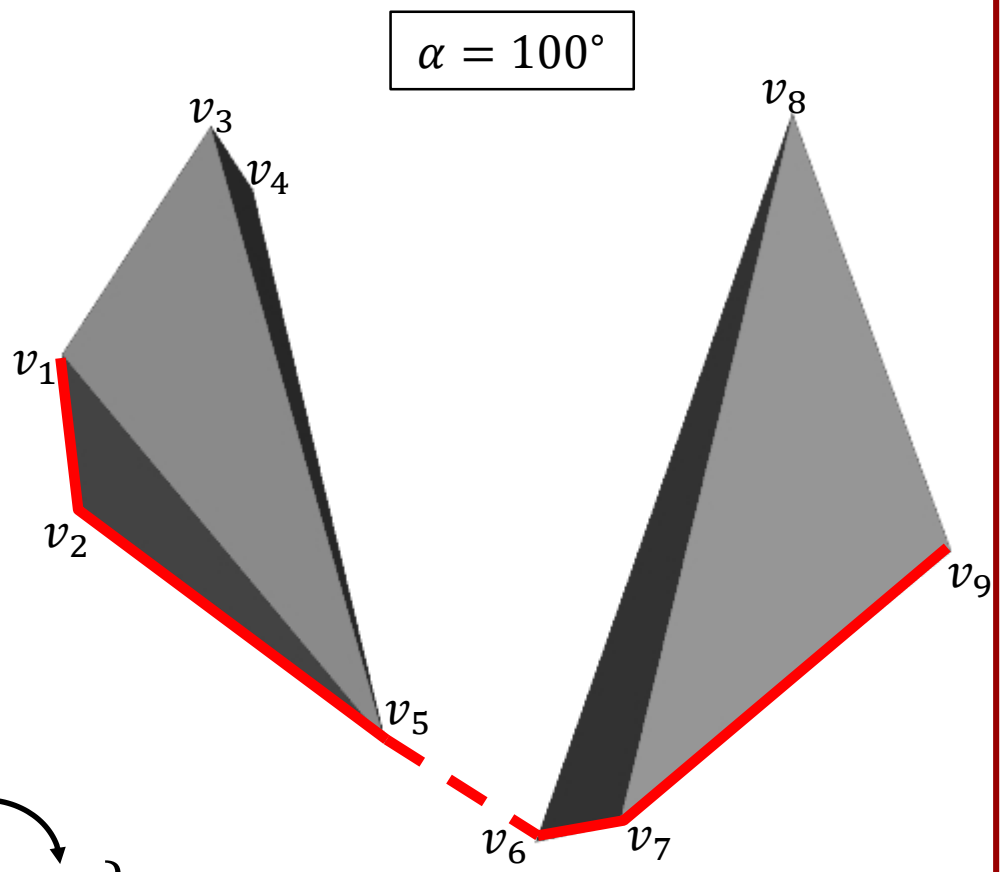
Advance to the next event:

$$E_L = \begin{Bmatrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{Bmatrix}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (120°, \boxed{100°})$$
$$R = (\ \cdot\ , 150°)$$

$$E = \begin{Bmatrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \end{Bmatrix}$$



$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

Divide-and-Conquer [Chan, 2003]

Advance to the next event:

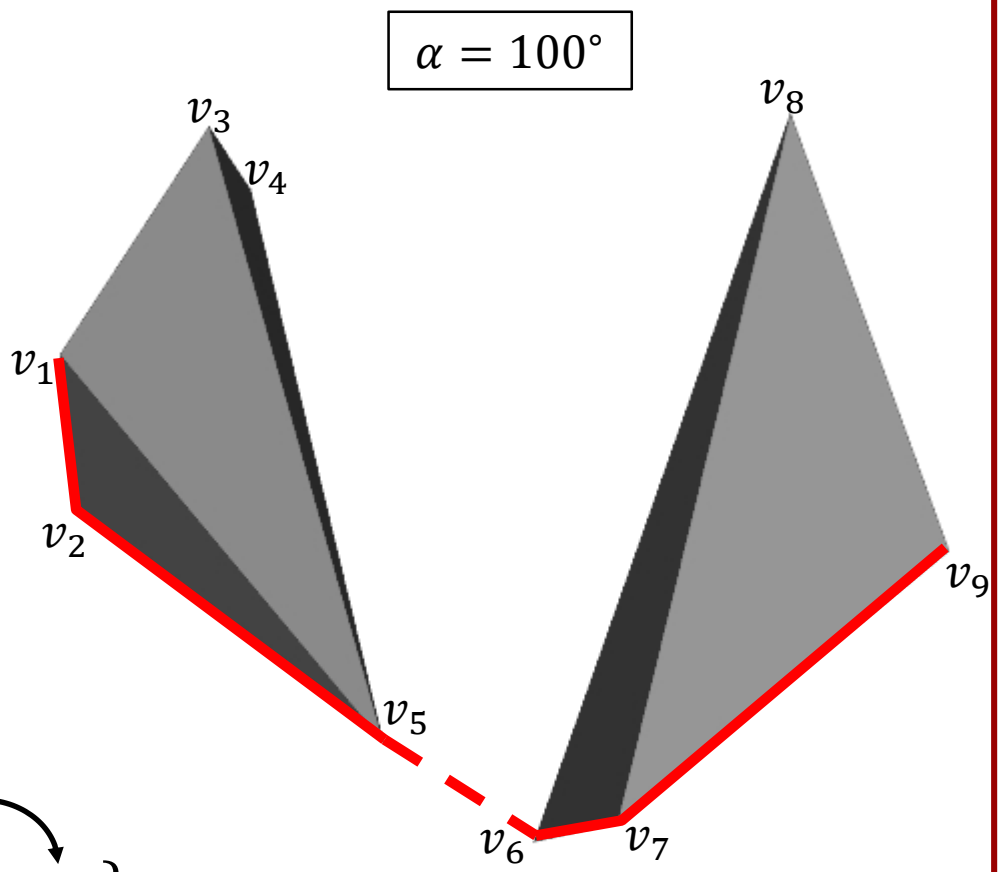
$$E_L = \begin{Bmatrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{Bmatrix}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (120°, \boxed{100°})$$
$$R = (\ \cdot\ , 150°)$$

$$E = \begin{Bmatrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \\ (I, 100°, v_2v_5v_6) \end{Bmatrix}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

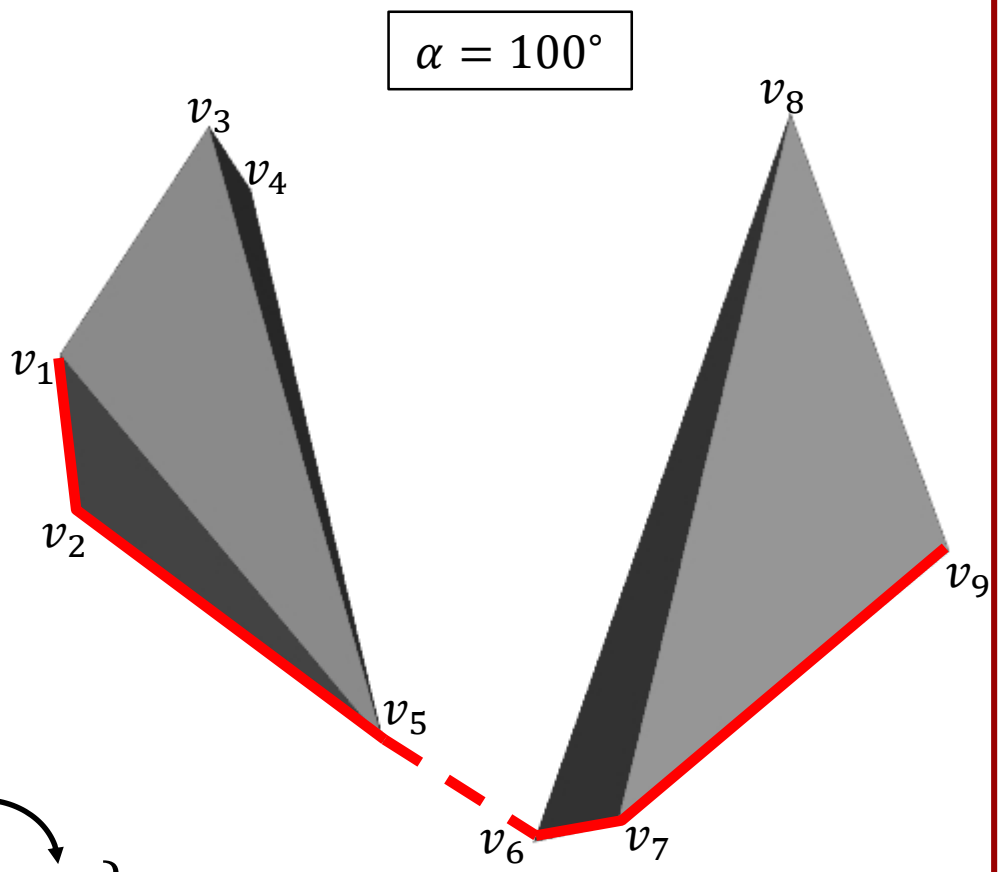
$$E_L = \begin{Bmatrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{Bmatrix}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (180°, \ \cdot \ )$$
$$R = (\ \cdot \ , \boxed{105°})$$

$$E = \begin{Bmatrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \\ (I, 100°, v_2v_5v_6) \end{Bmatrix}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

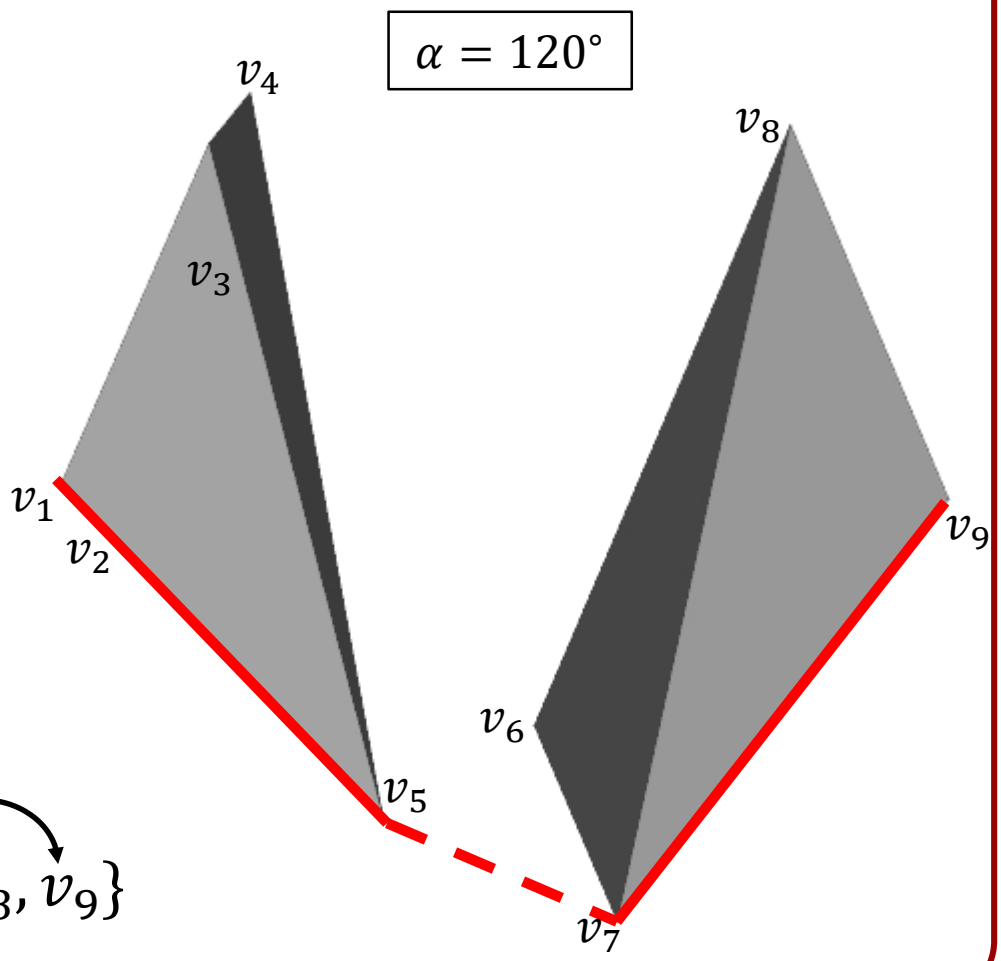
$$E_L = \begin{Bmatrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{Bmatrix}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (180°, \ \cdot \ )$$
$$R = (\ \cdot \ , \boxed{105°})$$

$$E = \begin{Bmatrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \\ (I, 100°, v_2v_5v_6) \\ (D, 105°, v_5v_6v_7) \end{Bmatrix}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

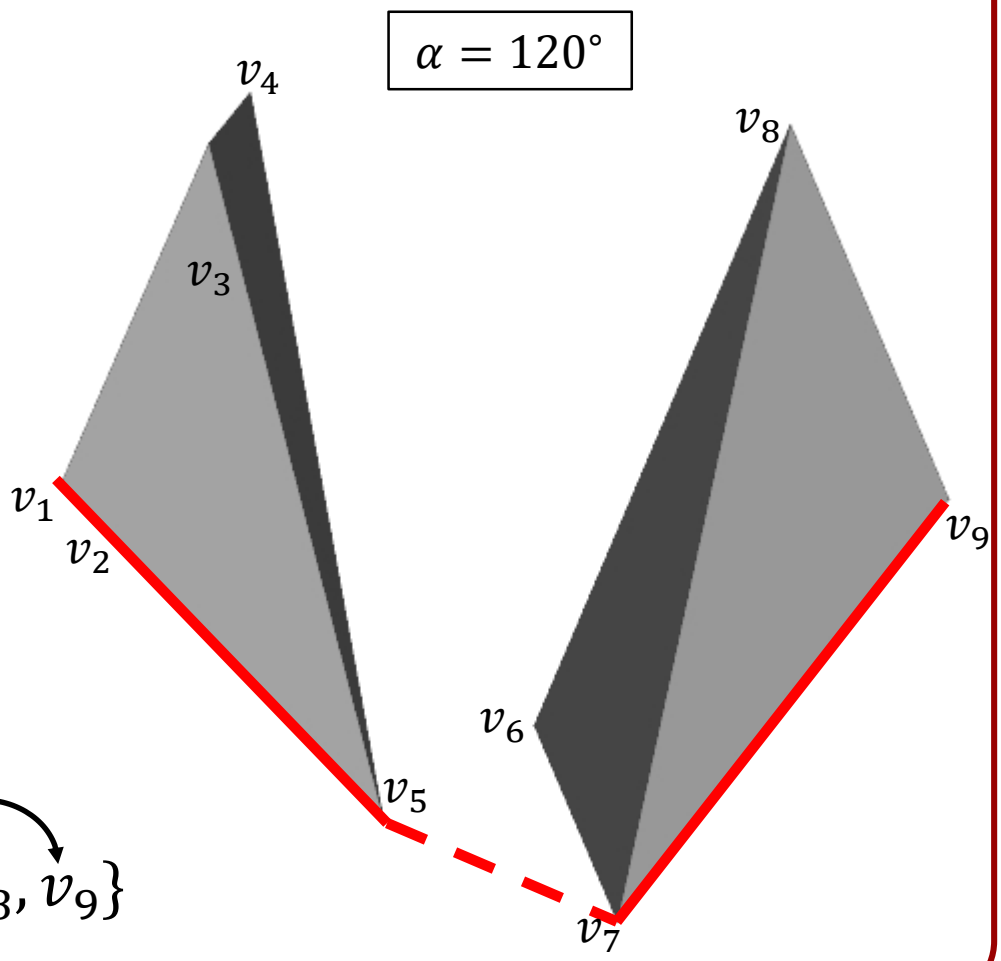
$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ \boxed{(D, 110°, v_1v_2v_5)} \end{matrix} \right\}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (155°, \quad \cdot \quad)$$
$$R = (180°, 180°)$$

$$E = \left\{ \begin{matrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \\ (I, 100°, v_2v_5v_6) \\ (D, 105°, v_5v_6v_7) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

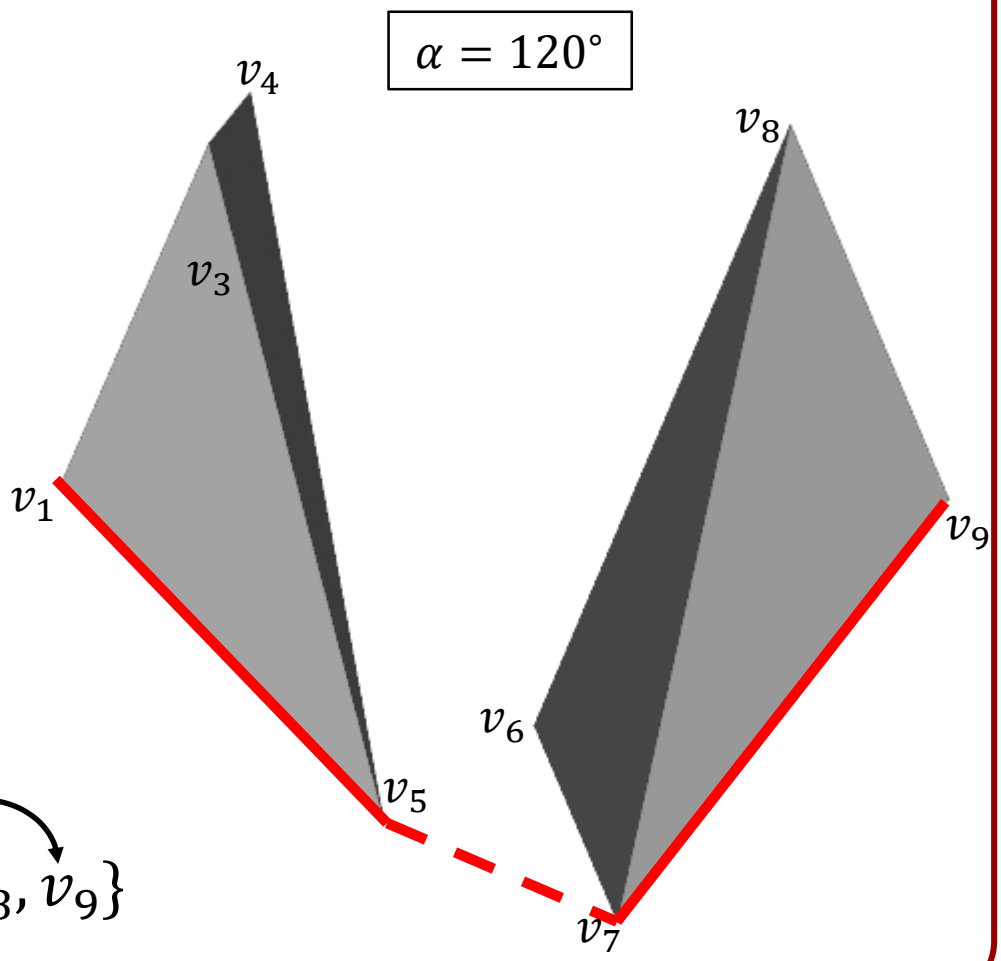
$$E_L = \{(D, 50°, v_2v_4v_5), (D, 110°, v_1v_2v_5)\}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (155°, \cdot)$$
$$R = (180°, 180°)$$

$$E = \{(I, 10°, v_4, v_6, v_9), (D, 35°, v_2v_4v_5), (I, 90°, v_6, v_7, v_9), (I, 100°, v_2v_5v_6), (D, 105°, v_5v_6v_7), (D, 110°, v_1v_2v_5)\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

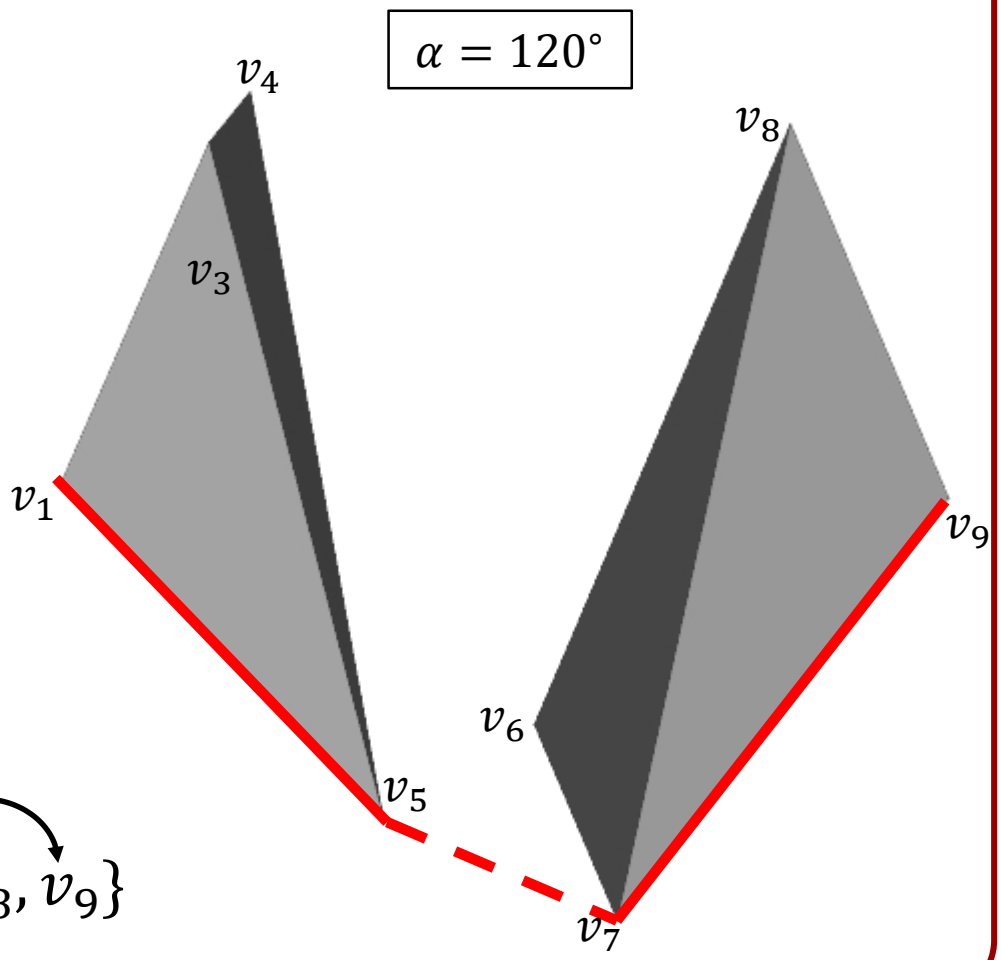
$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (\boxed{155°} \quad \cdot \quad)$$
$$R = (180°, 180°)$$

$$E = \left\{ \begin{matrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \\ (I, 100°, v_2v_5v_6) \\ (D, 105°, v_5v_6v_7) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

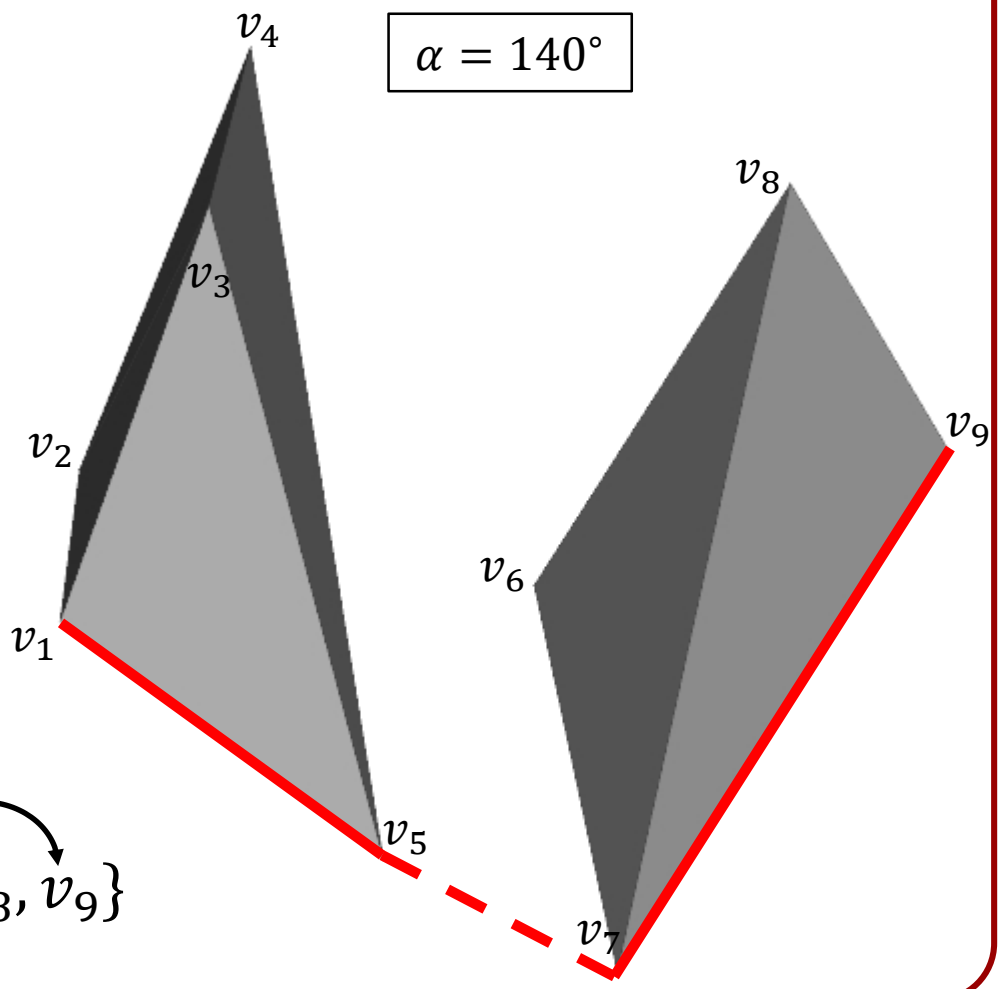
$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (\boxed{155°} \quad \cdot \quad)$$

$$R = (180°, 180°)$$

$$E = \left\{ \begin{matrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \\ (I, 100°, v_2v_5v_6) \\ (D, 105°, v_5v_6v_7) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

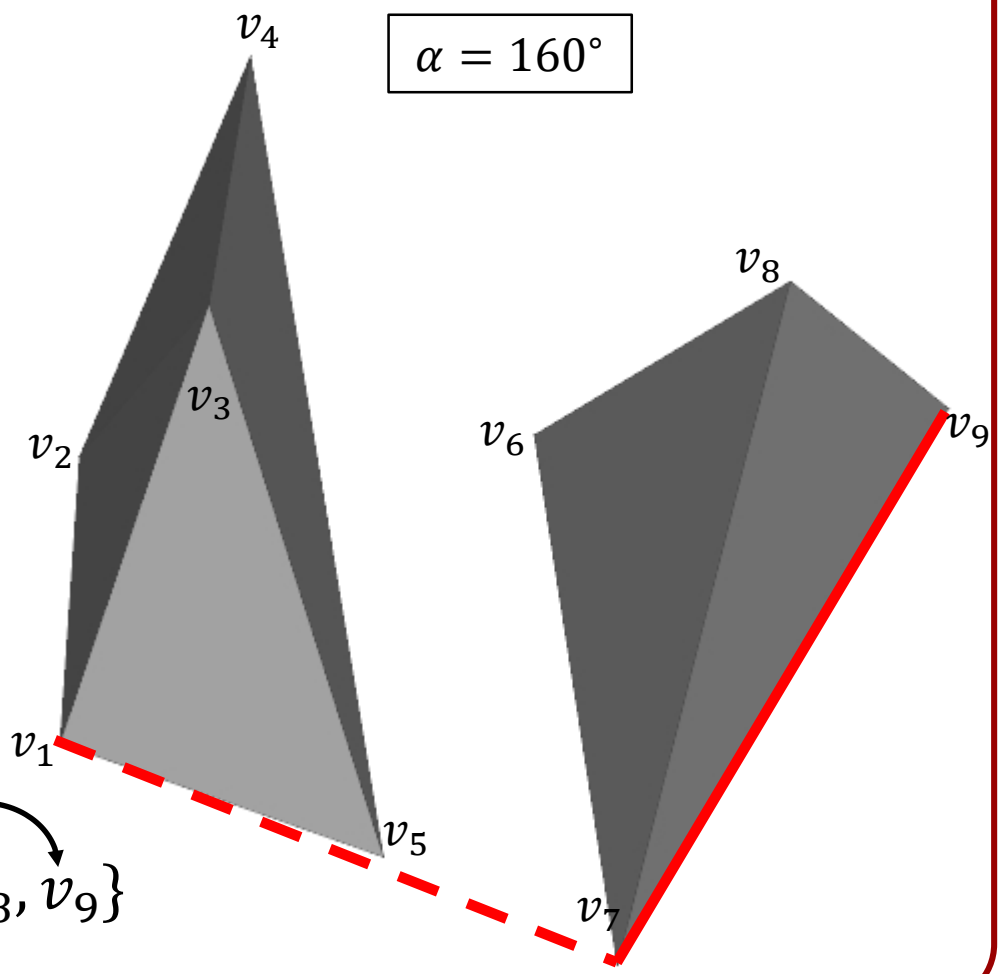
$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (\boxed{155°} \quad \cdot \quad)$$

$$R = (180°, 180°)$$

$$E = \left\{ \begin{matrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \\ (I, 100°, v_2v_5v_6) \\ (D, 105°, v_5v_6v_7) \\ (D, 110°, v_1v_2v_5) \\ (D, 155°, v_1v_5v_7) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

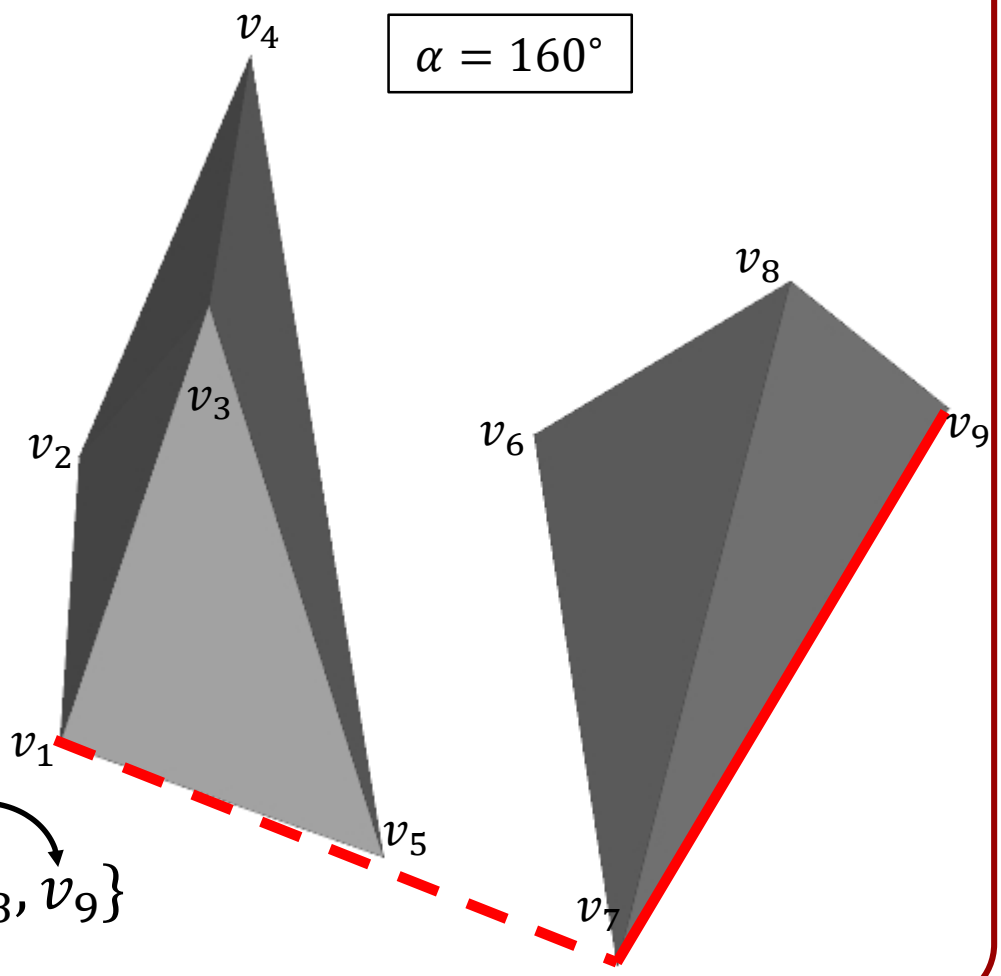
$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$L = (180°, 180°)$$
$$R = (180°, 180°)$$

$$E = \left\{ \begin{matrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \\ (I, 100°, v_2v_5v_6) \\ (D, 105°, v_5v_6v_7) \\ (D, 110°, v_1v_2v_5) \\ (D, 155°, v_1v_5v_7) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

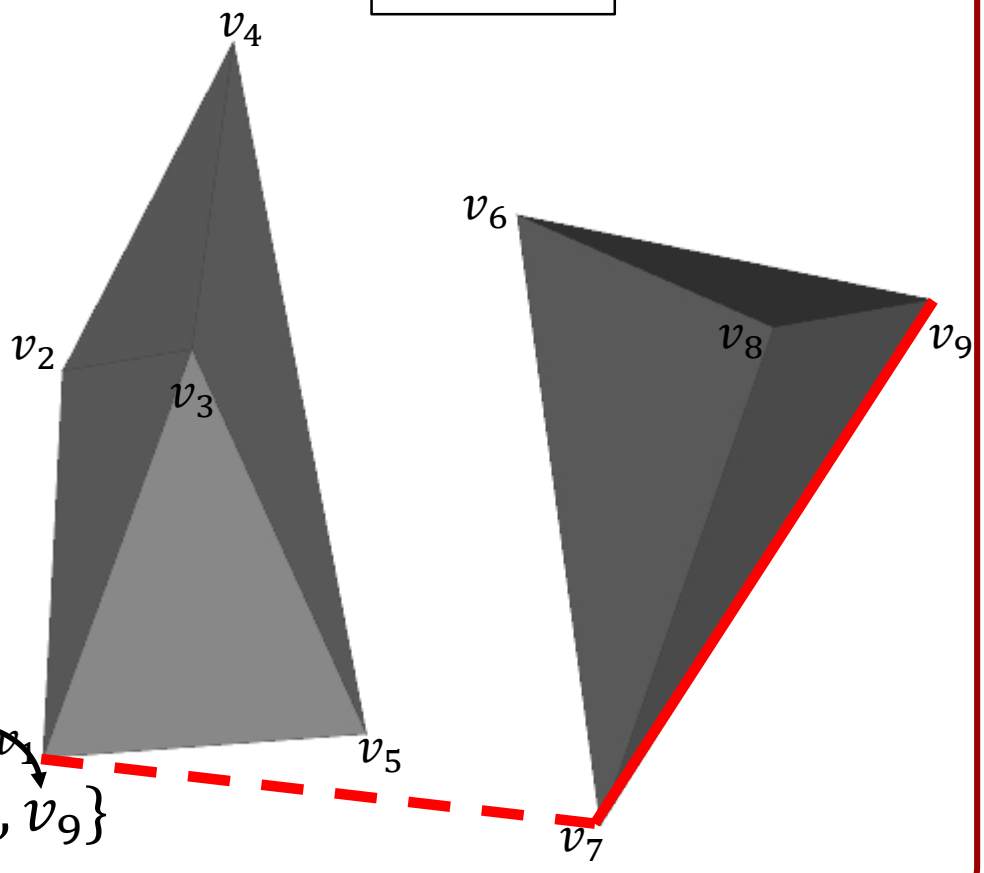
$$E_L = \begin{Bmatrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{Bmatrix}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$E = \begin{Bmatrix} (I, 10°, v_4, v_6, v_9) \\ (D, 35°, v_2v_4v_5) \\ (I, 90°, v_6, v_7, v_9) \\ (I, 100°, v_2v_5v_6) \\ (D, 105°, v_5v_6v_7) \\ (D, 110°, v_1v_2v_5) \\ (D, 155°, v_1v_5v_7) \end{Bmatrix}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

$\alpha = 180°$



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

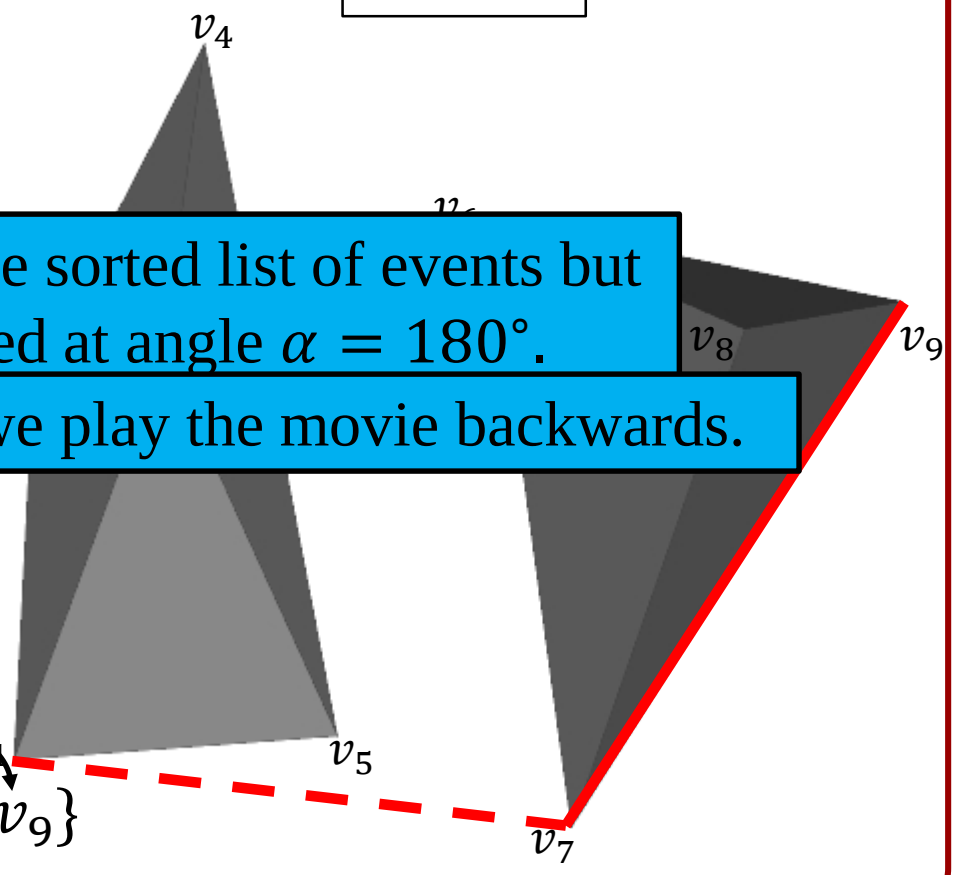
$$E_L = \left\{ \begin{matrix} (D, 50°, v_2v_4v_5) \\ (D, 110°, v_1v_2v_5) \end{matrix} \right\}$$

$$E_R = \{(I, 90°, v_6v_7v_9)\}$$

$$E = \left\{ \begin{matrix} (I, 100°, v_2v_5v_6) \\ (D, 105°, v_5v_6v_7) \\ (D, 110°, v_1v_2v_5) \\ (D, 155°, v_1v_5v_7) \end{matrix} \right\}$$

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

$\alpha = 180°$



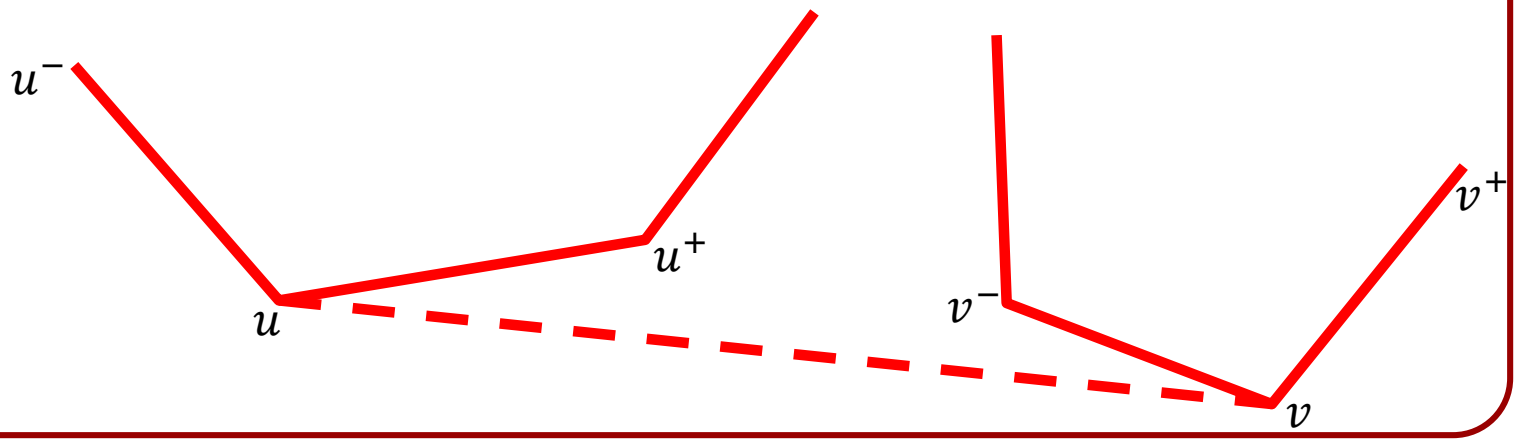
At this point we have the sorted list of events but the hull is represented at angle $\alpha = 180°$.

To get the hull at $\alpha = 0°$, we play the movie backwards.

Divide-and-Conquer [Chan, 2003]

Advance to the next event:

Given a lower tangent $\overline{uv}$ how do we find the angle α at which $p \in \{u^-, u^+, v^-, v^+\}$ is not left of $\overrightarrow{uv}$?



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

Given a lower tangent $\overline{uv}$ how do we find the angle α at which $p \in \{u^-, u^+, v^-, v^+\}$ is not left of $\overrightarrow{uv}$?

To find α we need to solve the trigonometric system:

$$\text{Area2}\big(\pi_\alpha(u), \pi_\alpha(v), \pi_\alpha(p)\big) = 0$$

with:

$$\pi_\alpha(q) = \big(q_x, \langle (q_y, q_z), (\cos\alpha\,, \sin\alpha)\rangle\big)$$

Divide-and-Conquer [Chan, 2003]

Advance to the next event:

Given a lower tangent $\overline{uv}$ how do we find the angle α at which $p \in \{u^-, u^+, v^-, v^+\}$ is not left of $\overrightarrow{uv}$?

Observation 1:

We can parameterize directions with $\alpha \in (-\pi, \pi)$ by points on the vertical line $(1, \alpha)$ with $\alpha \in (-\infty, \infty)$.

$$\pi_\alpha(q) = \big(q_x, \langle (q_y, q_z), (\cos\alpha\,, \sin\alpha) \rangle\big)$$

$$\downarrow$$

$$\pi_\alpha(q) = \left(q_x, \left\langle (q_y, q_z), \frac{(1, \alpha)}{\sqrt{1+\alpha^2}} \right\rangle\right)$$

Divide-and-Conquer [Chan, 2003]

Advance to the next event:

Given a lower tangent $\overline{uv}$ how do we find the angle α at which $p \in \{u^-, u^+, v^-, v^+\}$ is not left of $\overrightarrow{uv}$?

Observation 1:

This is no longer trigonometric, but it is still non-linear.

points on the vertical line $(1, \alpha)$ with $\alpha \in (-\infty, \infty)$:

$$\pi_\alpha(q) = \left(q_x, \left\langle (q_y, q_z), (\cos\alpha\,, \sin\alpha) \right\rangle\right)$$

$$\downarrow$$

$$\pi_\alpha(q) = \left(q_x, \left\langle (q_y, q_z), \frac{(1, \alpha)}{\sqrt{1+\alpha^2}} \right\rangle\right)$$

Divide-and-Conquer [Chan, 2003]

Advance to the next event:

Given a lower tangent $\overline{uv}$ how do we find the angle α at which $p \in \{u^-, u^+, v^-, v^+\}$ is not left of $\overrightarrow{uv}$?

Observation 2:

Since convex hulls are preserved under affine transformations, we can scale the y-component:

$$\pi_\alpha(q) = \left(q_x, \left\langle (q_y, q_z), \frac{(1,\alpha)}{\sqrt{1+\alpha^2}} \right\rangle\right)$$

$$\downarrow$$

$$\pi_\alpha(q) = \left(q_x, \langle (q_y, q_z), (1,\alpha) \rangle\right)$$

Divide-and-Conquer [Chan, 2003]

Advance to the next event:

Given a lower tangent $\overline{uv}$ how do we find the angle α at which $p \in \{u^-, u^+, v^-, v^+\}$ is not left of $\overrightarrow{uv}$?

Observation 2:

Sir

This is why the algorithm only sees half the hull.

transformations, we can scale the y-component:

$$\pi_\alpha(q) = \left(q_x, \left\langle (q_y, q_z), \frac{(1, \alpha)}{\sqrt{1+\alpha^2}} \right\rangle\right)$$

$$\downarrow$$

$$\pi_\alpha(q) = \left(q_x, \langle (q_y, q_z), (1, \alpha) \rangle\right)$$

Outline

- Incremental
- Divide-and-Conquer
- Higher Dimensions

Higher Dimensions

In 3D, the maximal complexity of the convex hull is linear in the number of input samples:

- $\chi = F - E + V$:
  - » Triangle mesh:

$$E = \frac{3F}{2} \Leftrightarrow F \approx 2V$$

In higher dimensions, the number of simplices on the hull can be as large as $O(V^{\lfloor d/2 \rfloor})$, giving a lower-bound on the worst-case complexity of convex hull computation.