



Convex Hulls (3D)

O'Rourke, Chapter 4

Announcements



- Assignment 2 has been posted (yay!)

Outline

- Incremental
- Divide-and-Conquer
- Higher Dimensions

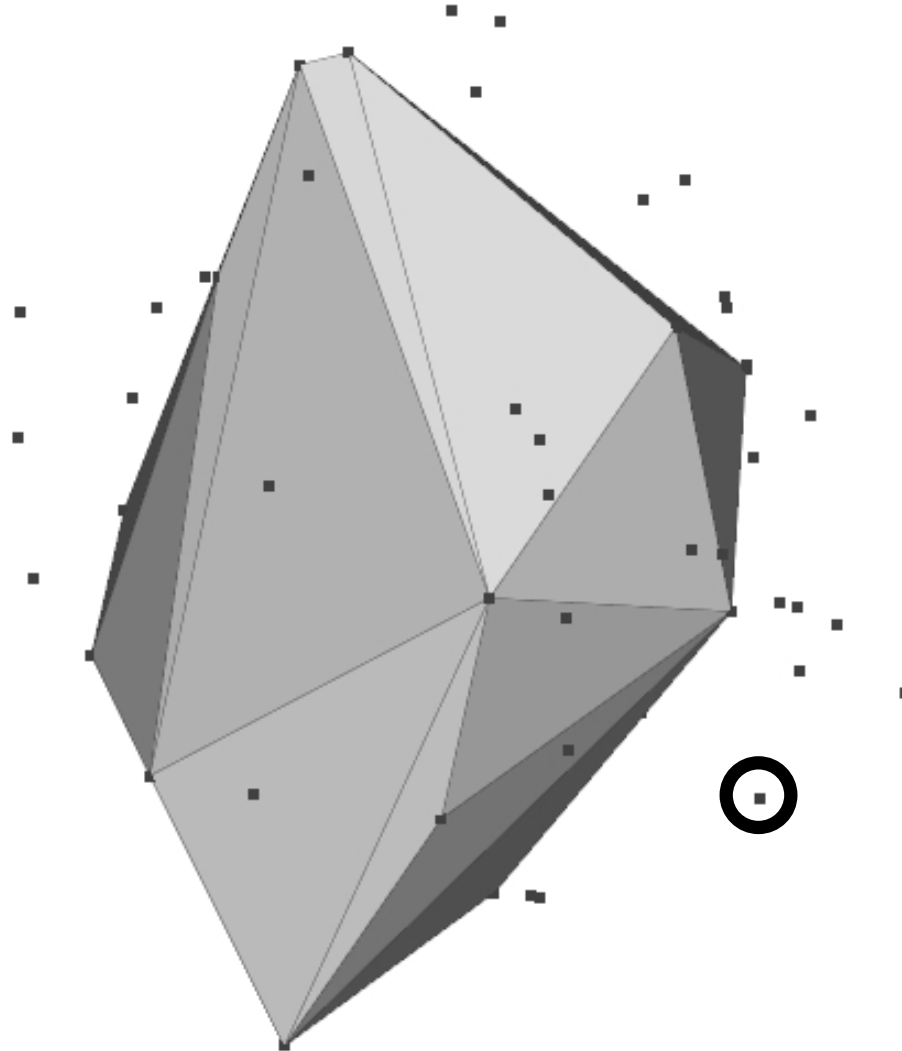




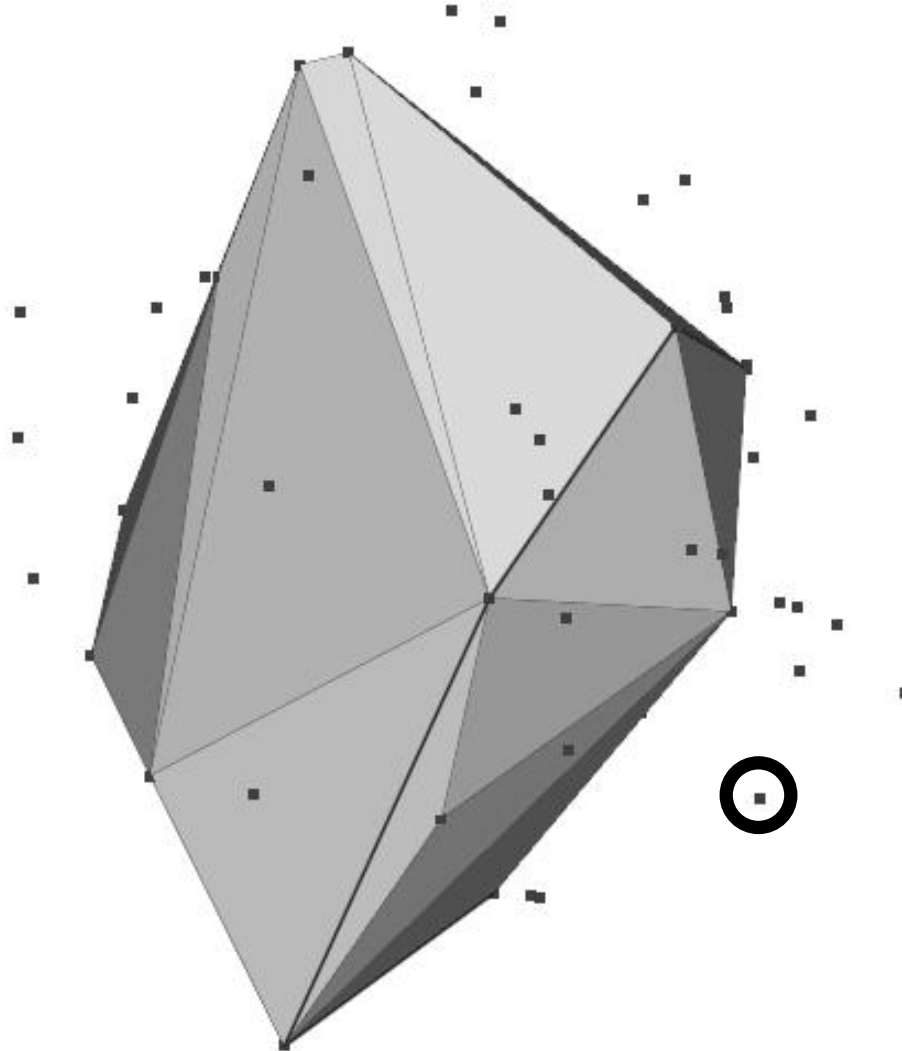
Incremental Algorithm

- 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

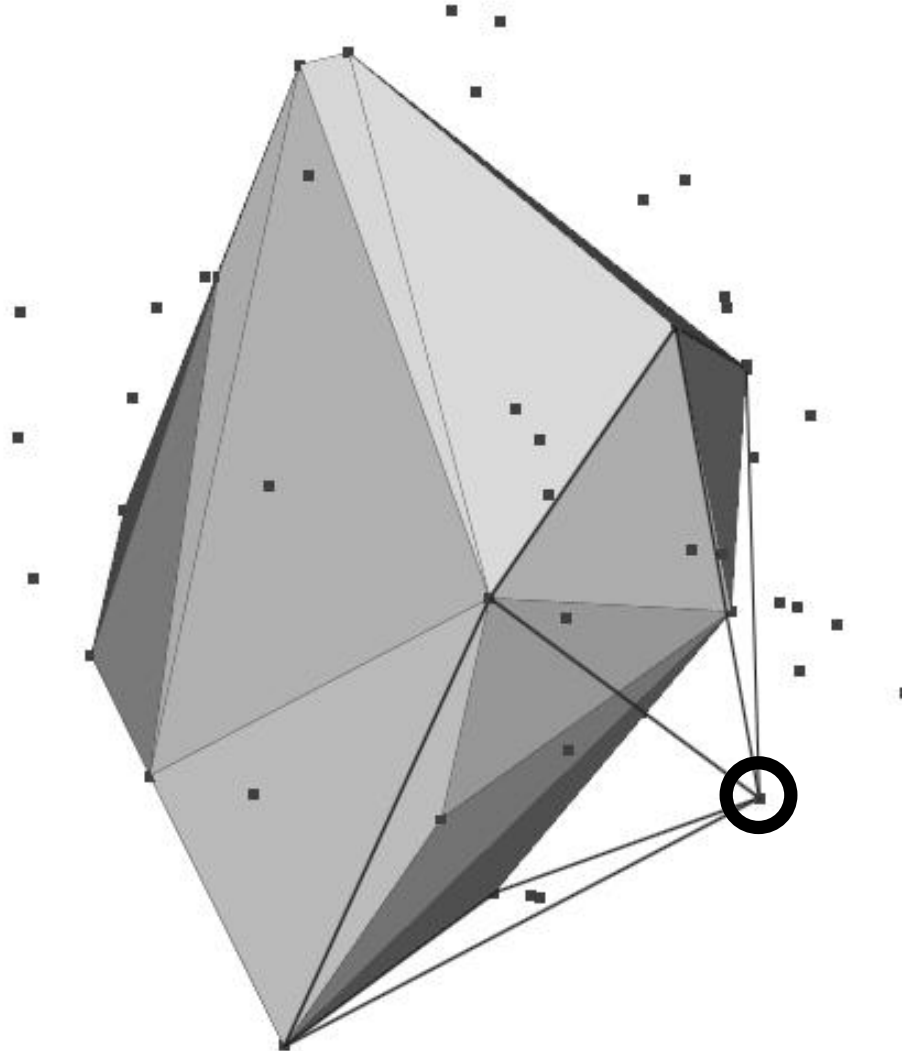
Incremental Algorithm



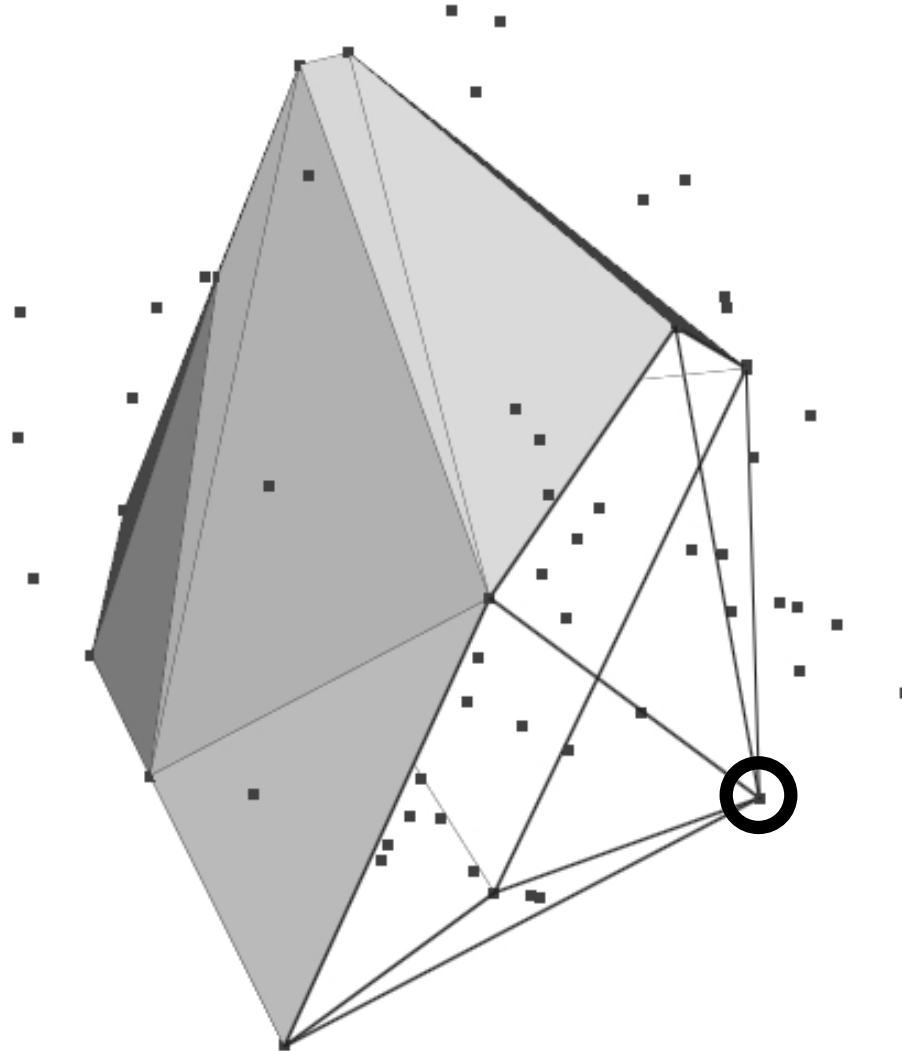
Incremental Algorithm



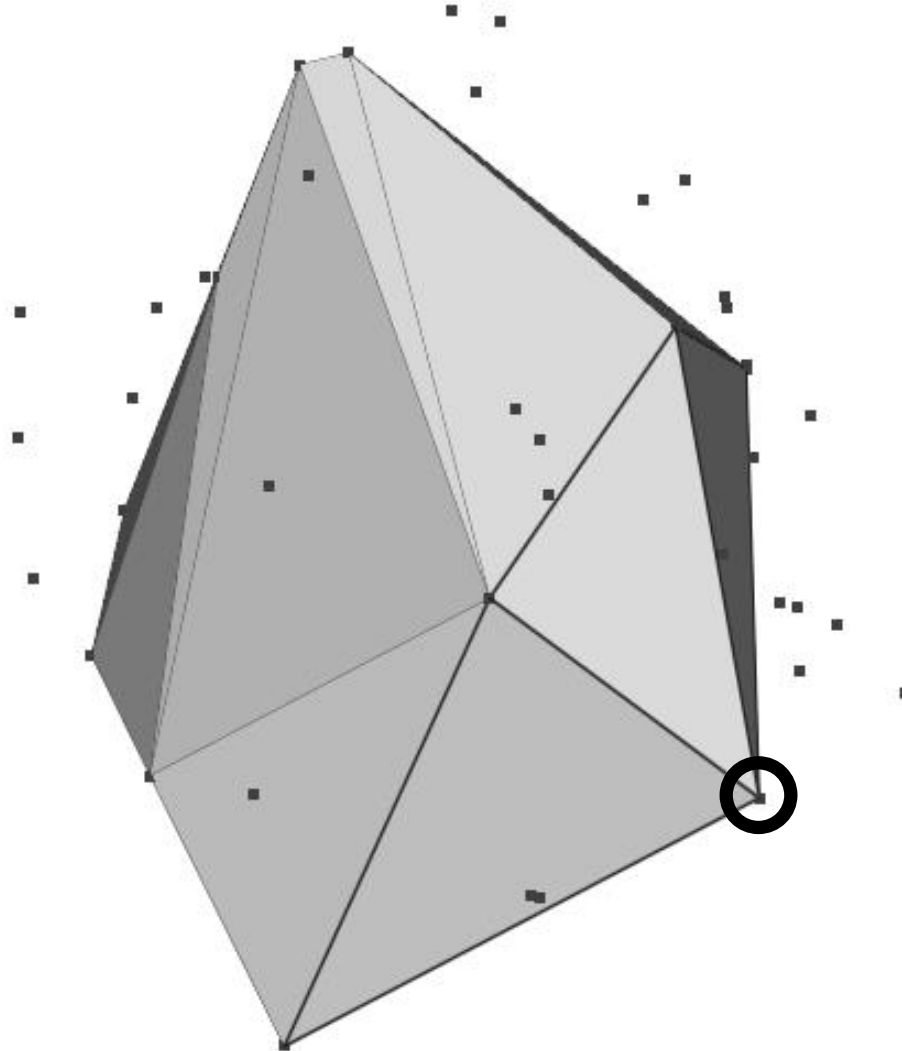
Incremental Algorithm



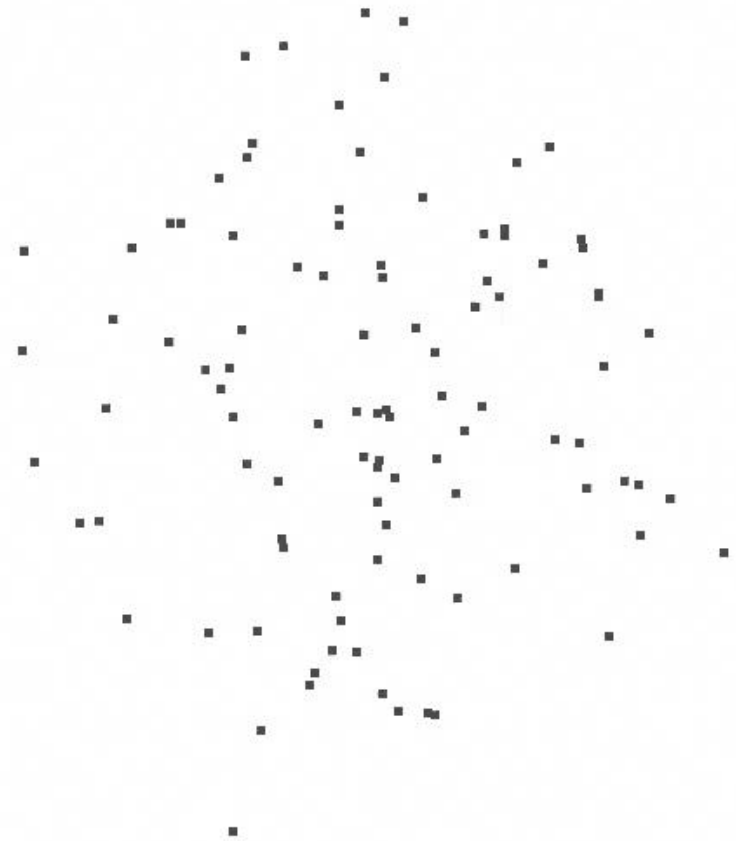
Incremental Algorithm



Incremental Algorithm



Incremental Algorithm





Incremental Algorithm

IncrmentalHull(P):

$H \leftarrow \text{InitializeHull}(P)$

for $p \in P$

$\text{AddPointToHull}(p , H)$

$P.\text{remove}(p)$



Incremental Algorithm

InitializeHull(P):

for $i \in [0, |P|-3)$

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")

Complexity: $O(|P|)$



Incremental Algorithm

AddPointToHull(p , H):

for $f \in \text{Faces}(H)$

$f.\text{visible} = \text{Right}(f.v[0], f.v[1], f.v[2], p)$

for $he \in \text{HalfEdges}(H)$

 if($he.\text{face}.\text{visible} \ \&\& \ !he.\text{opposite}.\text{face}.\text{visible}$)

$H.\text{insert}(\text{Face}(he.v[0], he.v[1], p))$

$H.\text{remove}(he.\text{face})$

for $f \in \text{Faces}(H)$

$f.\text{visible} = \text{false}$

Complexity: $O(|H|) \leq O(|P|)$



Incremental Algorithm

AddPointToHull(p , H):

for $f \in \text{Faces}(H)$

$f.\text{visible} = \text{Right}(f.v[0], f.v[1], f.v[2], p)$

for $he \in \text{HalfEdges}(H)$

i. 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.

f. Store temporary information:

- Faces store visibility flag
- Edges store the new cone face
- Etc.

Complexity: $O(|H|) \leq O(|P|)$



Incremental Algorithm

IncrmentalHull(P):

$H \leftarrow \text{InitializeHull}(P)$

for $p \in P$

$\text{AddPointToHull}(p , H)$

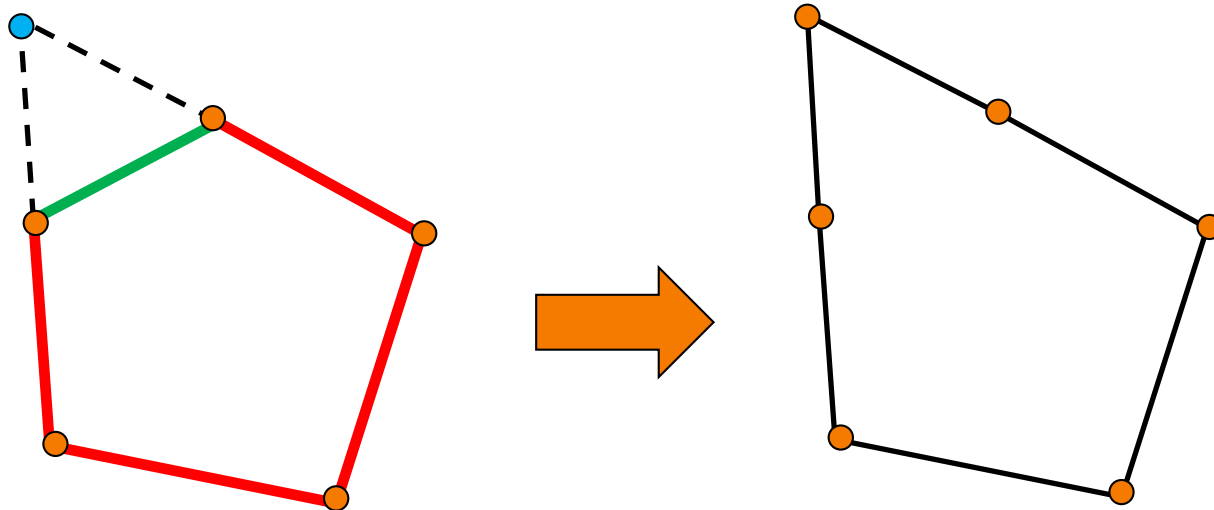
$P.\text{remove}(p)$

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



Incremental Algorithm

Note that the algorithm can result in a hull containing non-extremal points.

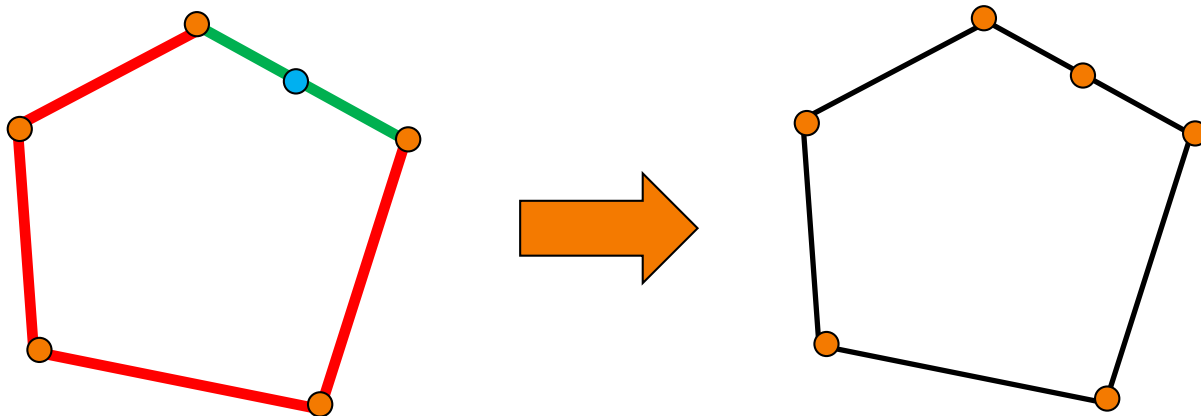


If we use **Right** to determine visibility.



Incremental Algorithm

Note that the algorithm can result in a hull containing non-extremal points.



If we use **RightOn** to determine visibility.

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:

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

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:

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

Proof:

Affine transformations take segments to segments:

- 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.

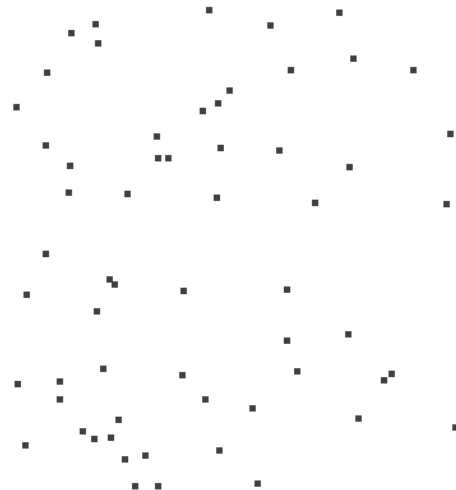
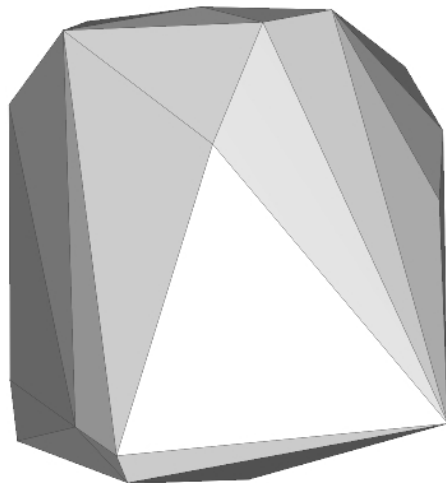
Divide-and-Conquer [Chan, 2003]



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.



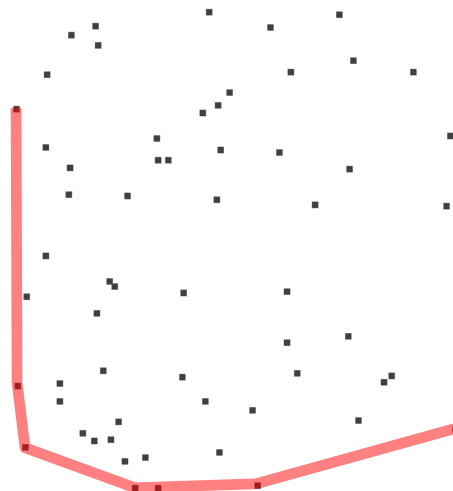
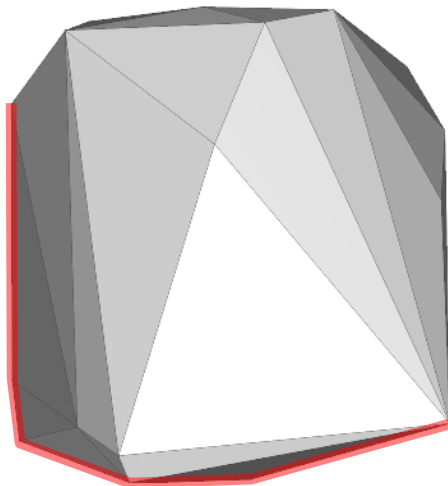
Divide-and-Conquer [Chan, 2003]



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.



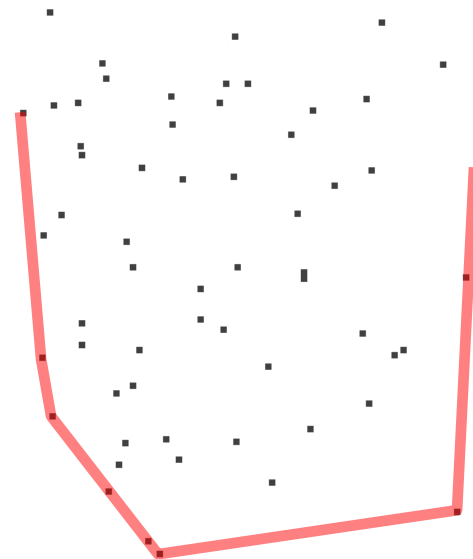
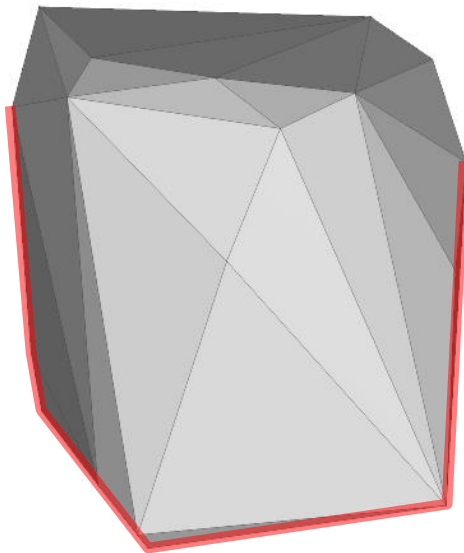
Divide-and-Conquer [Chan, 2003]



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.



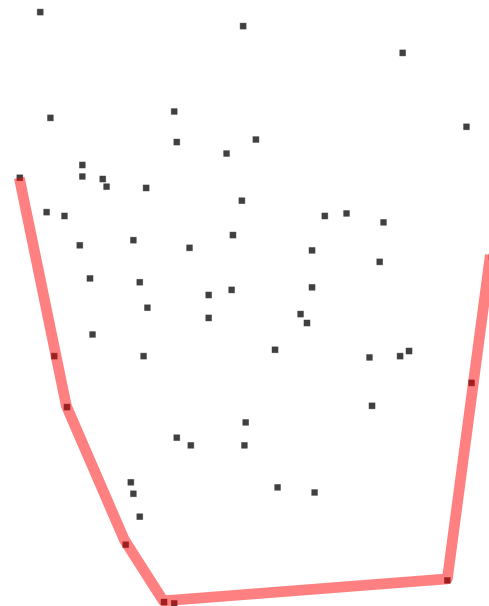
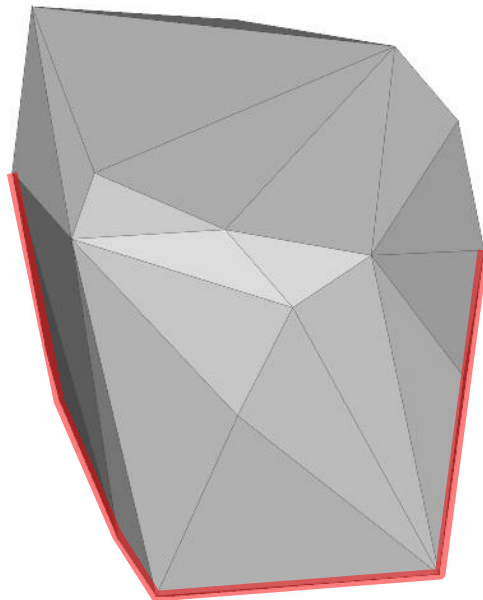
Divide-and-Conquer [Chan, 2003]



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.



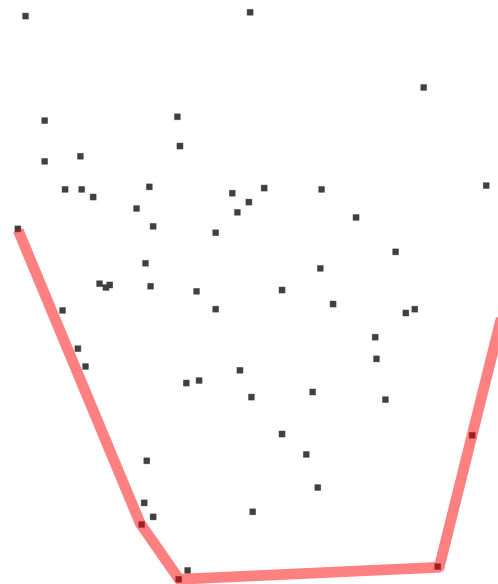
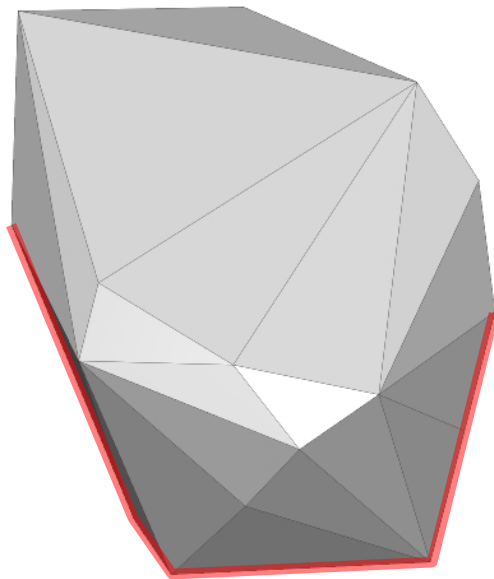
Divide-and-Conquer [Chan, 2003]



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 $\rightarrow$ two edges
- A point being deleted:
two edges $\rightarrow$ one edge

At both events, a hull triangle is generated.

Divide-and-Conquer [Chan, 2003]



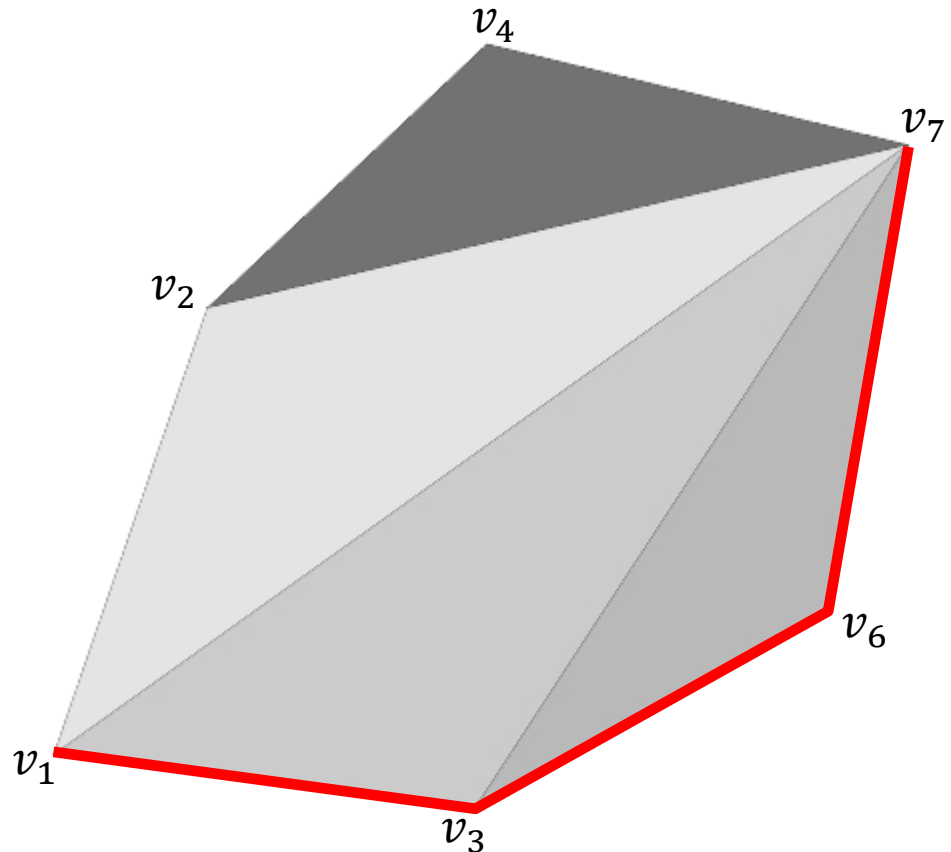
Approach:

Use a *kinetic* representation.

$$\alpha = 0^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]



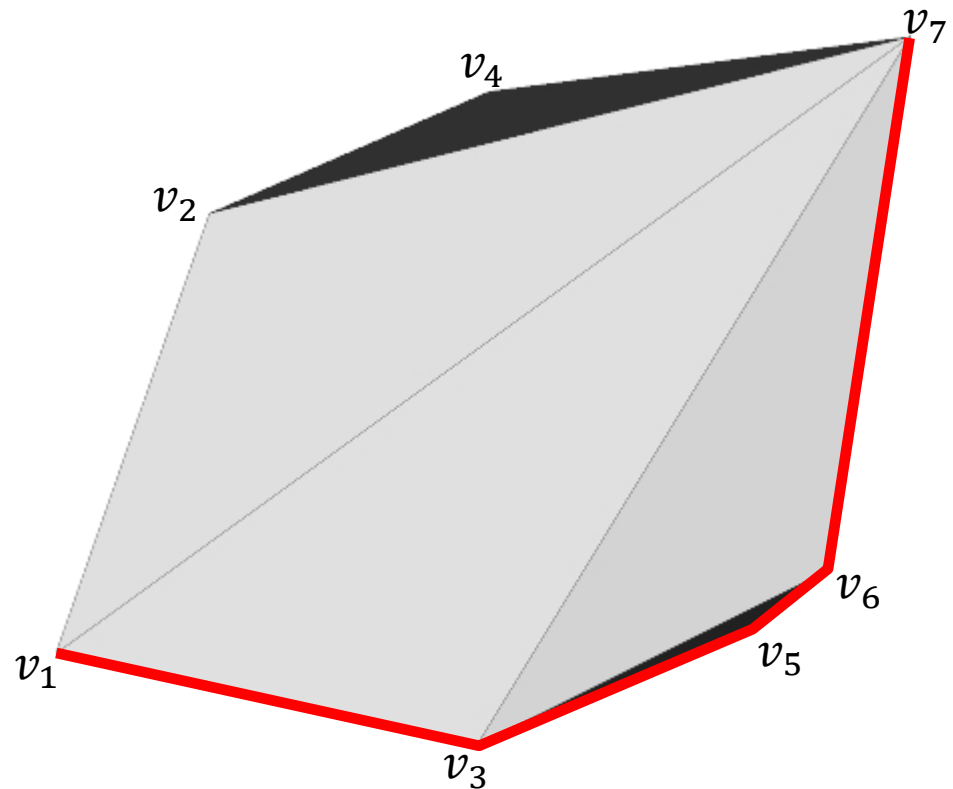
Approach:

Use a *kinetic* representation.

$$\alpha = 20^\circ$$

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

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





Divide-and-Conquer [Chan, 2003]

Approach:

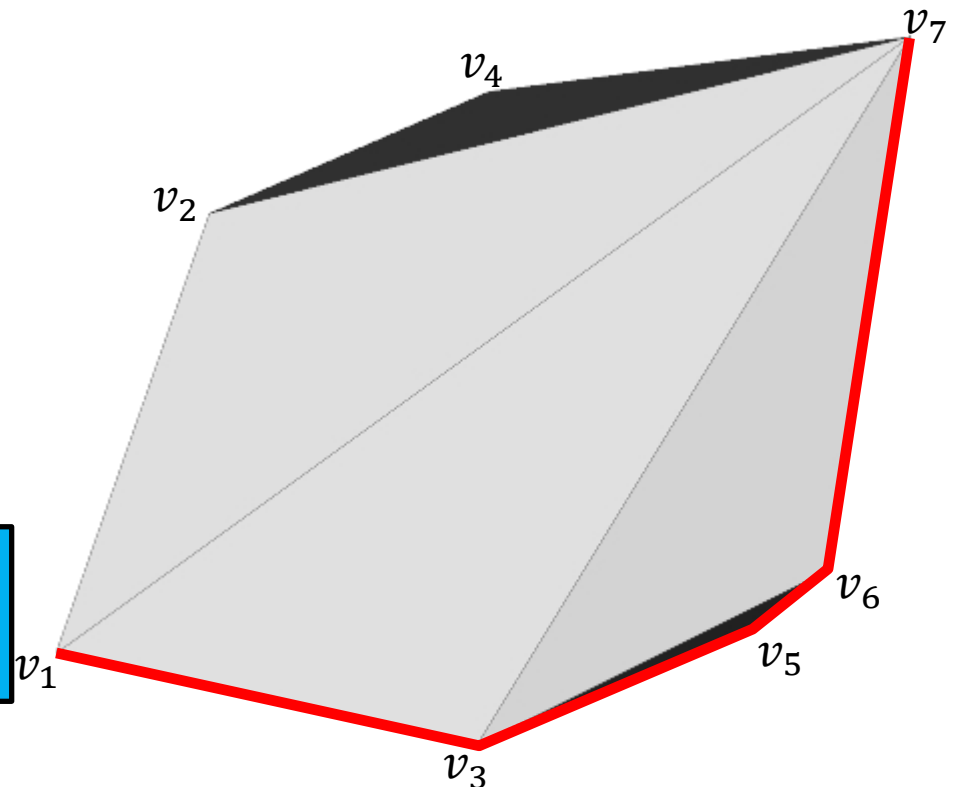
Use a *kinetic* representation.

$$\alpha = 20^\circ$$

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

The insert triplet $\{v_3, v_5, v_6\}$ 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]



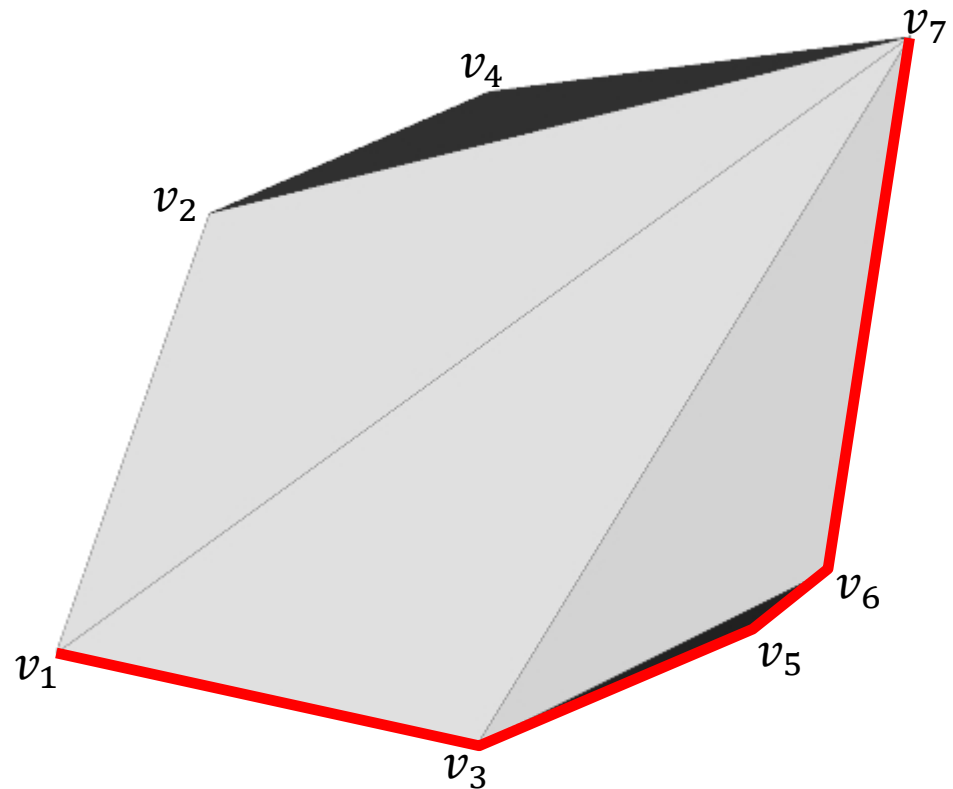
Approach:

Use a *kinetic* representation.

$$\alpha = 20^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]



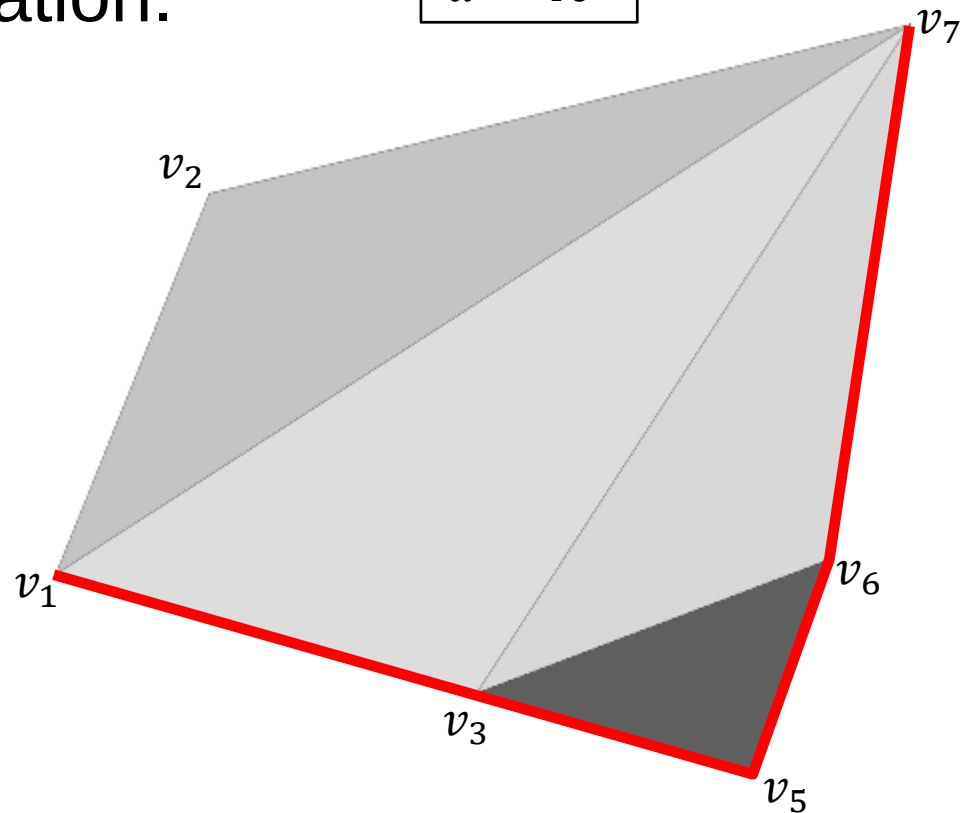
Approach:

Use a *kinetic* representation.

$$\alpha = 40^\circ$$

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

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





Divide-and-Conquer [Chan, 2003]

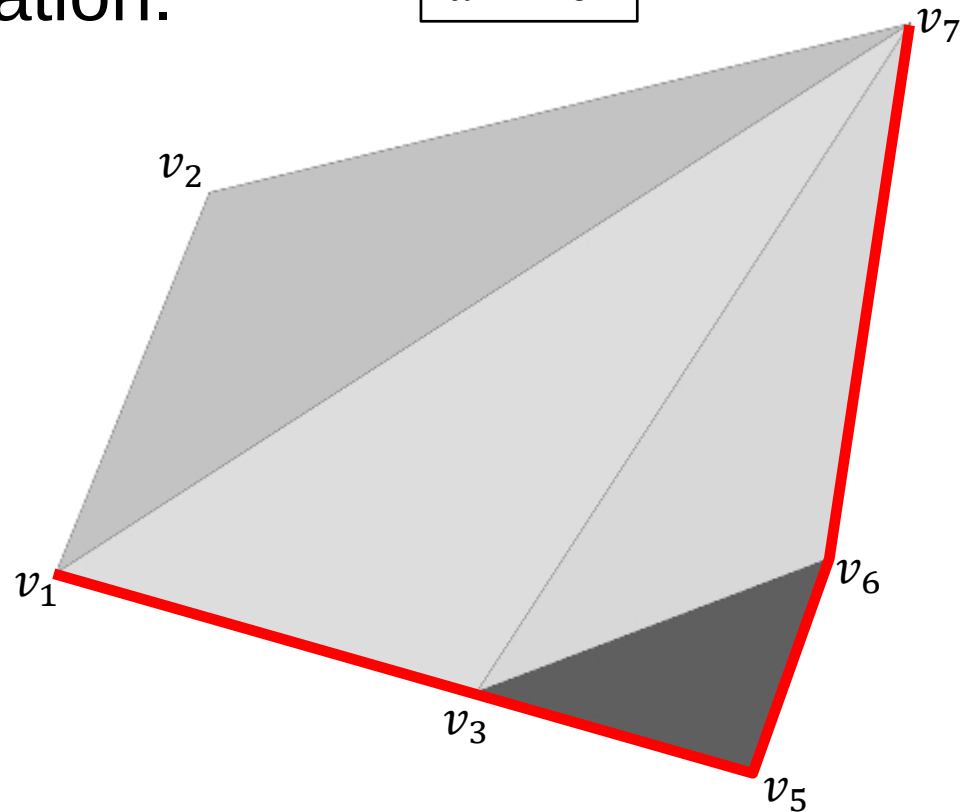
Approach:

Use a *kinetic* representation.

$$\alpha = 40^\circ$$

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

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



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

Divide-and-Conquer [Chan, 2003]



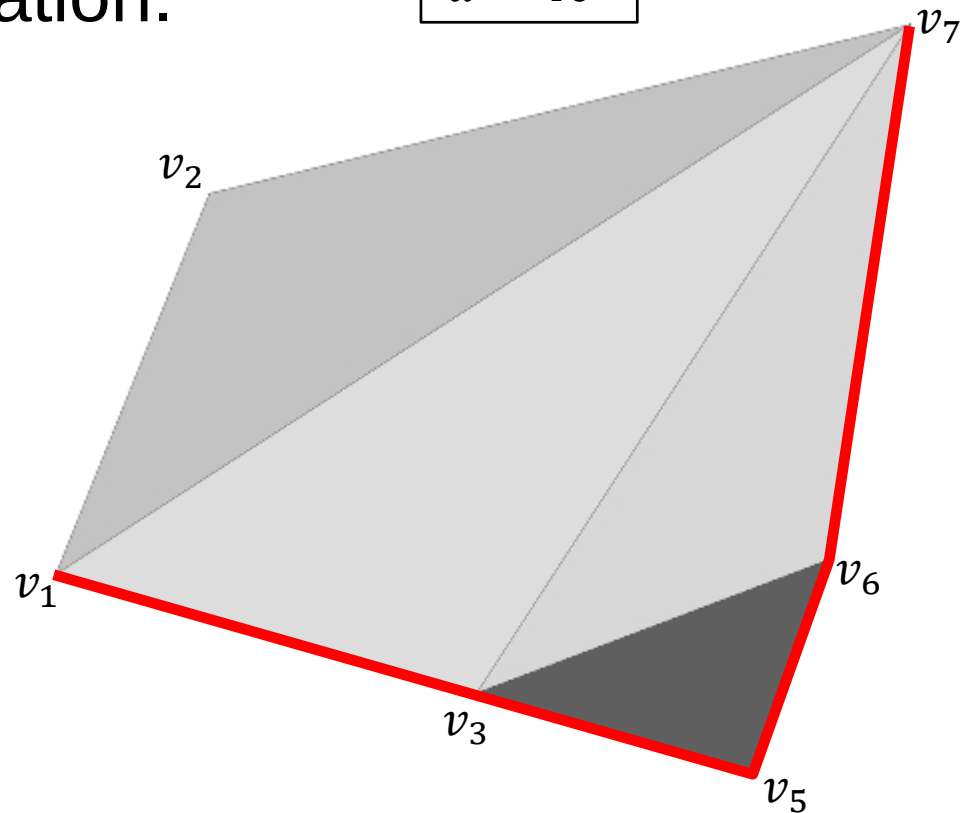
Approach:

Use a *kinetic* representation.

$$\alpha = 40^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]



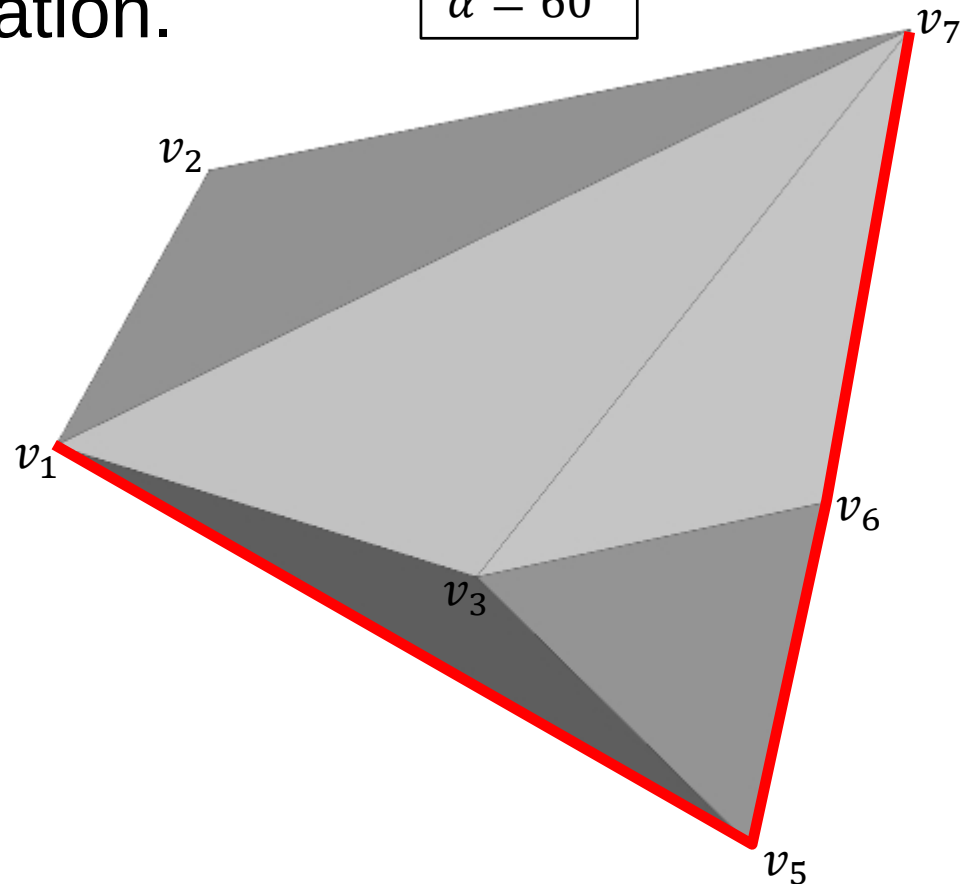
Approach:

Use a *kinetic* representation.

$$\alpha = 60^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]



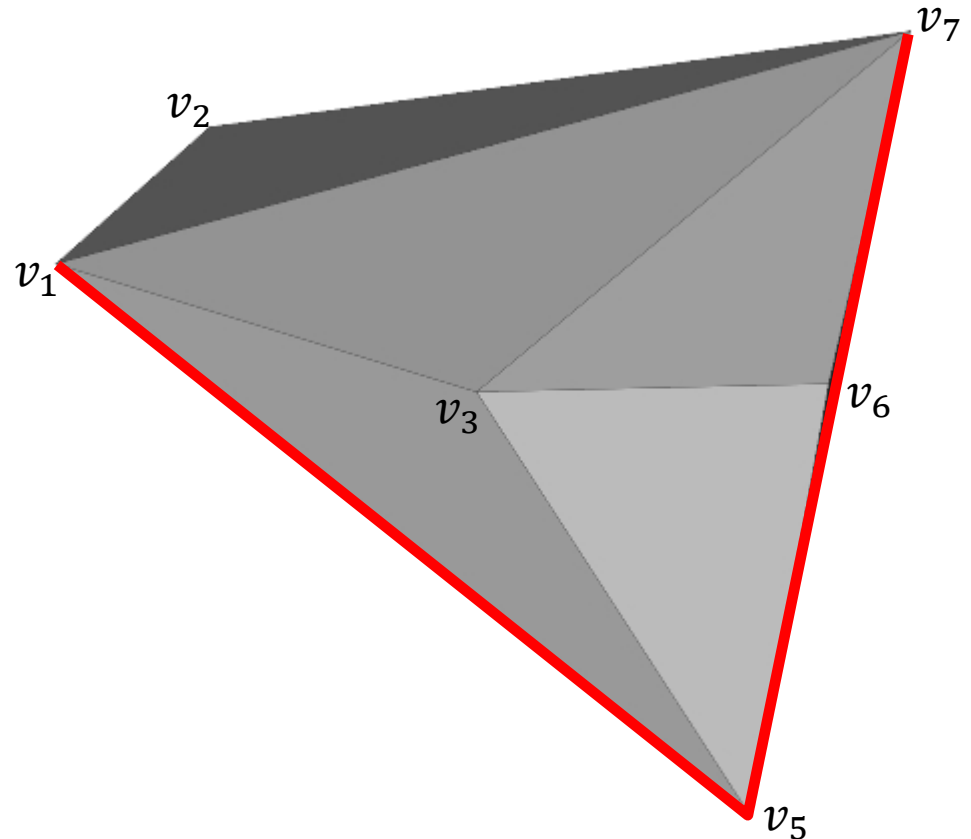
Approach:

Use a *kinetic* representation.

$$\alpha = 80^\circ$$

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

$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$
Two curved arrows above the set P. The first arrow starts at v2 and points to v4. The second arrow starts at v3 and points to v5.



Divide-and-Conquer [Chan, 2003]

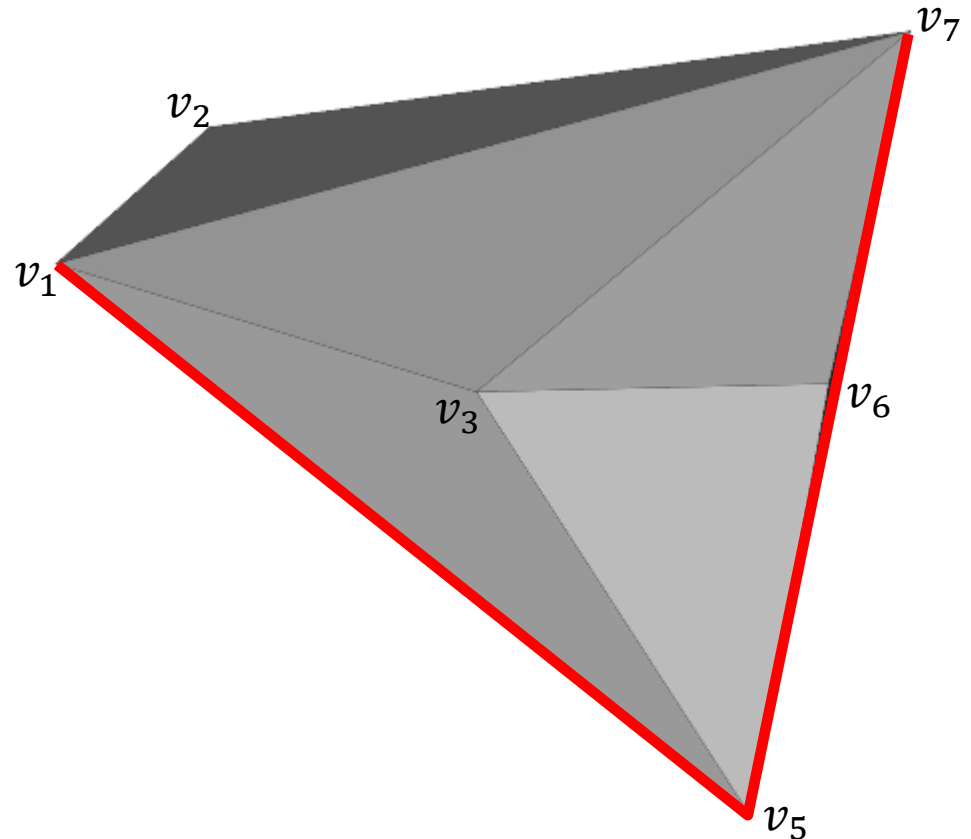


Approach:

Use a *kinetic* representation.

$$\alpha = 80^\circ$$

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



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

Divide-and-Conquer [Chan, 2003]



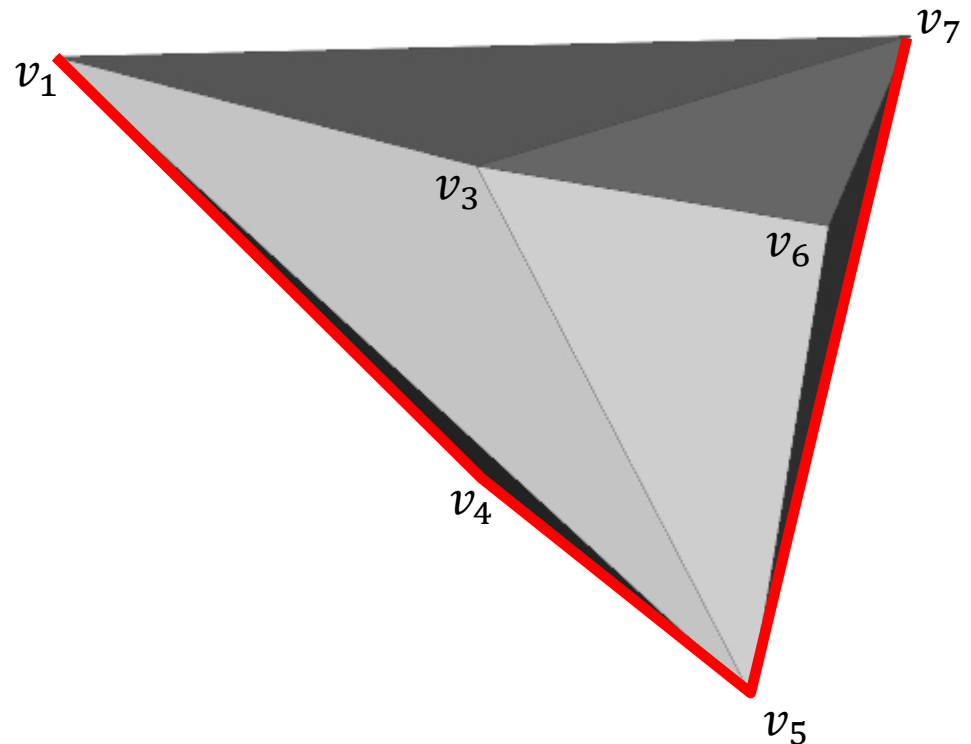
Approach:

Use a *kinetic* representation.

$$\alpha = 100^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]



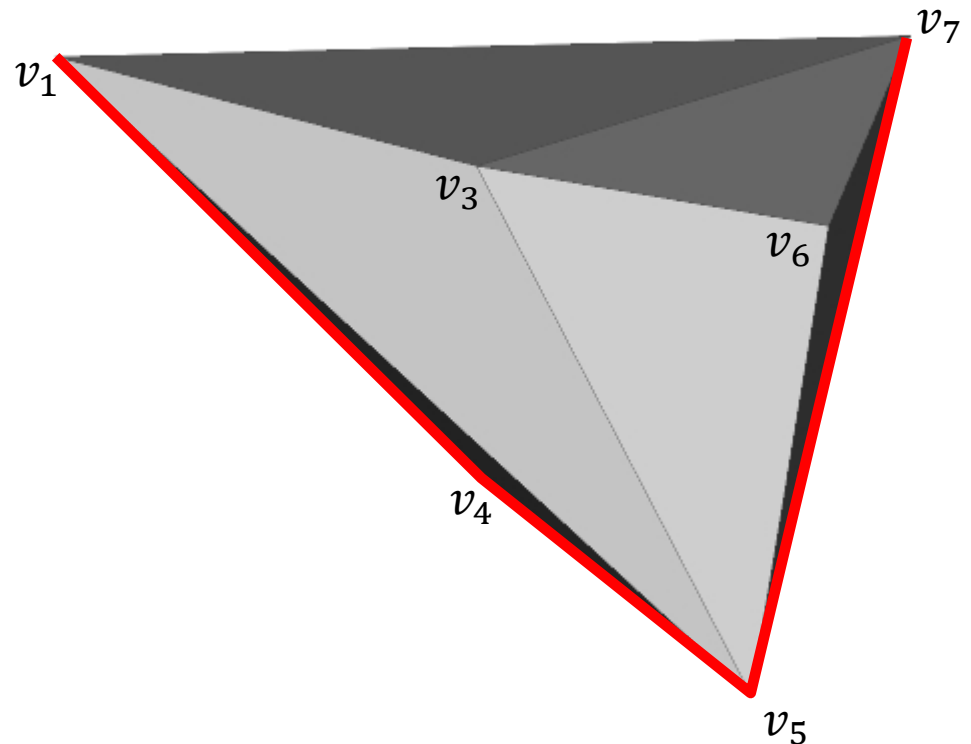
Approach:

Use a *kinetic* representation.

$$\alpha = 100^\circ$$

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

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





Divide-and-Conquer [Chan, 2003]

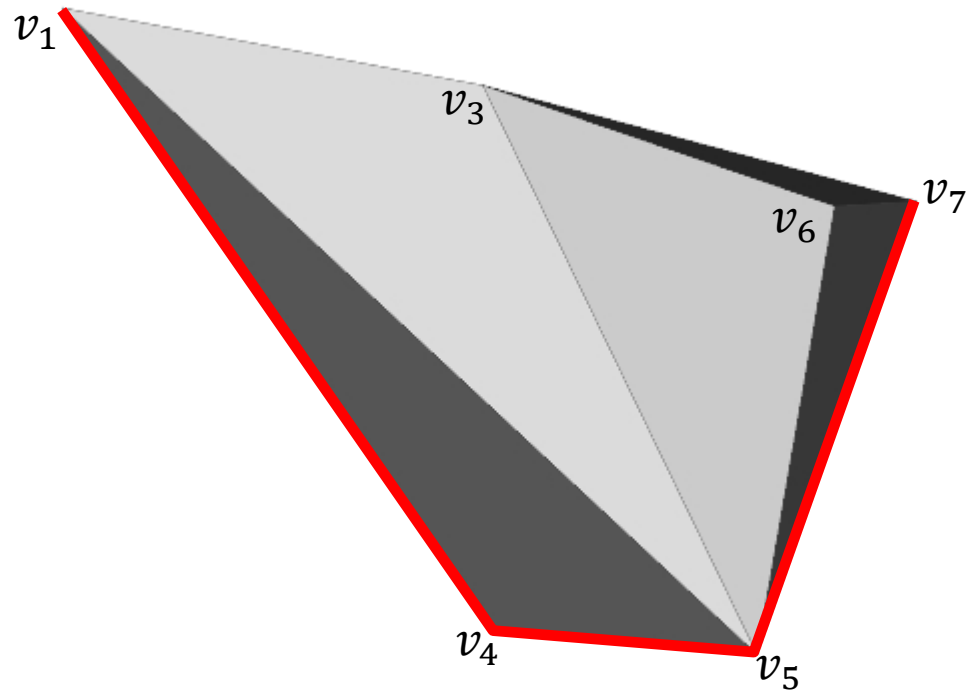
Approach:

Use a *kinetic* representation.

$$\alpha = 120^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]



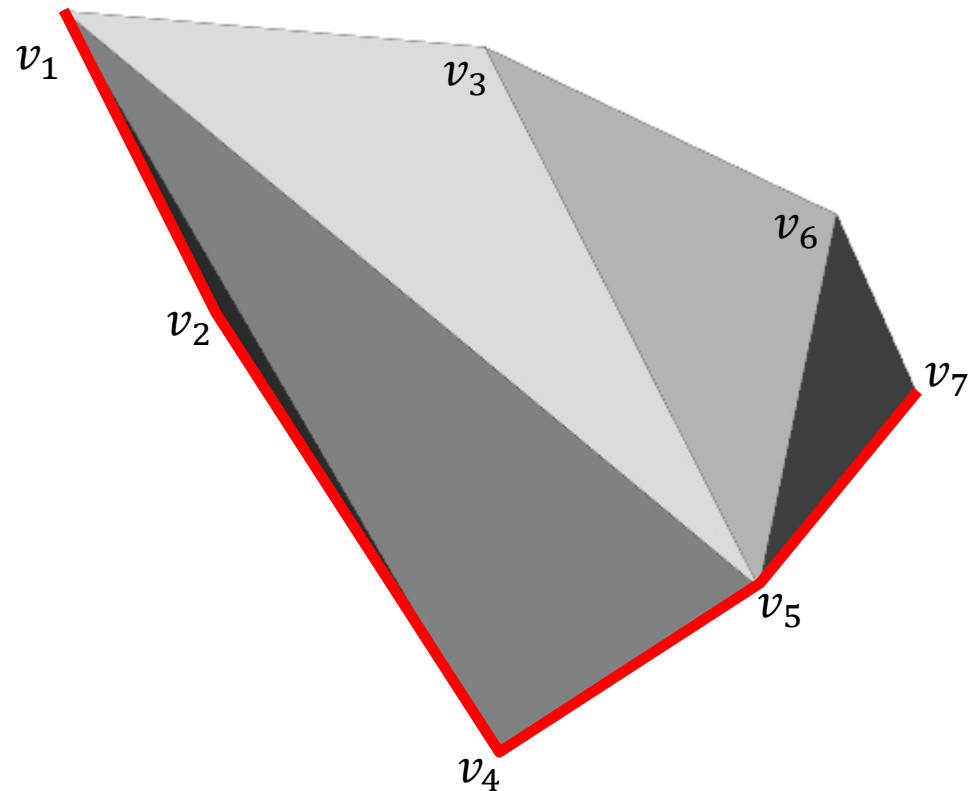
Approach:

Use a *kinetic* representation.

$$\alpha = 140^\circ$$

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

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





Divide-and-Conquer [Chan, 2003]

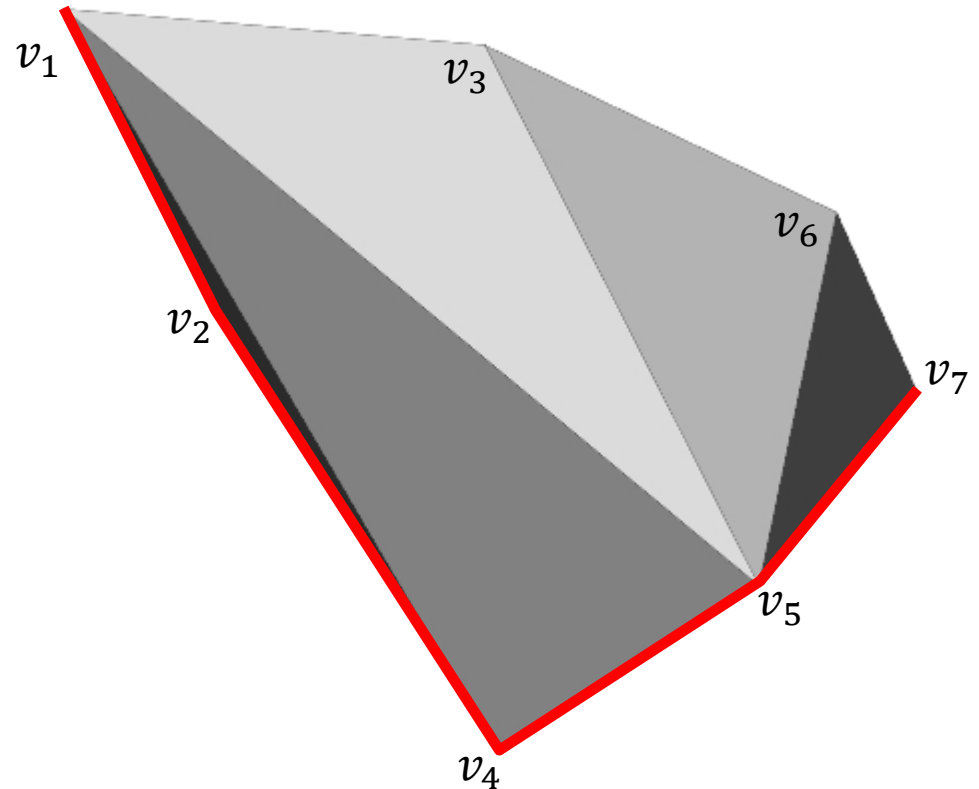
Approach:

Use a *kinetic* representation.

$$\alpha = 140^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]



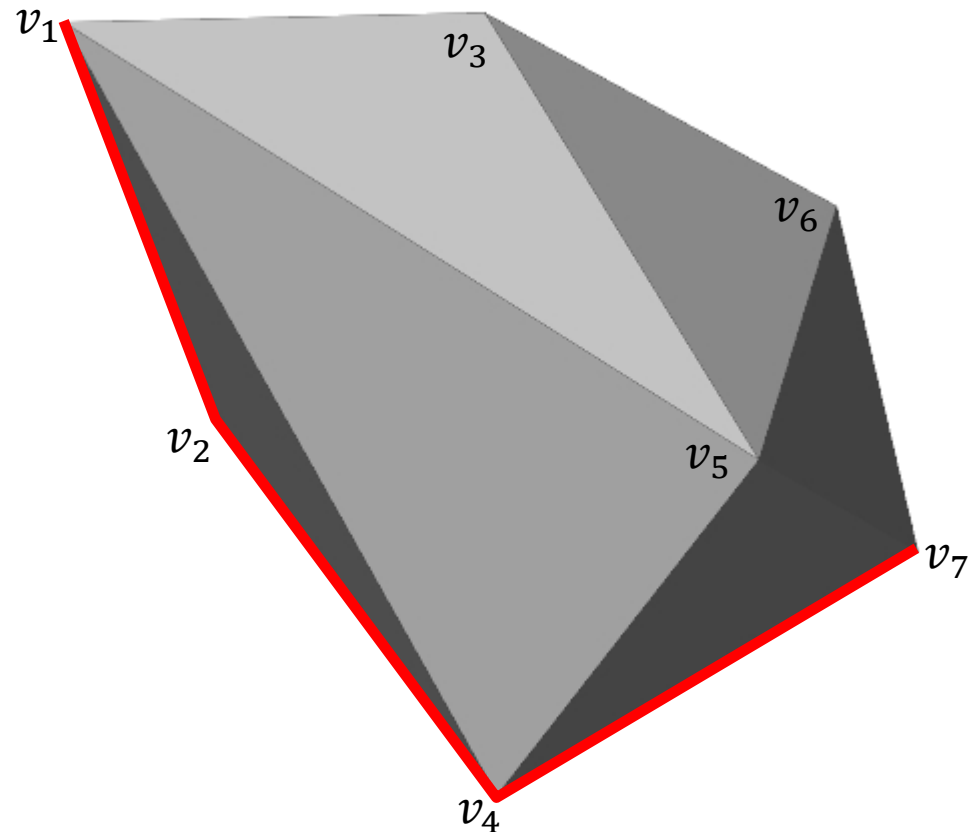
Approach:

Use a *kinetic* representation.

$$\alpha = 160^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]



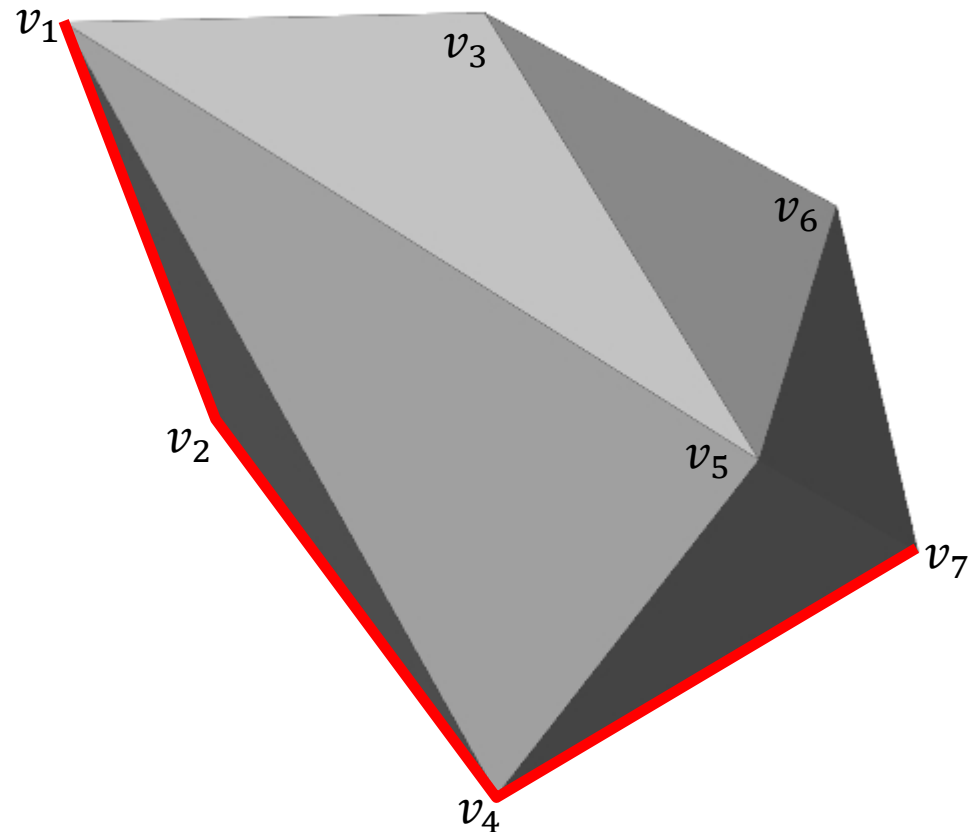
Approach:

Use a *kinetic* representation.

$$\alpha = 160^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]



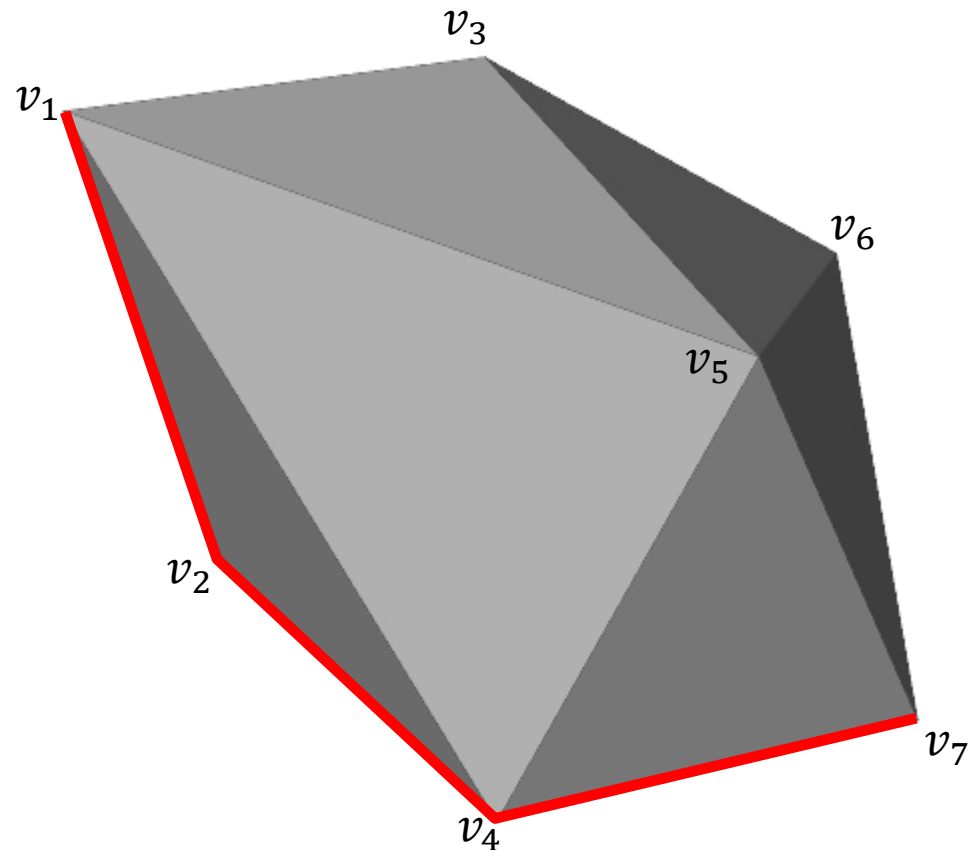
Approach:

Use a *kinetic* representation.

$$\alpha = 180^\circ$$

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

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



Divide-and-Conquer [Chan, 2003]

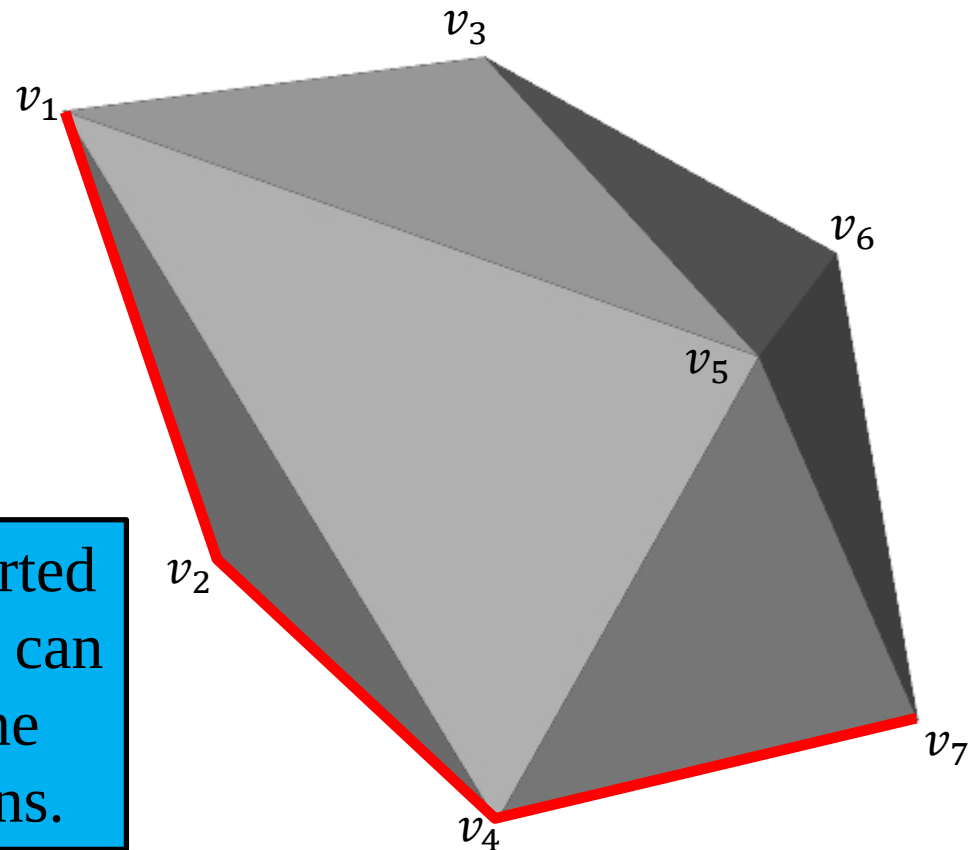
Approach:

Use a *kinetic* representation.

$$\alpha = 180^\circ$$

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

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:

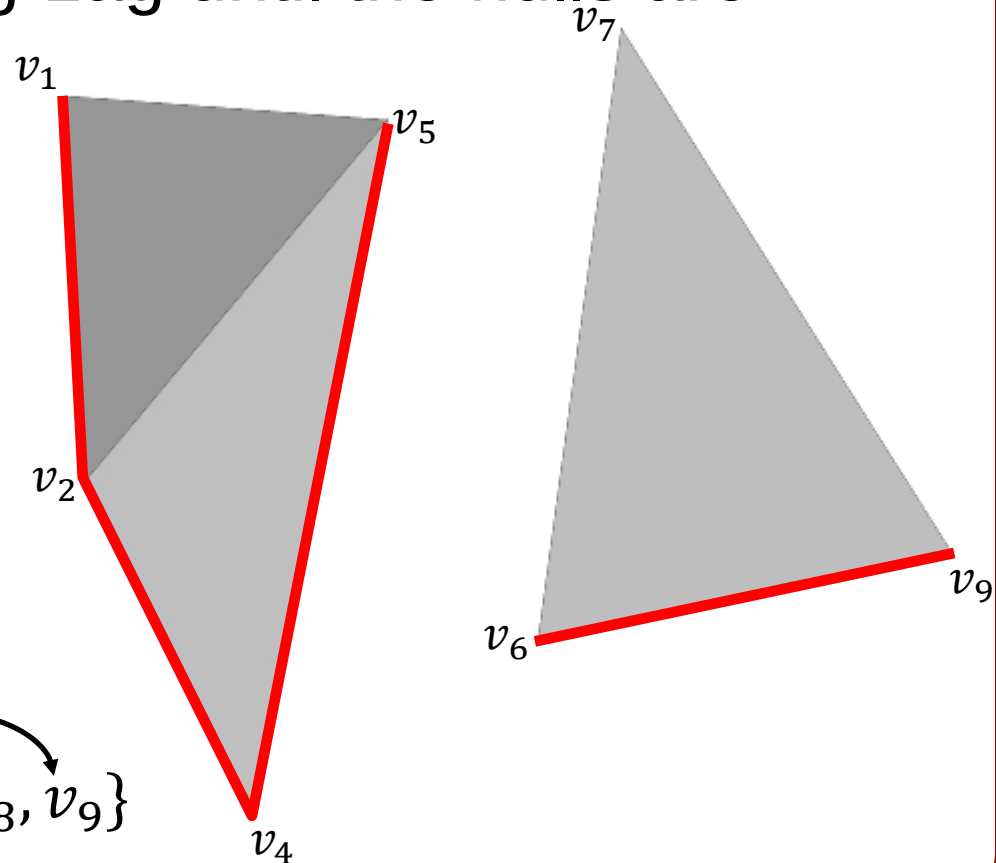
- Initialize the lower tangent.
- Compute the minimum of the angles:
 - For the next left hull event
 - For the next right hull event
 - When the lower tangent stops supporting:
 - » the left neighbor of the left end-point
 - » the right neighbor of the left end-point
 - » the left neighbor of the right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



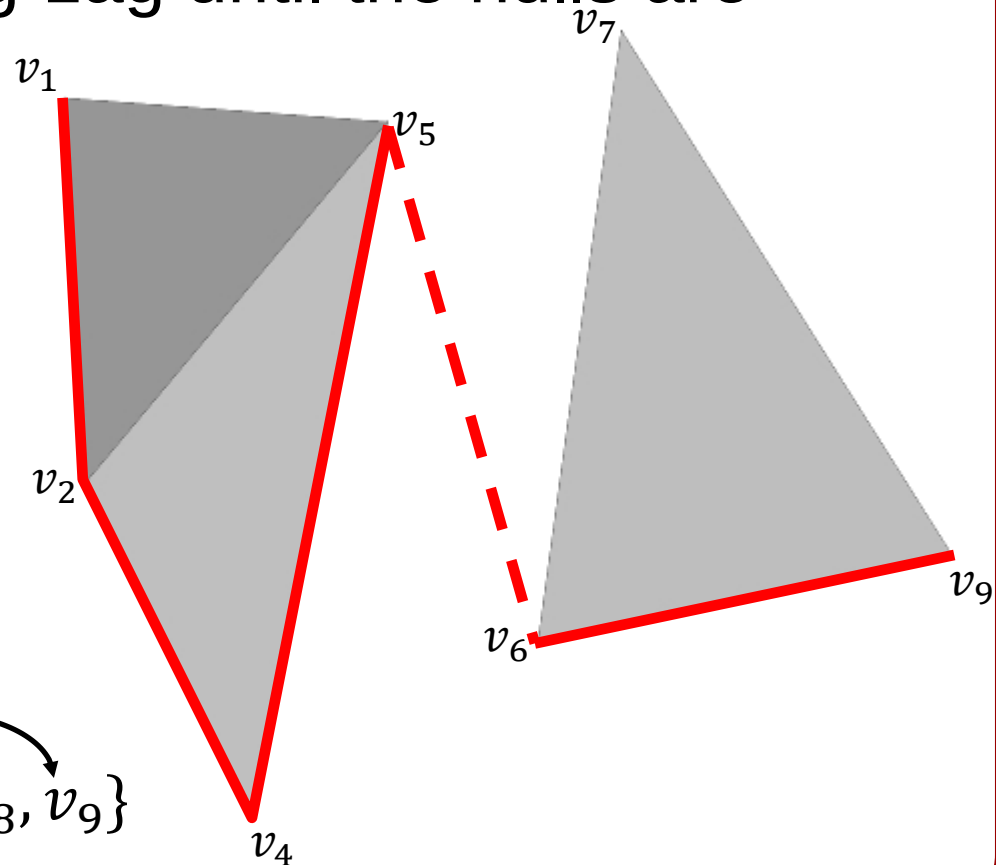
$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Initialize the Lower Tangent:

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



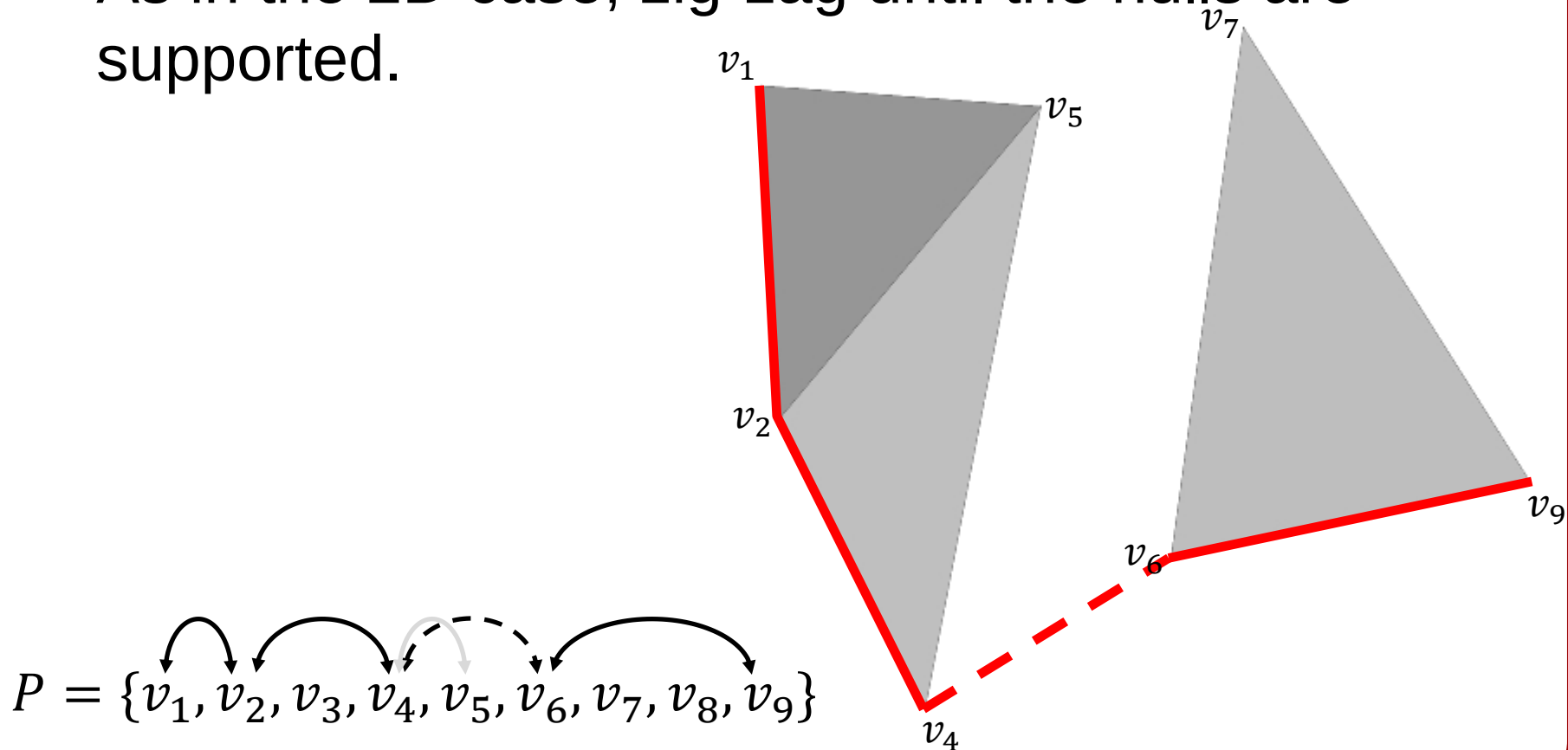
$$P = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$



Divide-and-Conquer [Chan, 2003]

Initialize the Lower Tangent:

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





Divide-and-Conquer [Chan, 2003]

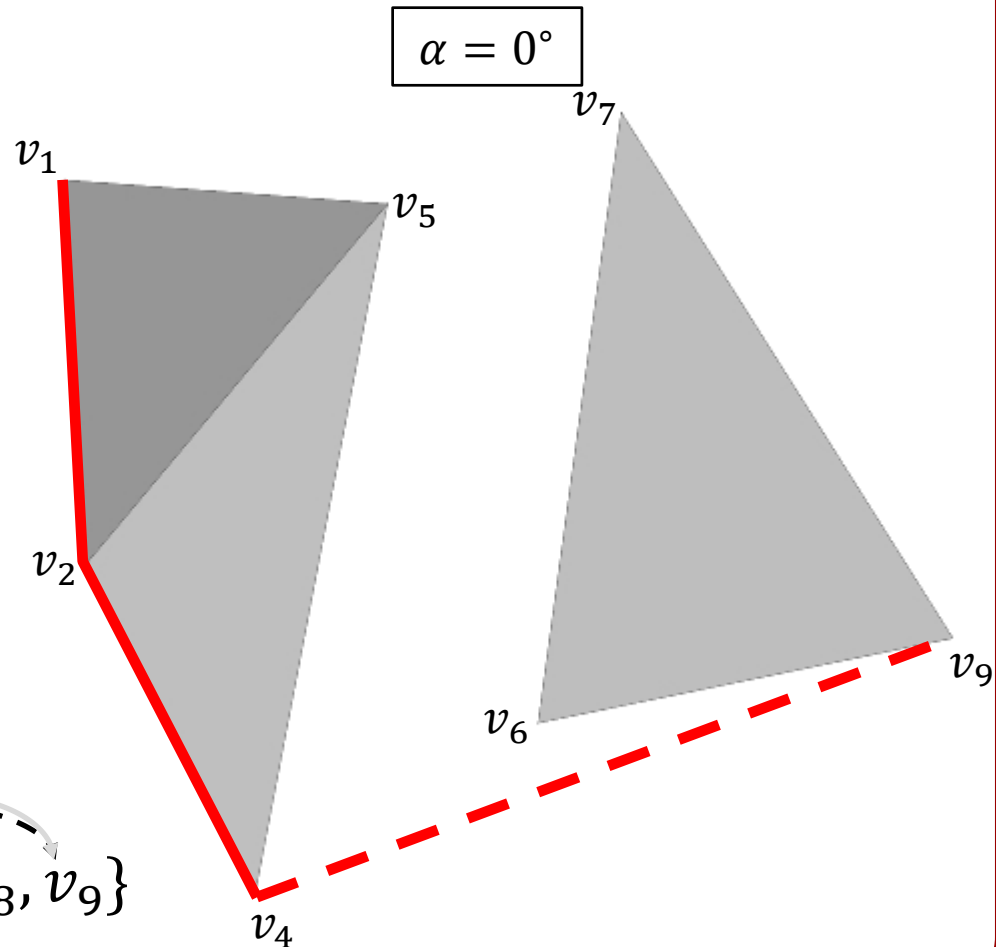
Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (45^\circ, 55^\circ)$$

$$R = (\boxed{10^\circ} \cdot)$$



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

Divide-and-Conquer [Chan, 2003]



Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

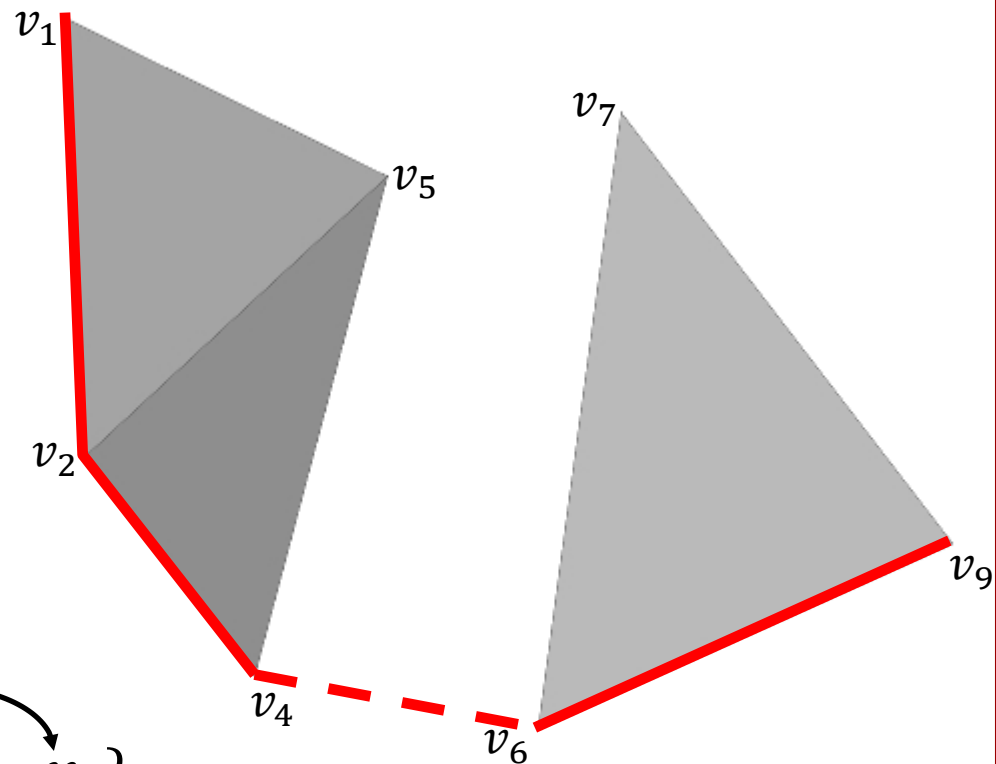
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (45^\circ, 55^\circ)$$

$$R = (\boxed{10^\circ} \cdot)$$

$$E = \{(I, 10^\circ, v_4, v_6, v_9)\}$$

$$\alpha = 20^\circ$$



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



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

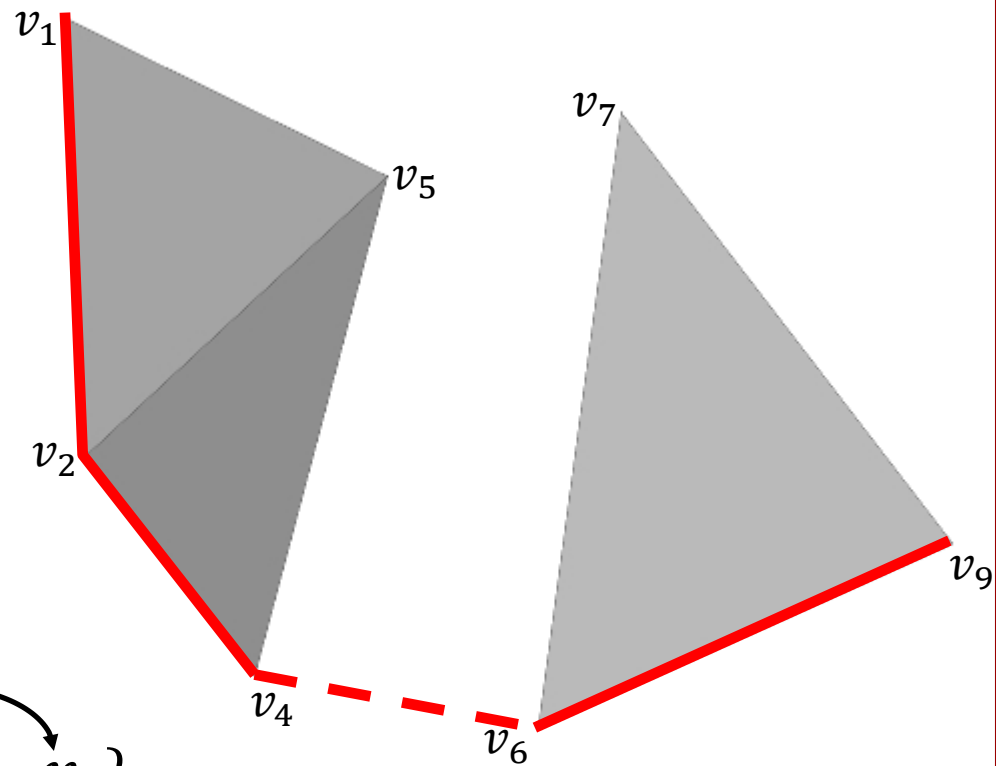
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (\boxed{35^\circ}, 70^\circ)$$

$$R = (\cdot, 170^\circ)$$

$$E = \{(I, 10^\circ, v_4, v_6, v_9)\}$$

$$\alpha = 20^\circ$$



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



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

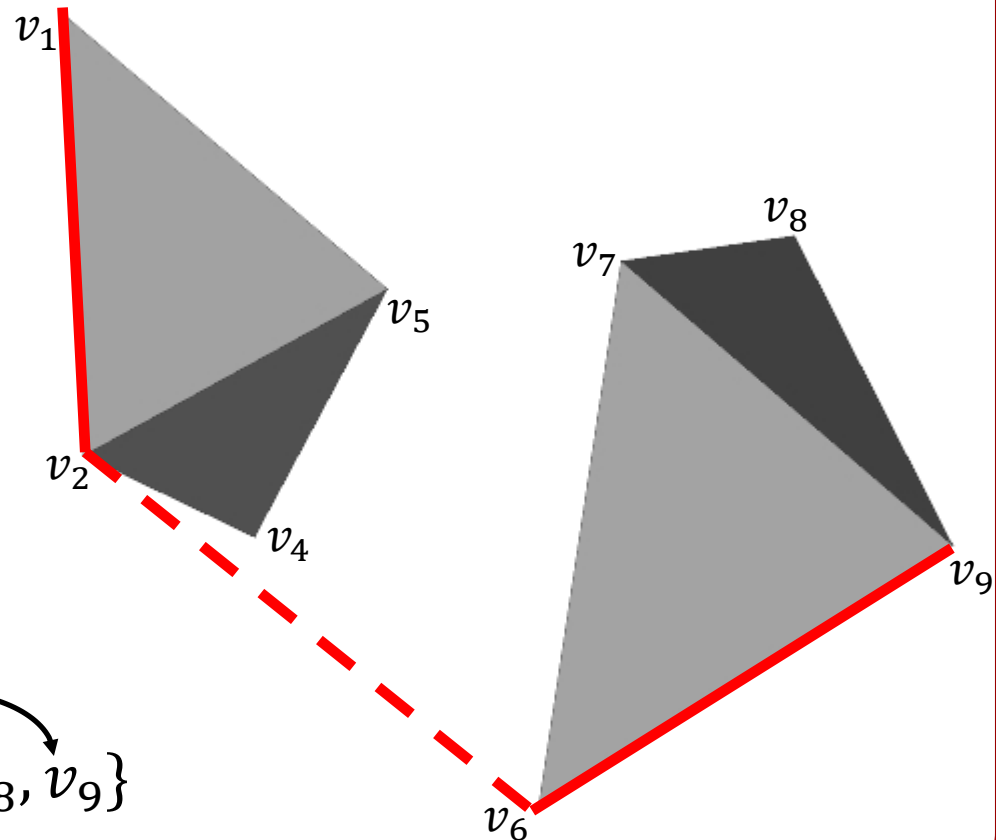
$$L = (\boxed{35^\circ}, 70^\circ)$$

$$R = (110^\circ, \cdot)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \end{array} \right\}$$

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

$$\alpha = 40^\circ$$



Divide-and-Conquer [Chan, 2003]



Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

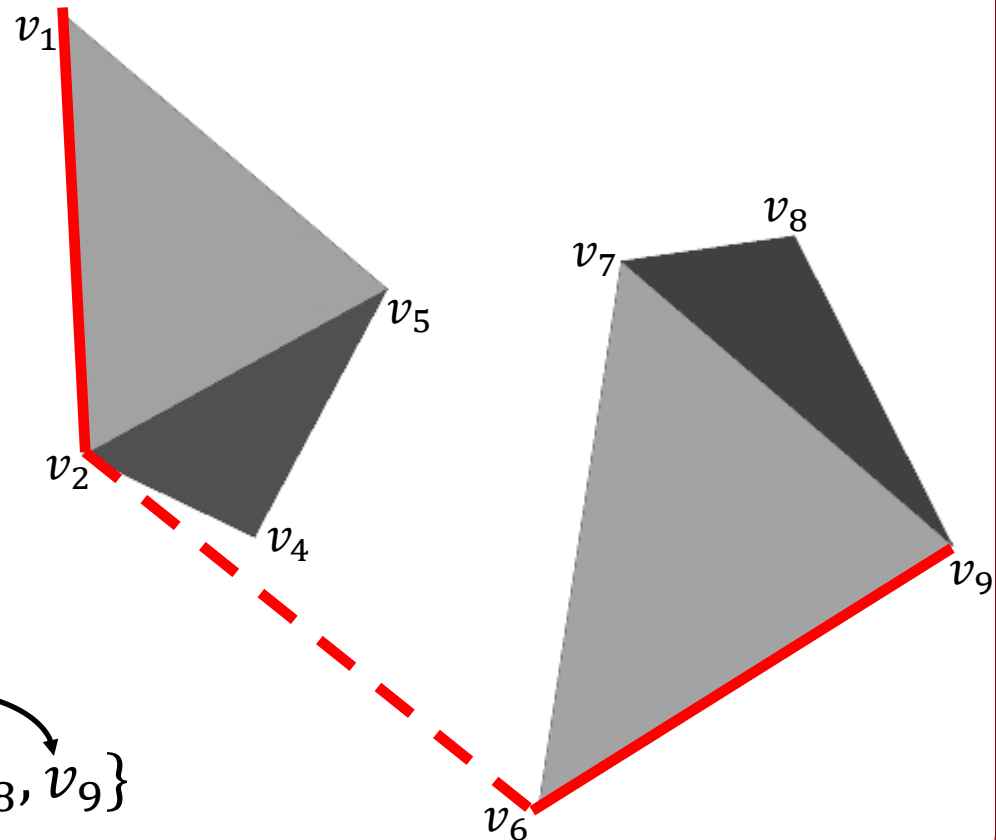
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (120^\circ, 100^\circ)$$

$$R = (\cdot, 150^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \end{array} \right\}$$

$$\alpha = 40^\circ$$



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

Divide-and-Conquer [Chan, 2003]



Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

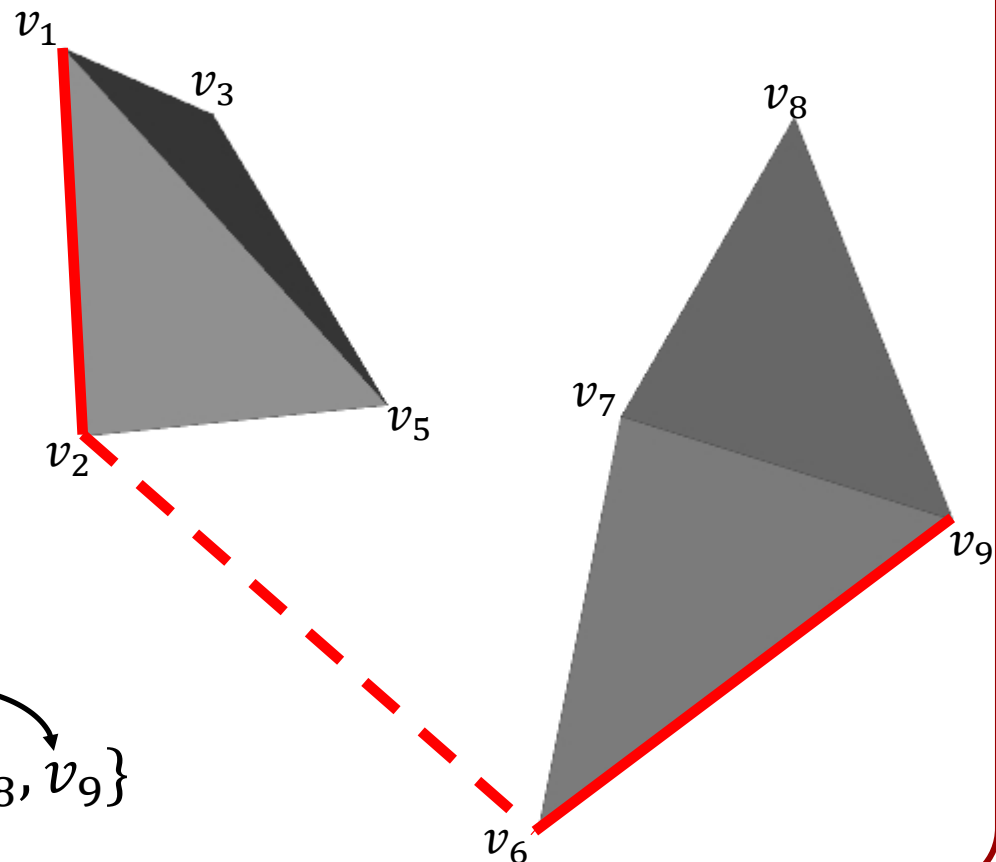
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (120^\circ, 100^\circ)$$

$$R = (\cdot, 150^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \end{array} \right\}$$

$$\alpha = 60^\circ$$



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



Divide-and-Conquer [Chan, 2003]

Advance to the next event:

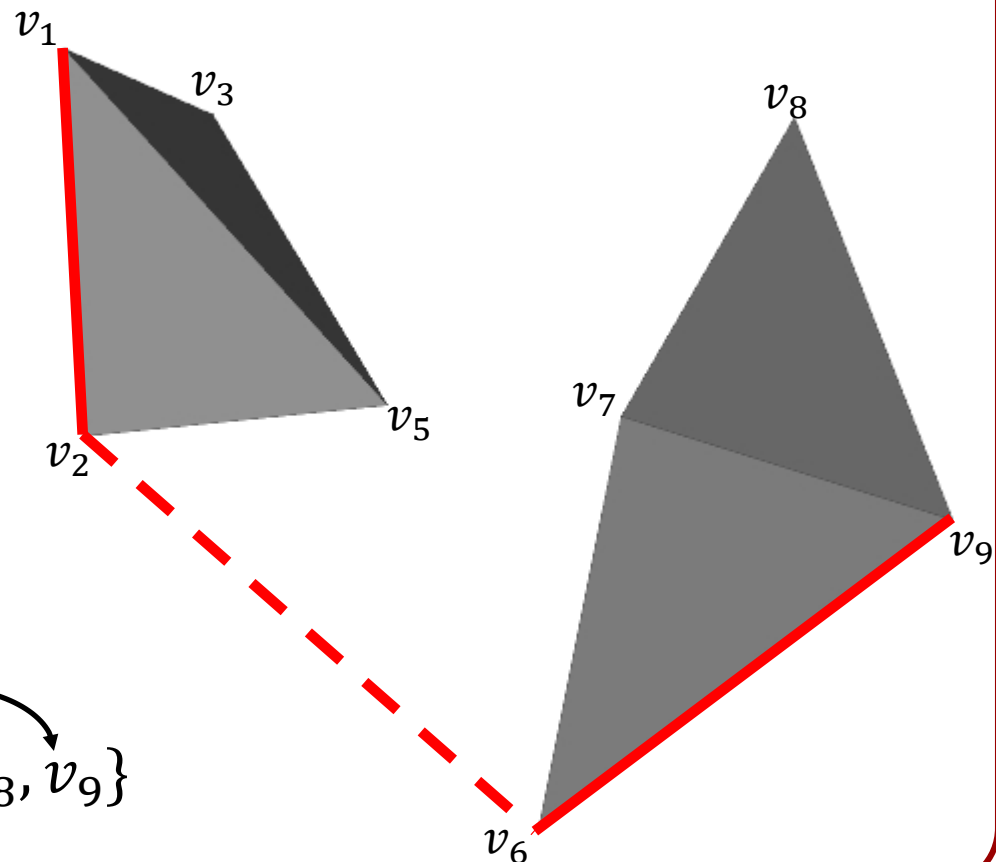
$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (120^\circ, 100^\circ)$$

$$R = (\cdot, 150^\circ)$$

$$\alpha = 60^\circ$$



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

Divide-and-Conquer [Chan, 2003]



Advance to the next event:

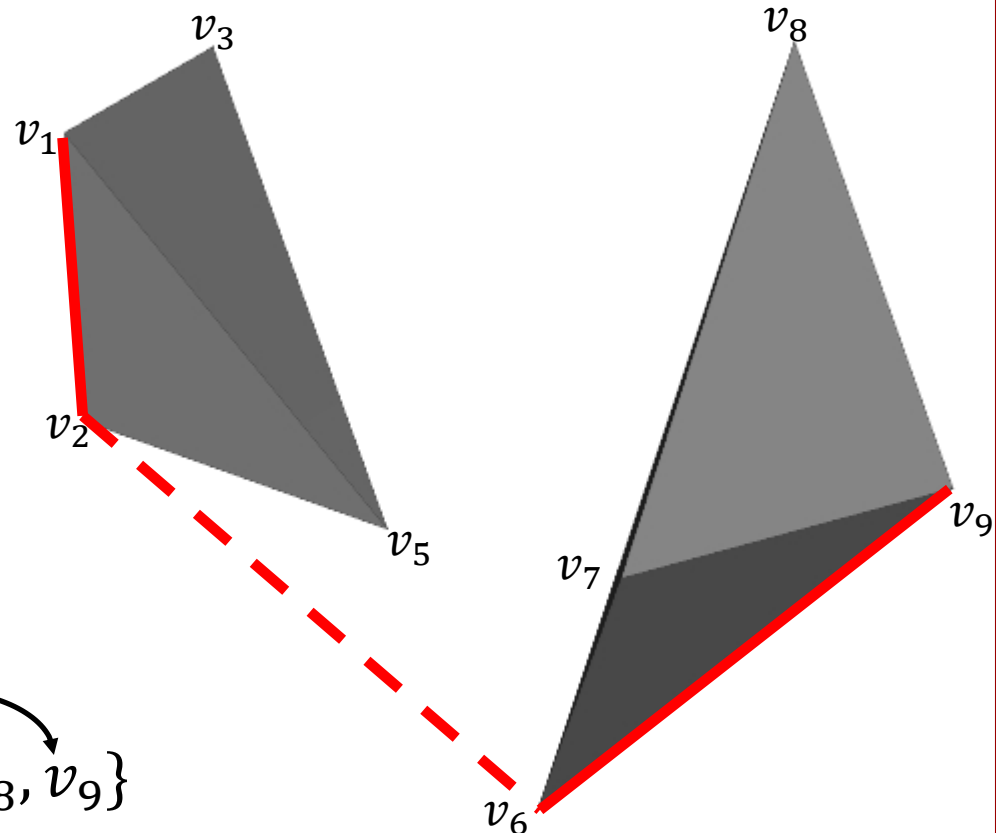
$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (120^\circ, 100^\circ)$$

$$R = (\cdot, 150^\circ)$$

$$\alpha = 80^\circ$$



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

Divide-and-Conquer [Chan, 2003]



Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

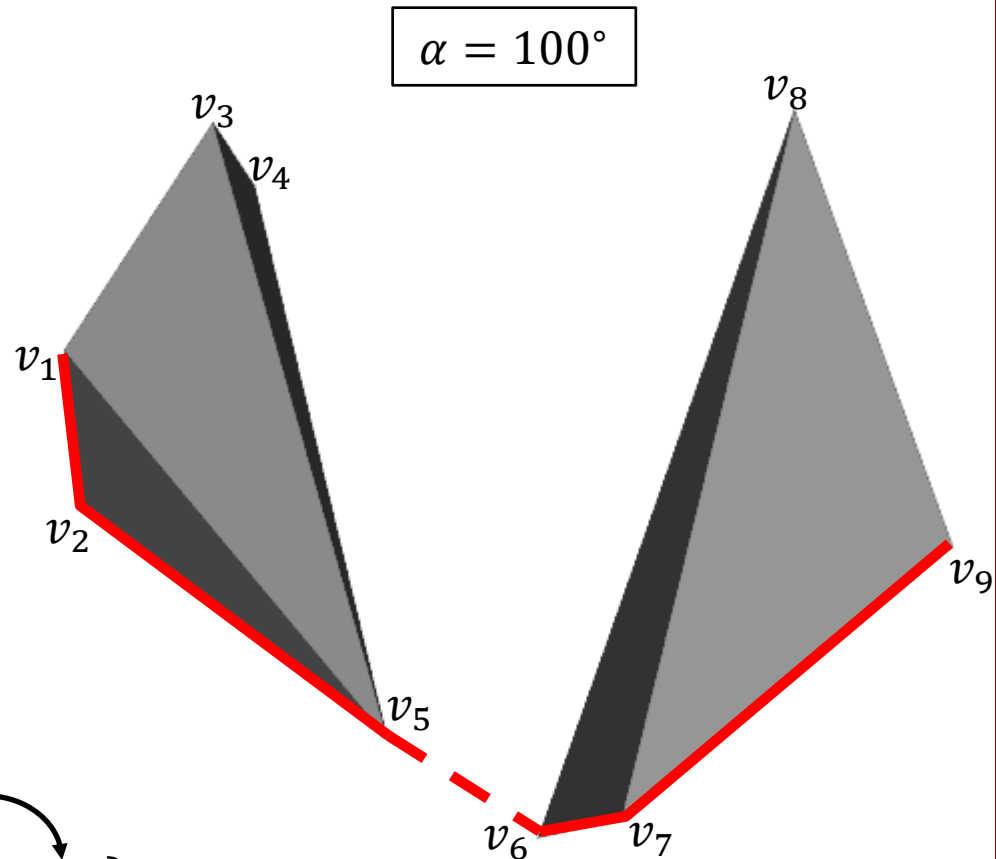
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (120^\circ, 100^\circ)$$

$$R = (\cdot, 150^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \end{array} \right\}$$

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



Divide-and-Conquer [Chan, 2003]



Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

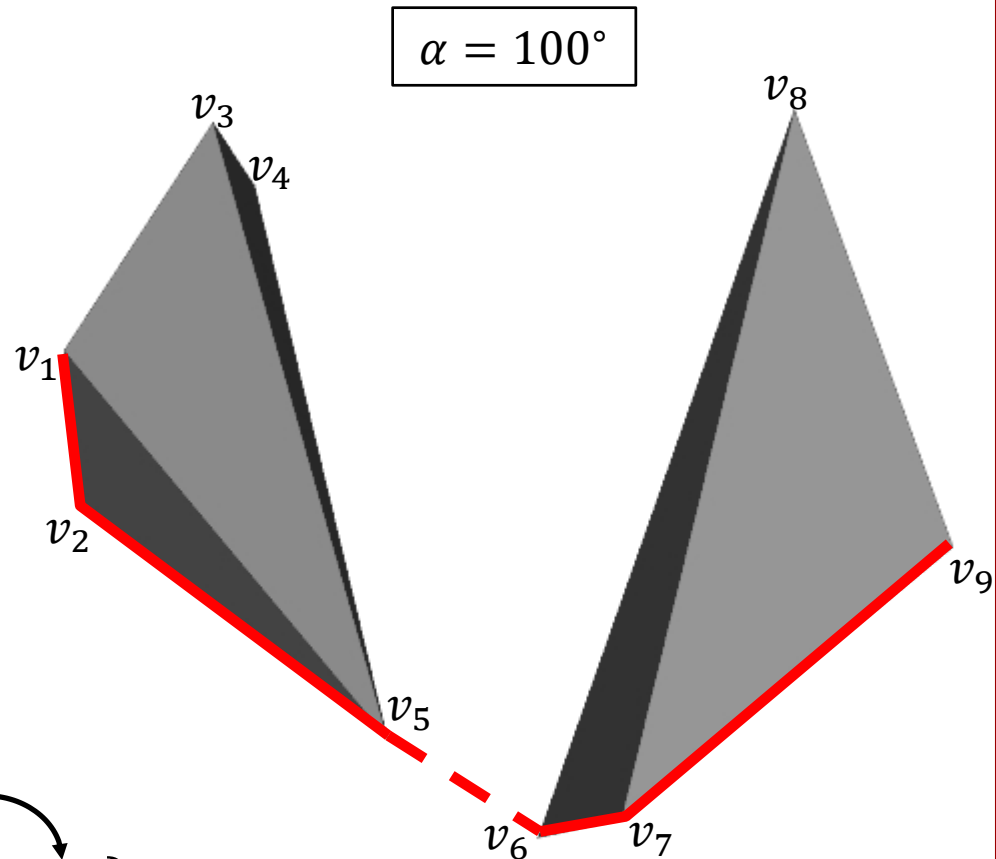
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (120^\circ, \boxed{100^\circ})$$

$$R = (\cdot, 150^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \end{array} \right\}$$

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



Divide-and-Conquer [Chan, 2003]



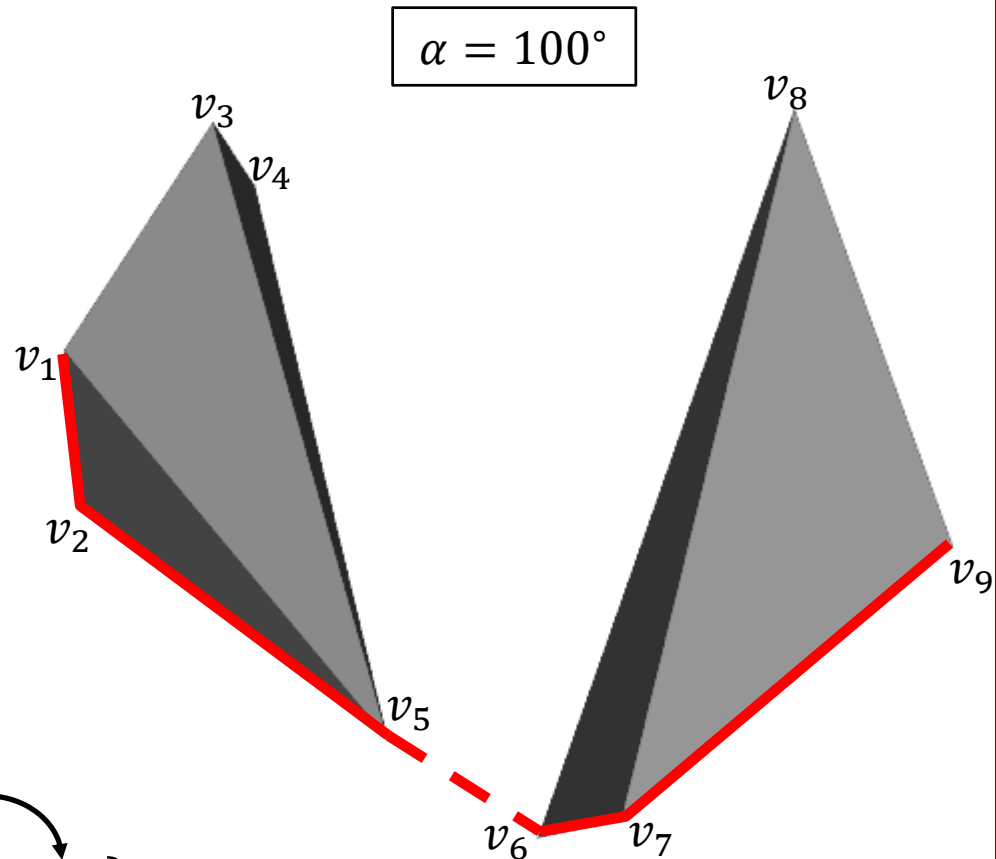
Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (120^\circ, \boxed{100^\circ})$$

$$R = (\cdot, 150^\circ)$$



$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \end{array} \right\}$$

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

Divide-and-Conquer [Chan, 2003]



Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

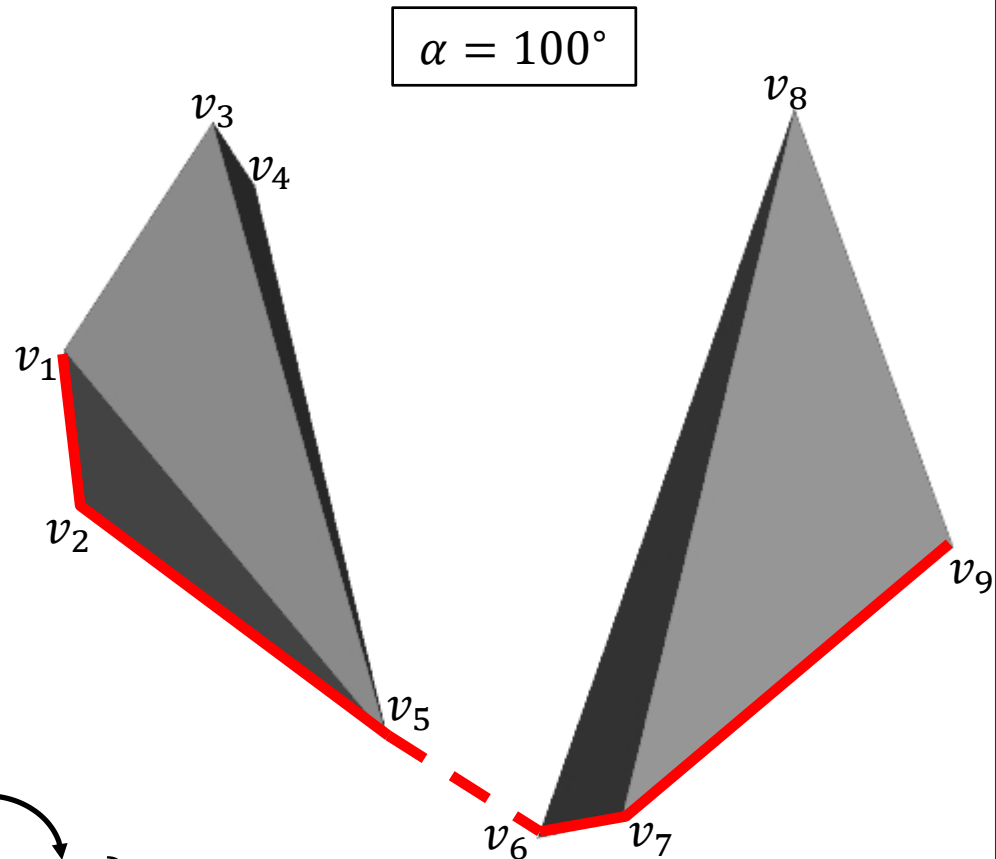
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (180^\circ, \cdot)$$

$$R = (\cdot, \boxed{105^\circ})$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \end{array} \right\}$$

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



Divide-and-Conquer [Chan, 2003]



Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

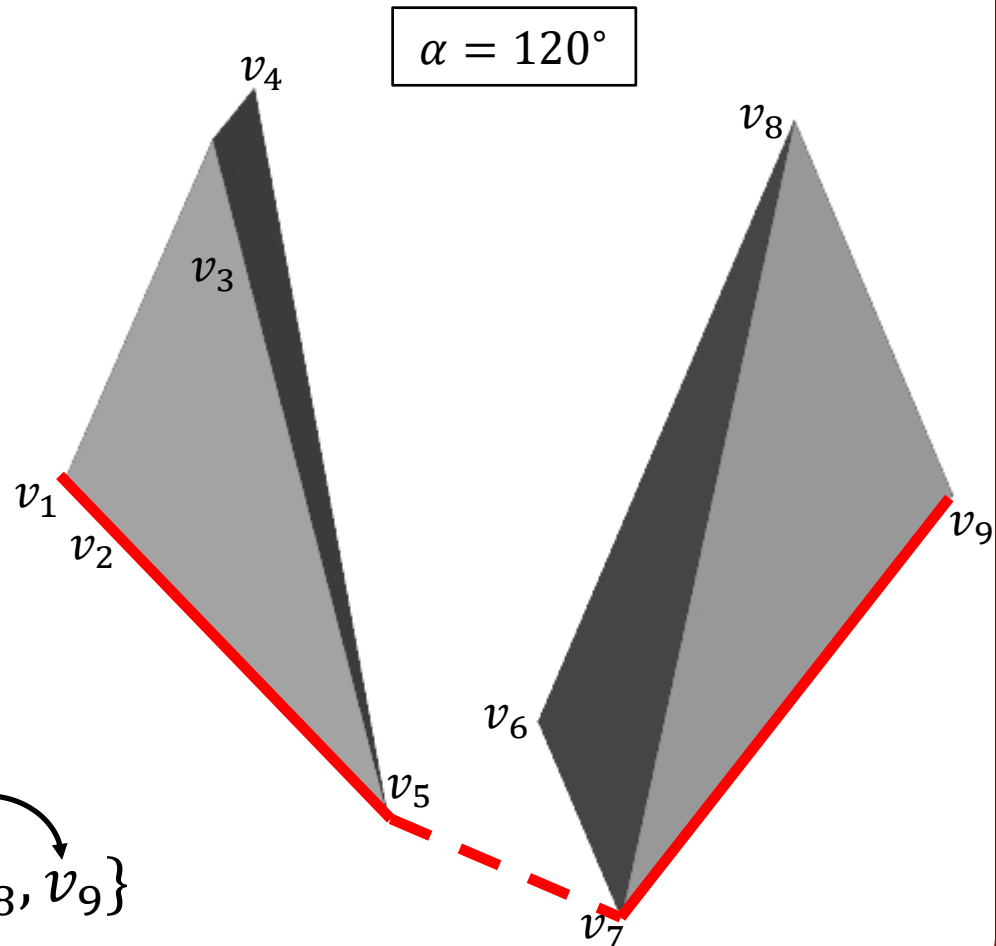
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (180^\circ, \cdot)$$

$$R = (\cdot, \boxed{105^\circ})$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \\ (D, 105^\circ, v_5 v_6 v_7) \end{array} \right\}$$

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





Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

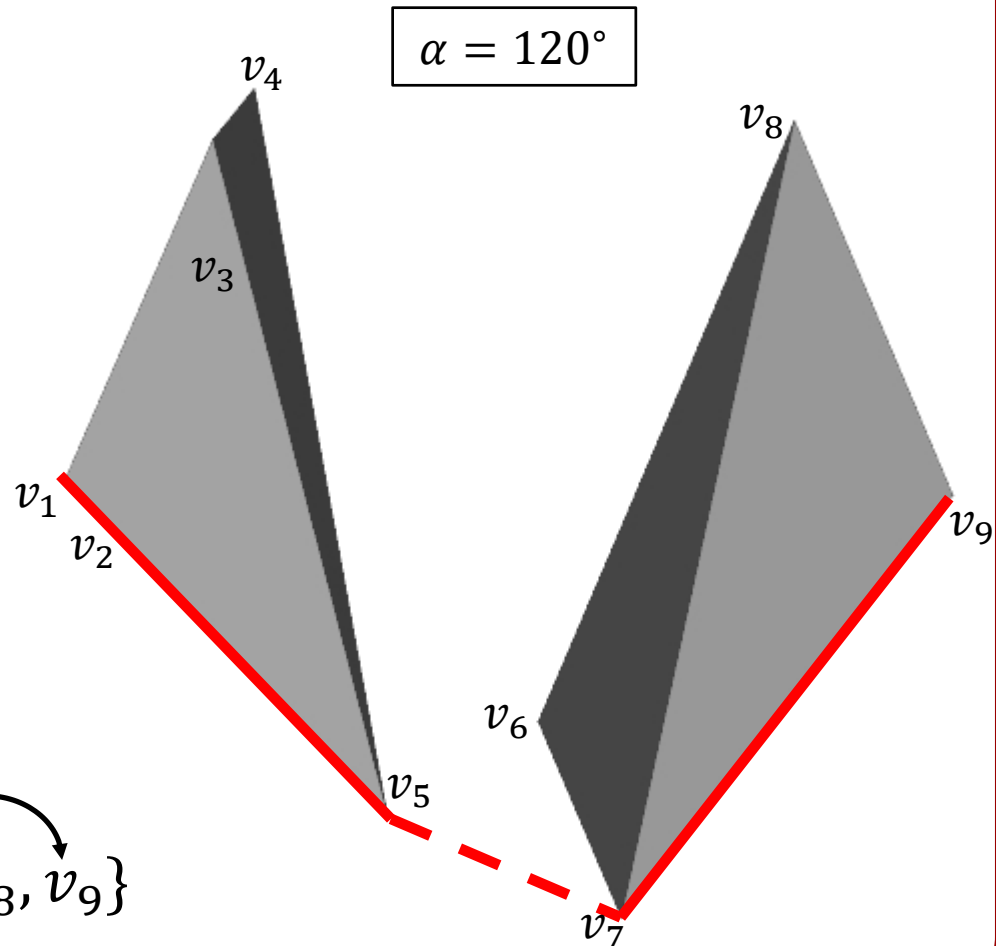
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (155^\circ, \cdot)$$

$$R = (180^\circ, 180^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \\ (D, 105^\circ, v_5 v_6 v_7) \end{array} \right\}$$

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





Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

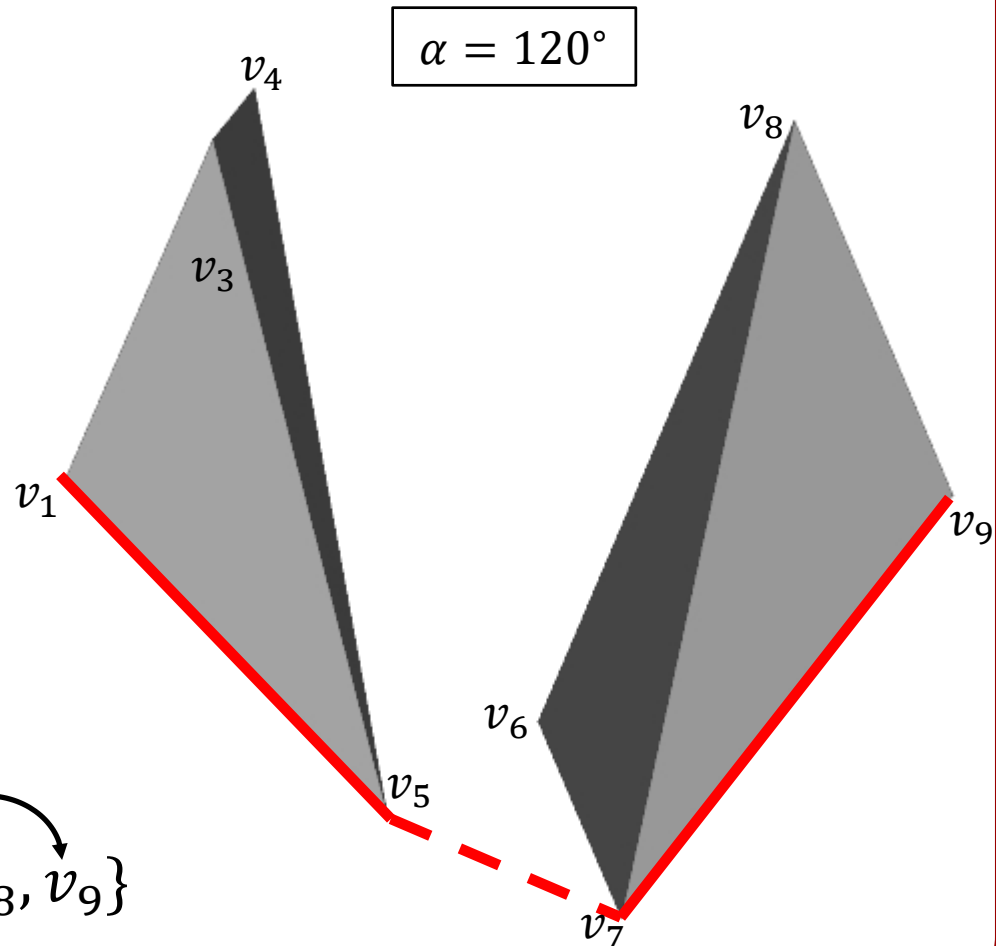
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (155^\circ, \cdot)$$

$$R = (180^\circ, 180^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4 v_6 v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6 v_7 v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \\ (D, 105^\circ, v_5 v_6 v_7) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

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



Divide-and-Conquer [Chan, 2003]



Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

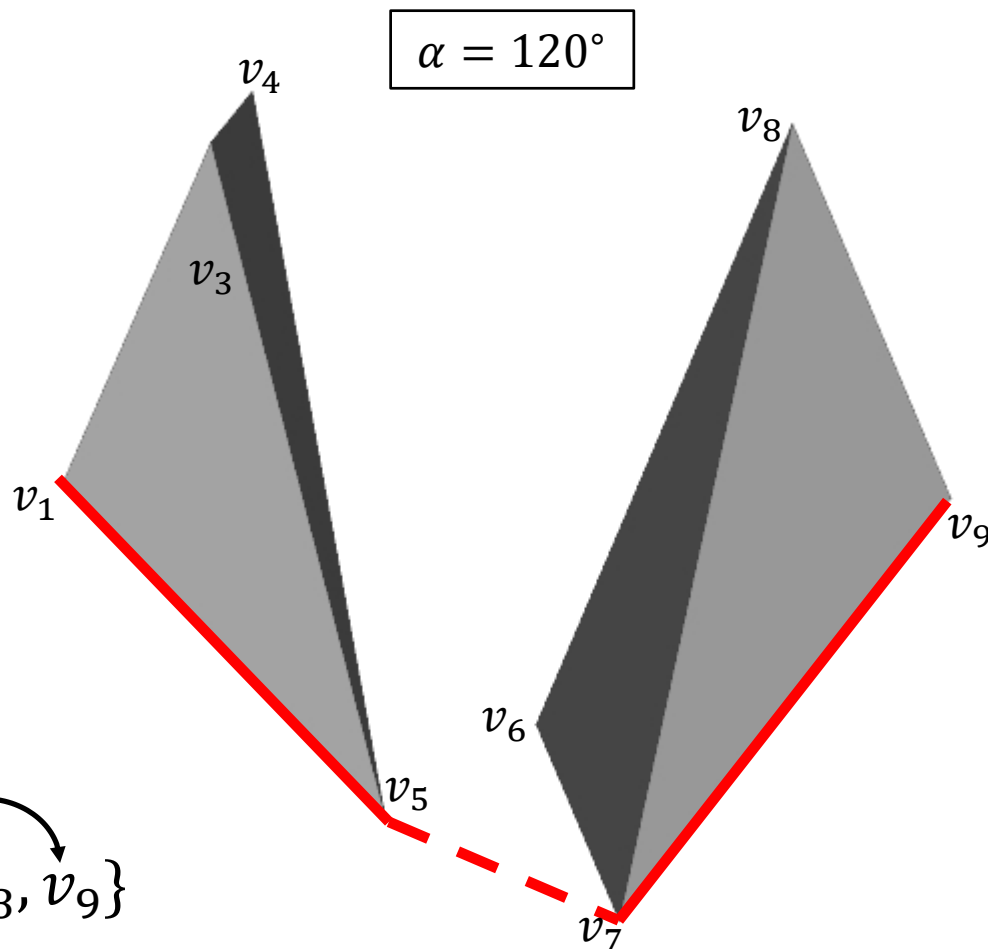
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (155^\circ \quad \cdot \quad)$$

$$R = (180^\circ, 180^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \\ (D, 105^\circ, v_5 v_6 v_7) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

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





Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

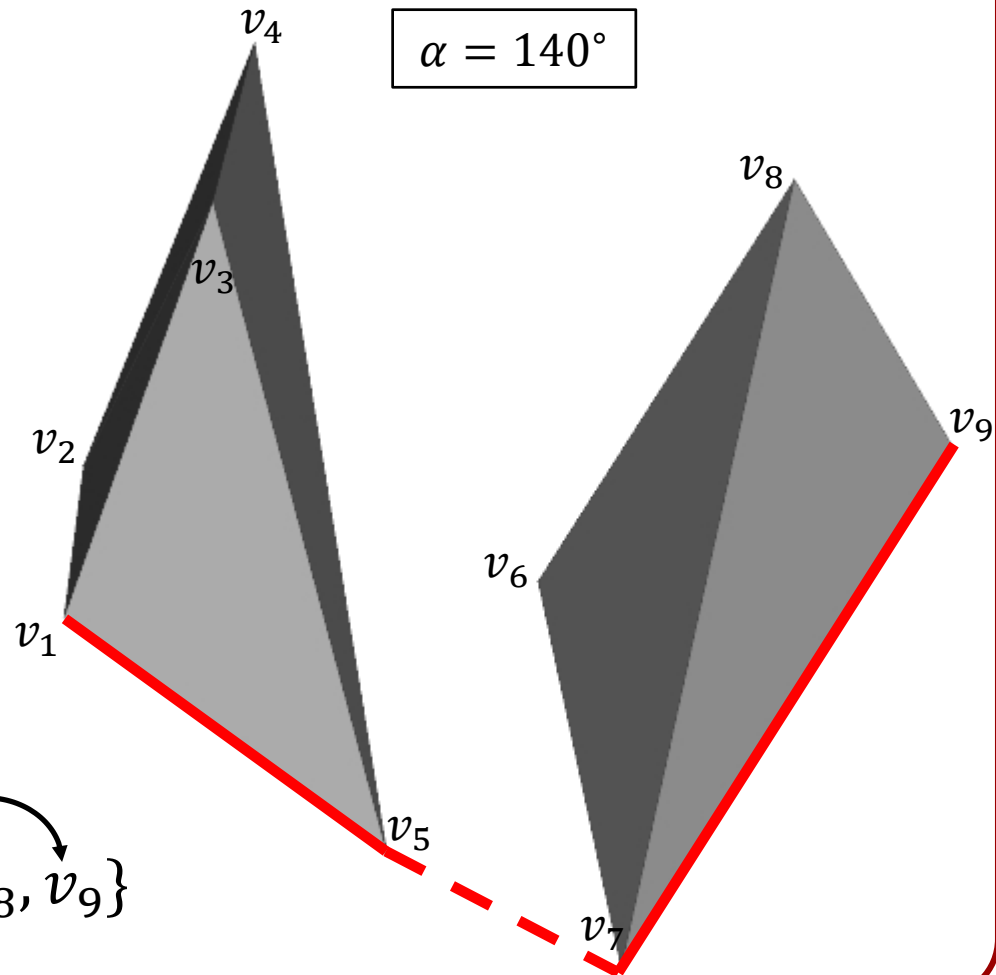
$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$L = (155^\circ \quad \cdot \quad)$$

$$R = (180^\circ, 180^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \\ (D, 105^\circ, v_5 v_6 v_7) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

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





Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

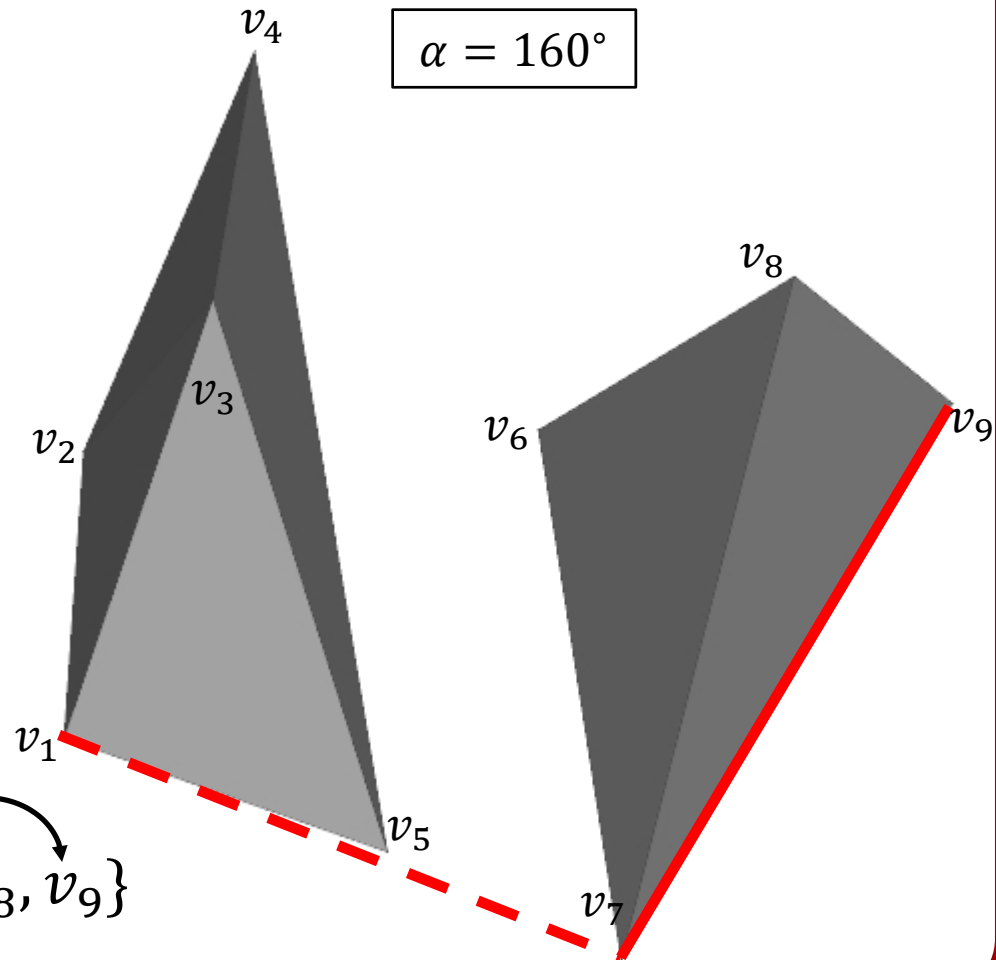
$$L = (155^\circ \cdot)$$

$$R = (180^\circ, 180^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \\ (D, 105^\circ, v_5 v_6 v_7) \\ (D, 110^\circ, v_1 v_2 v_5) \\ (D, 155^\circ, v_1 v_5 v_7) \end{array} \right\}$$

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

$$\alpha = 160^\circ$$





Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

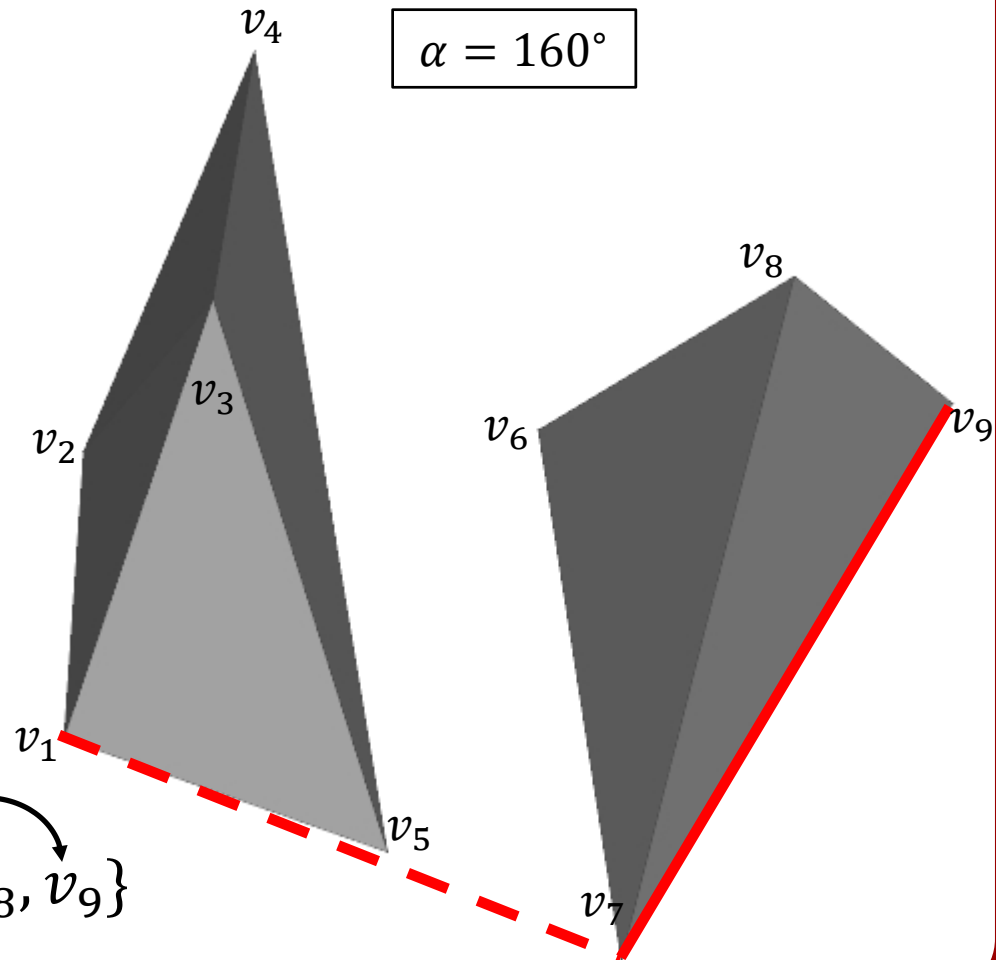
$$L = (180^\circ, 180^\circ)$$

$$R = (180^\circ, 180^\circ)$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \\ (D, 105^\circ, v_5 v_6 v_7) \\ (D, 110^\circ, v_1 v_2 v_5) \\ (D, 155^\circ, v_1 v_5 v_7) \end{array} \right\}$$

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

$$\alpha = 160^\circ$$





Divide-and-Conquer [Chan, 2003]

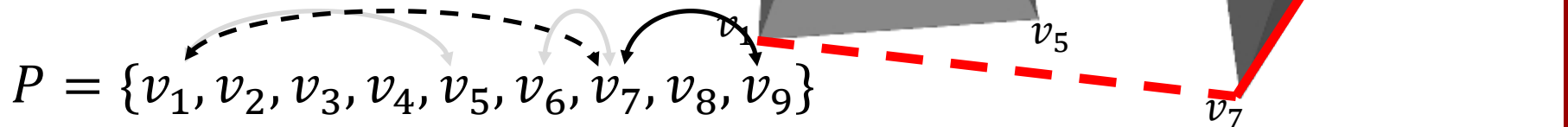
Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

$$\alpha = 180^\circ$$

$$E = \left\{ \begin{array}{l} (I, 10^\circ, v_4, v_6, v_9) \\ (D, 35^\circ, v_2 v_4 v_5) \\ (I, 90^\circ, v_6, v_7, v_9) \\ (I, 100^\circ, v_2 v_5 v_6) \\ (D, 105^\circ, v_5 v_6 v_7) \\ (D, 110^\circ, v_1 v_2 v_5) \\ (D, 155^\circ, v_1 v_5 v_7) \end{array} \right\}$$





Divide-and-Conquer [Chan, 2003]

Advance to the next event:

$$E_L = \left\{ \begin{array}{l} (D, 50^\circ, v_2 v_4 v_5) \\ (D, 110^\circ, v_1 v_2 v_5) \end{array} \right\}$$

$$E_R = \{(I, 90^\circ, v_6 v_7 v_9)\}$$

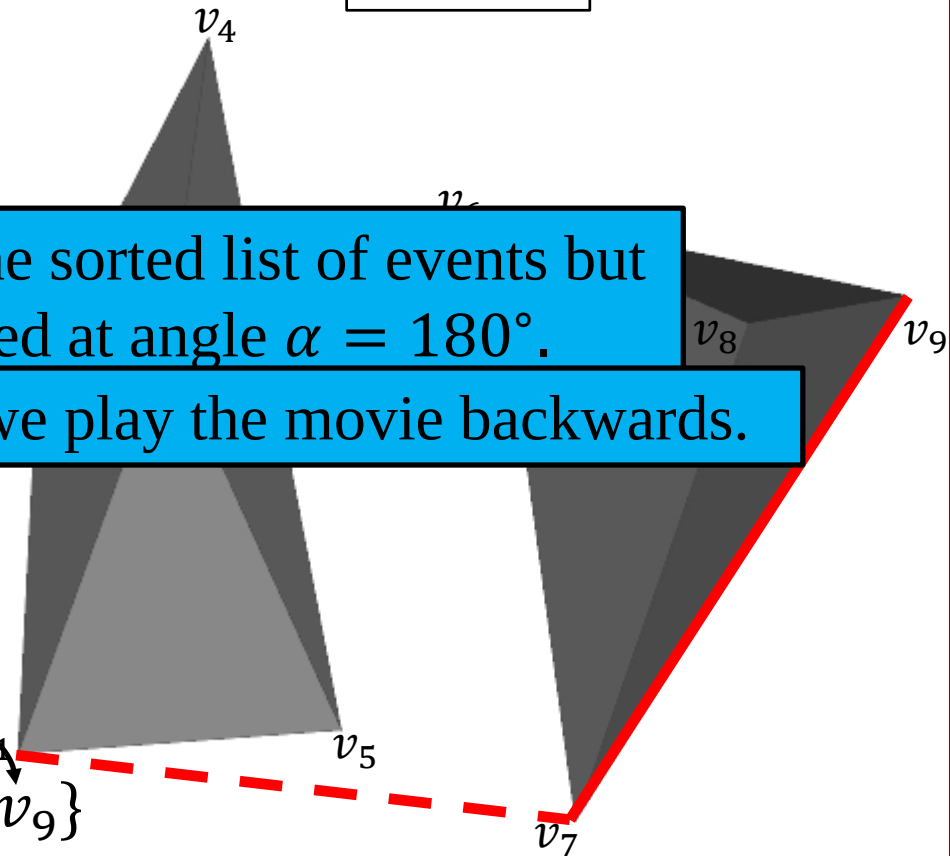
$$\alpha = 180^\circ$$

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

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

$$E = \left\{ \begin{array}{l} (I, 100^\circ, v_2 v_5 v_6) \\ (D, 105^\circ, v_5 v_6 v_7) \\ (D, 110^\circ, v_1 v_2 v_5) \\ (D, 155^\circ, v_1 v_5 v_7) \end{array} \right\}$$

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

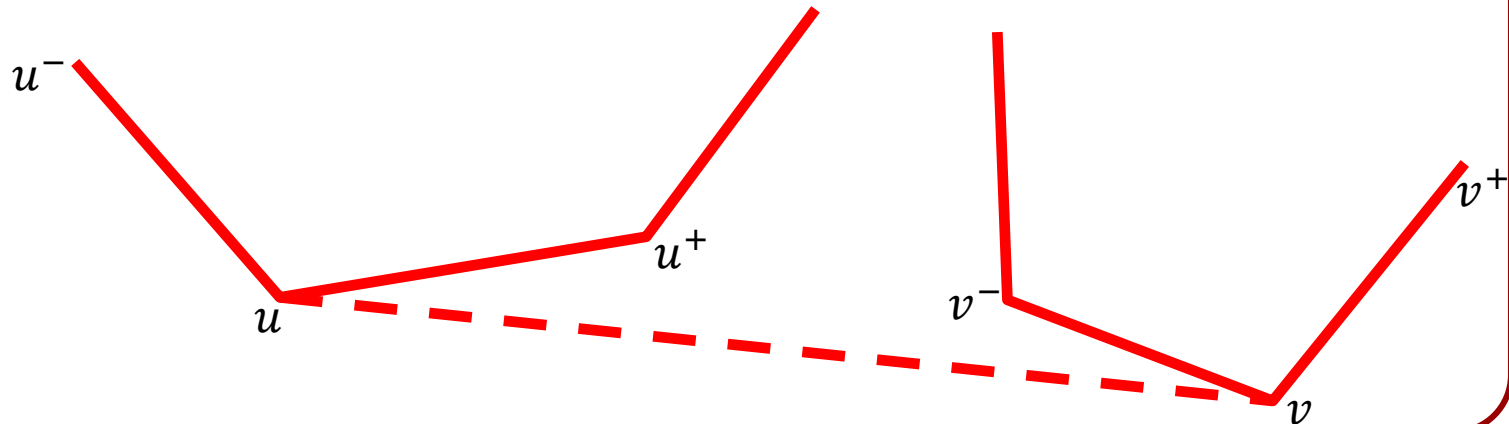


Divide-and-Conquer [Chan, 2003]



Advance to the next event:

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 $\overline{uv}$?



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 $\overline{uv}$?

To find α we need to solve the trigonometric system:

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

with:

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

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 $\overline{uv}$?

Observation 1:

Can parameterize directions with $\alpha \in (-\pi/2, \pi/2)$ by points on the vertical line $(1, \alpha)$ with $\alpha \in (-\infty, \infty)$.

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

$\Downarrow$

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

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 $\overline{uv}$?

Observation 1:

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

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

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

$\Downarrow$

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

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 $\overline{uv}$?

Observation 2:

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

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

$\downarrow$

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

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 $\overline{uv}$?

Observation 2:

Since **This is why the algorithm only sees half the hull.**
transformations, we can scale the y-component.

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

↓

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

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.