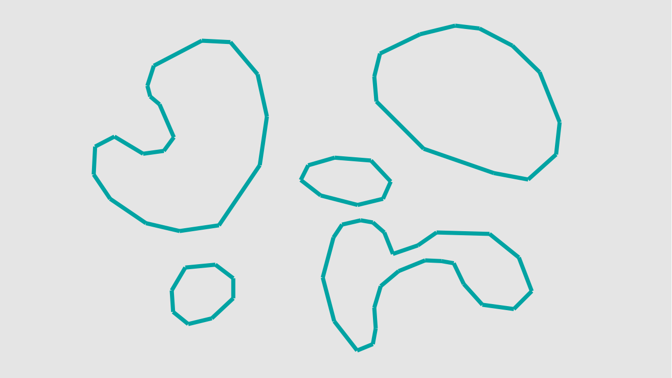
The Possible Hull of Imprecise Points

Jeff Sember
William Evans

University of British Columbia

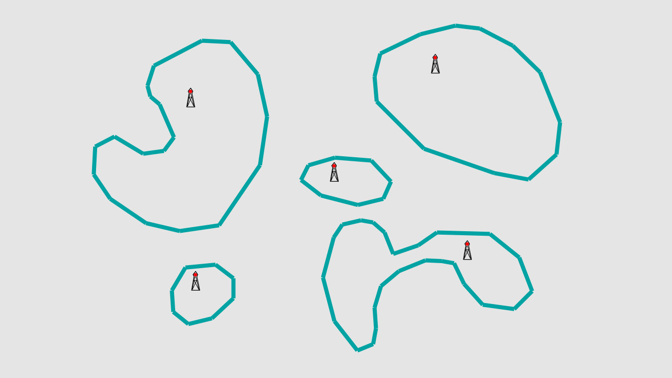
August, 2011

Introduction



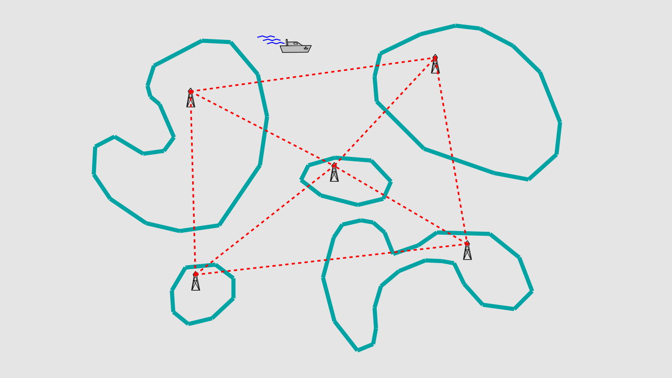
A number of islands

Introduction



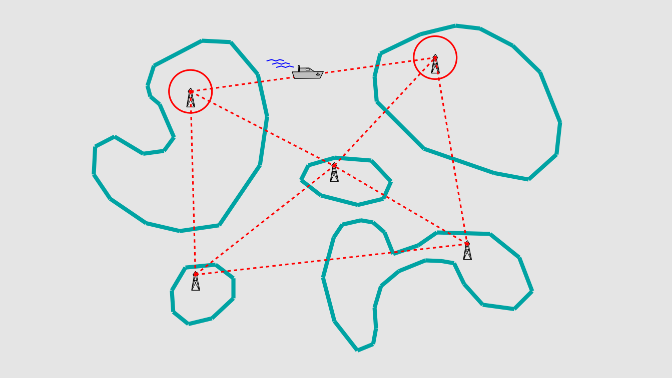
Each island contains a sensor

Introduction



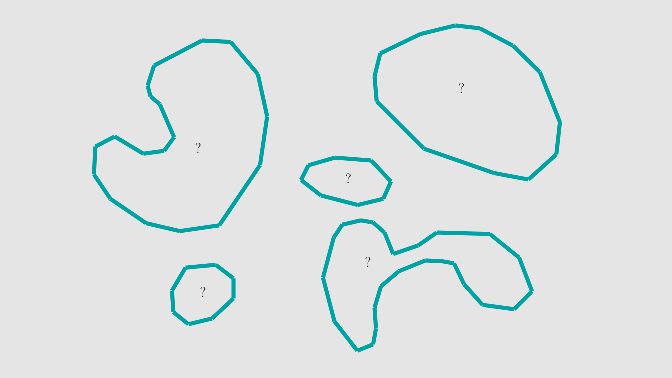
Sensors detect intrusions

Introduction



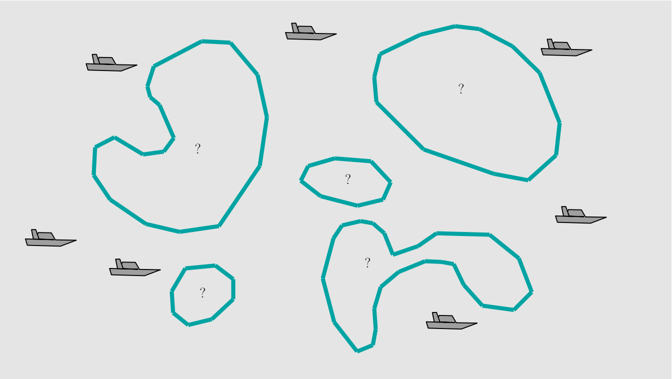
Sensors detect intrusions

Introduction



If precise location of each sensor is not known...

Introduction



...where can a boat safely approach the islands?

Introduction

- Given a planar set of points $S = \{s_1, \ldots, s_n\}$

Introduction

- Given a planar set of points $S = \{s_1, \ldots, s_n\}$
- Each point s_i not known precisely

Introduction

- Given a planar set of points $S = \{s_1, \ldots, s_n\}$
- Each point s_i not known precisely
- ...but known to lie within a region of uncertainty $R_i \in \mathcal{R}$

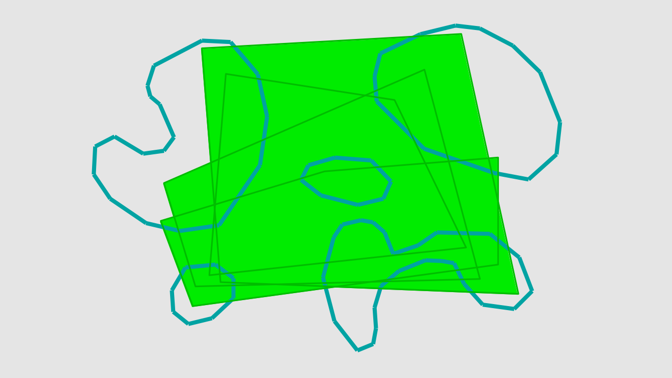
Introduction

- Given a planar set of points $S = \{s_1, \ldots, s_n\}$
- Each point s_i not known precisely
- ...but known to lie within a region of uncertainty $R_i \in \mathcal{R}$
- What parts of plane might lie within convex hull $\mathrm{CH}(S)$?

Introduction

- Given a planar set of points $S = \{s_1, \ldots, s_n\}$
- Each point s_i not known precisely
- ...but known to lie within a region of uncertainty $R_i \in \mathcal{R}$
- What parts of plane might lie within convex hull $\mathrm{CH}(S)$?
- This is $PH(\mathcal{R})$, the possible hull of $\mathcal{R}$

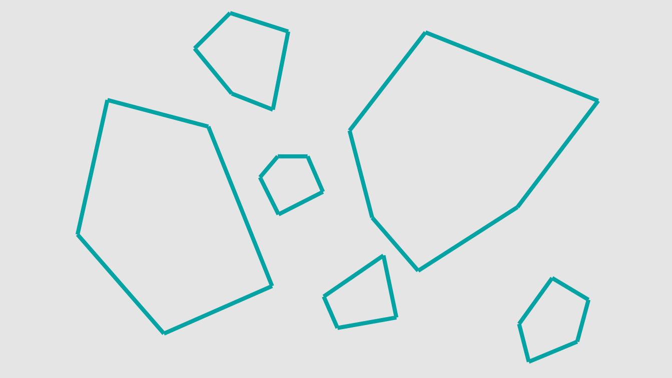
Introduction



The possible hull is the union of (infinitely many) feasible convex hulls:

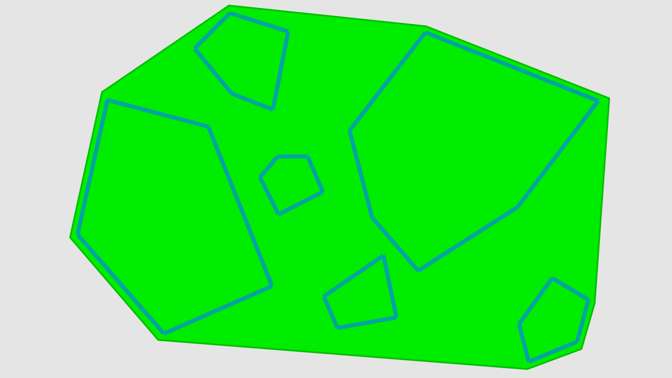
$$PH(\mathcal{R}) = \bigcup_{\{s_1 \in R_1, \ldots, s_n \in R_n\}} \mathrm{CH}(\{s_1, \ldots, s_n\})$$

Introduction



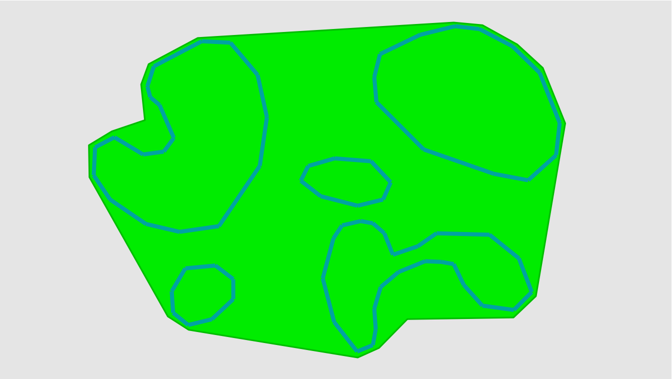
Possible hull of convex uncertain regions...

Introduction



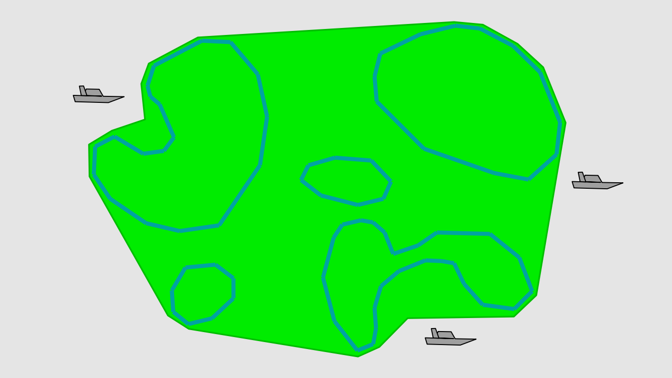
...is simply the hull of the regions: $PH(\mathcal{R}) = \mathrm{CH}(\mathcal{R})$ [Nagai et al., 2000]

Introduction



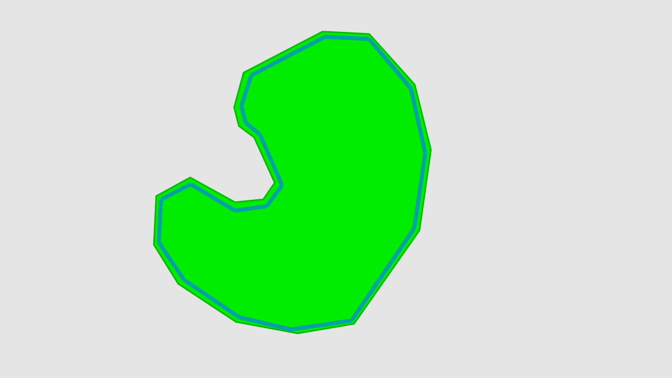
This is not true for nonconvex uncertain regions

Introduction



In some areas, boat can safely approach closer to islands

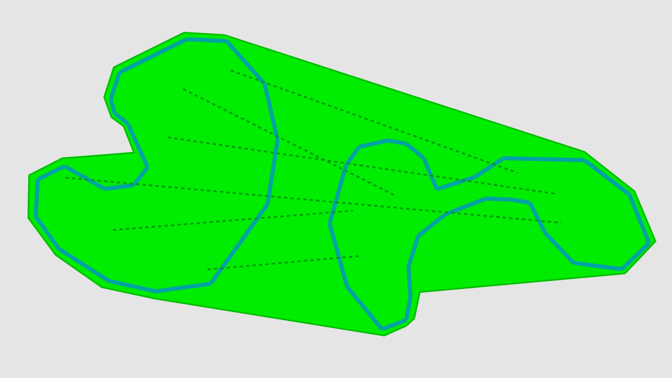
Properties



Lemma 1

Possible hull of a single region is equal to the region

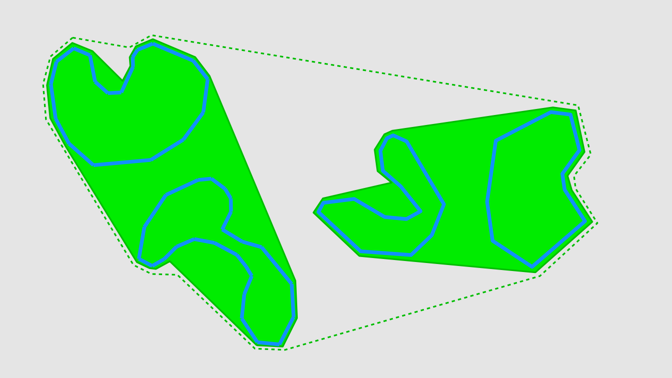
Properties



Lemma 2

Possible hull of pair of regions is union of line segments

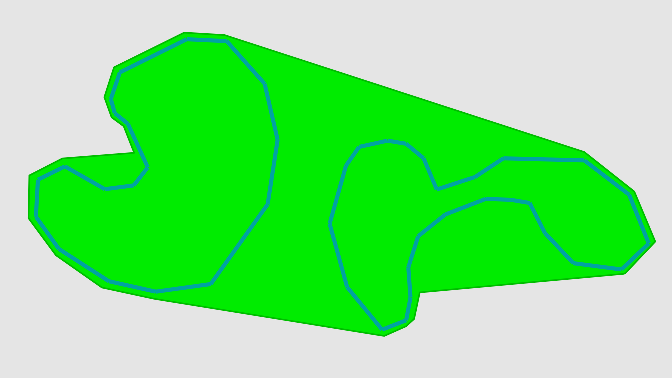
Properties



Lemma 4

Possible hull of regions obtained recursively by partitioning regions into two sets, constructing possible hull of each set, and constructing possible hull of possible hulls

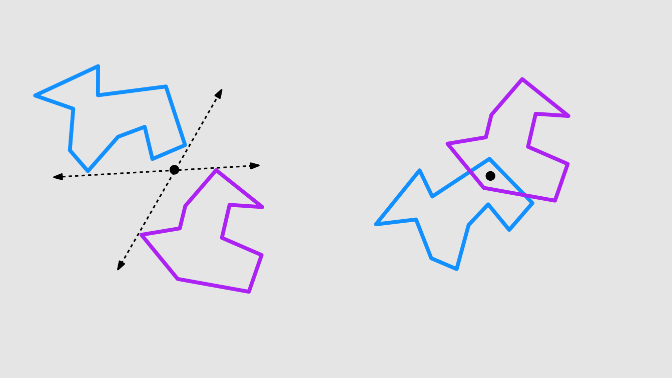
Properties



Lemma 5

Possible hull of (≥ 2) regions is simply connected (no holes)

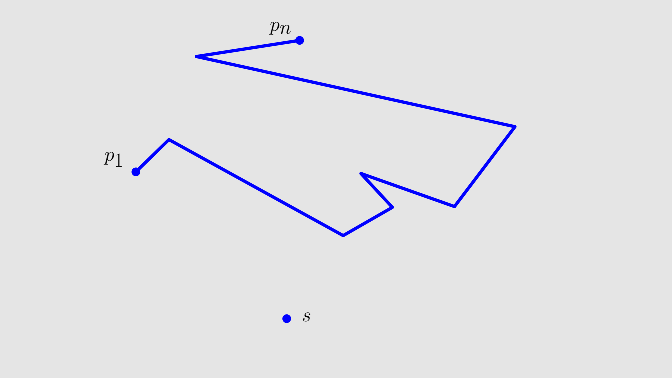
Properties



Theorem 7

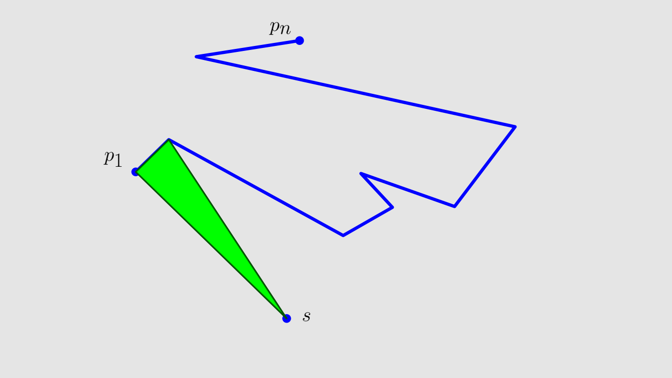
The possible hull of any two or more connected uncertain regions is star-shaped

Point and Polygon



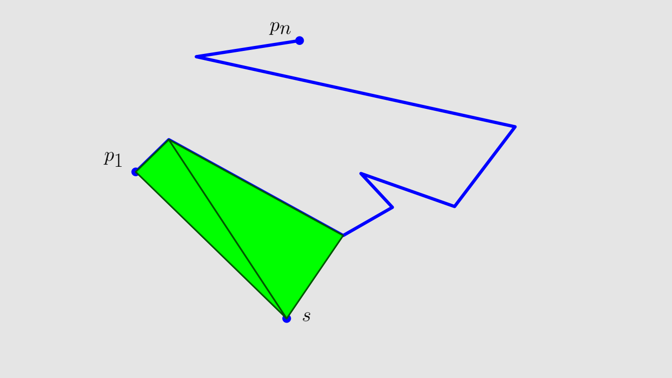
Possible hull of point and polygonal chain

Point and Polygon



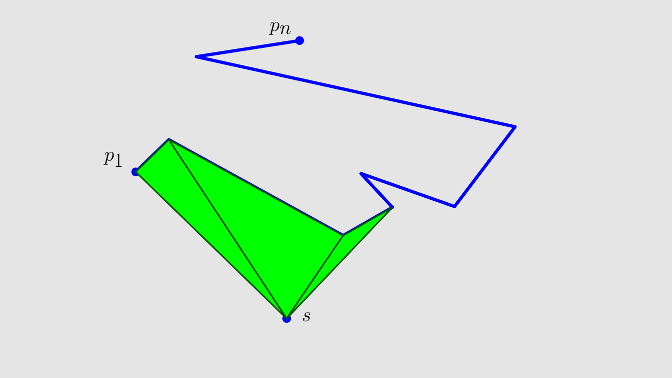
...a union of triangles

Point and Polygon



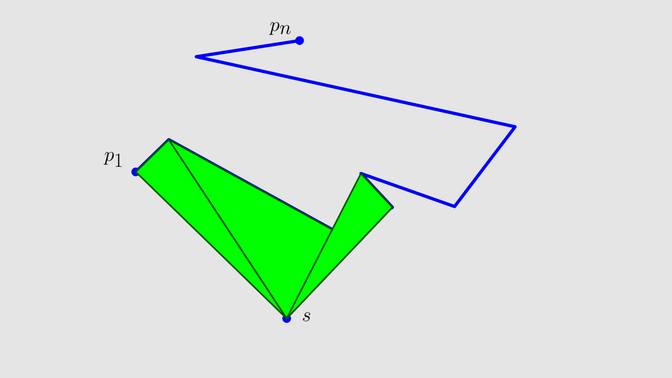
...a union of triangles

Point and Polygon



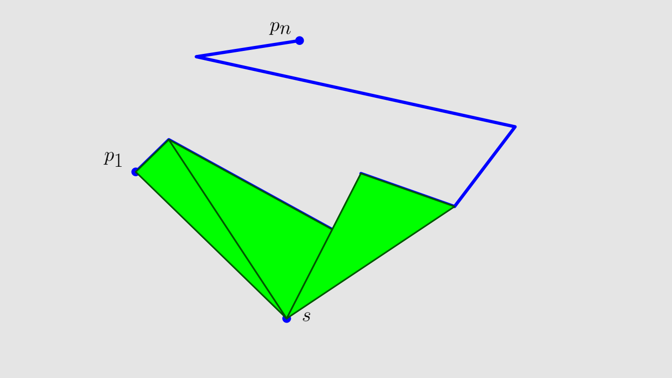
...a union of triangles

Point and Polygon



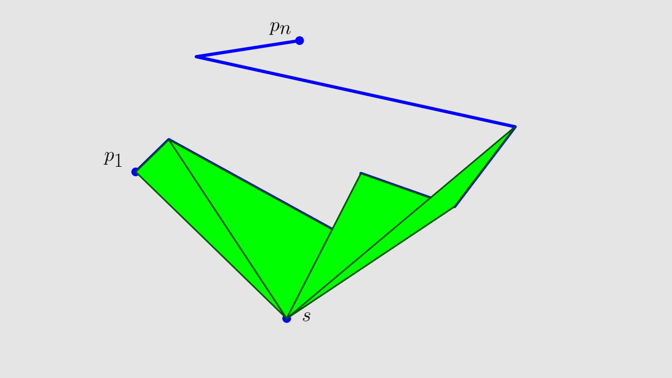
...a union of triangles

Point and Polygon



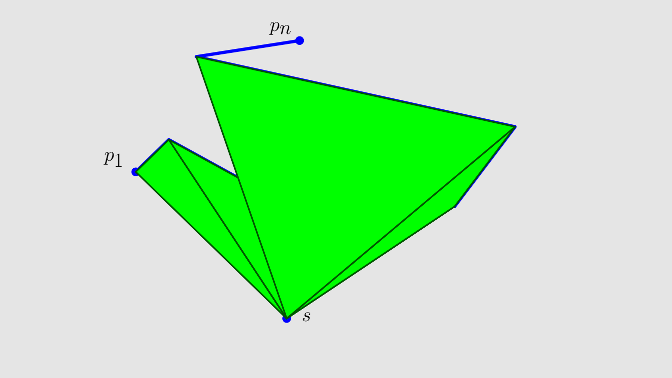
...a union of triangles

Point and Polygon



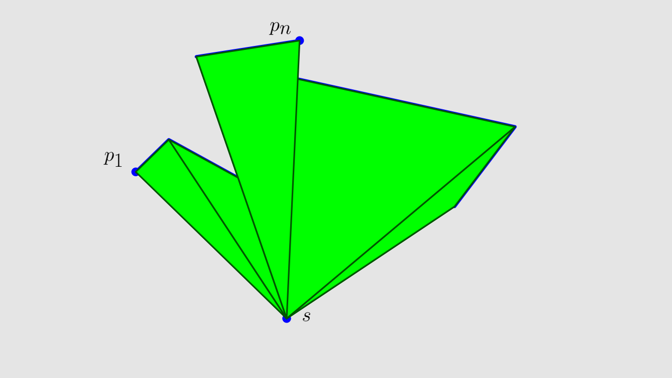
...a union of triangles

Point and Polygon



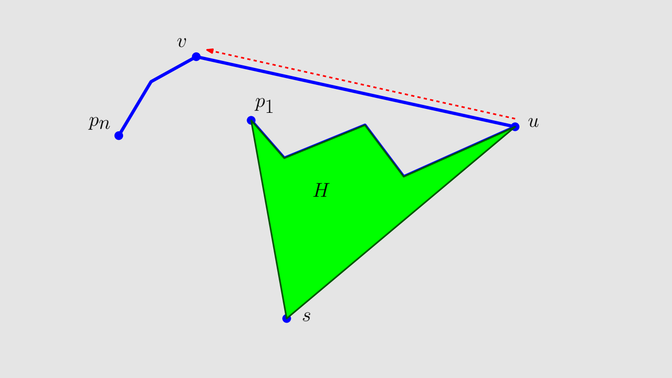
...a union of triangles

Point and Polygon



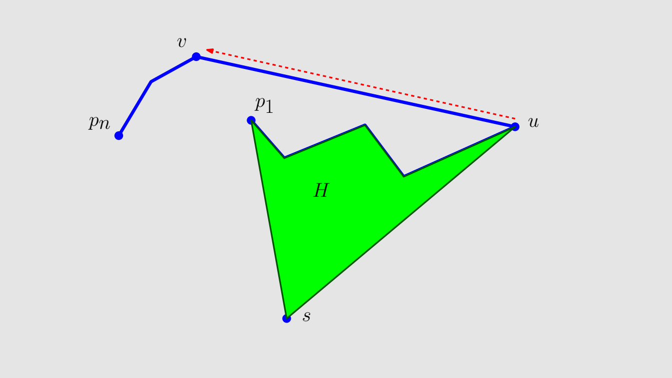
...a union of triangles

Point and Polygon



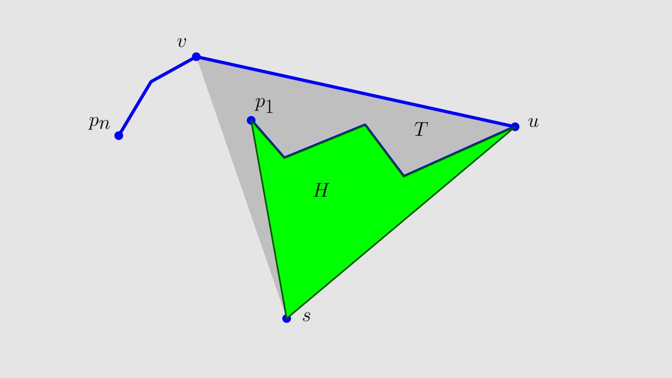
Each step is either expansion or interior step

Point and Polygon: Expansion Step



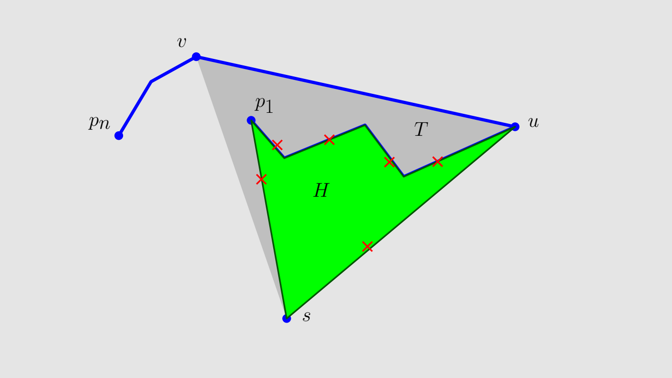
Expansion step: next chain edge lies outside of hull

Point and Polygon: Expansion Step



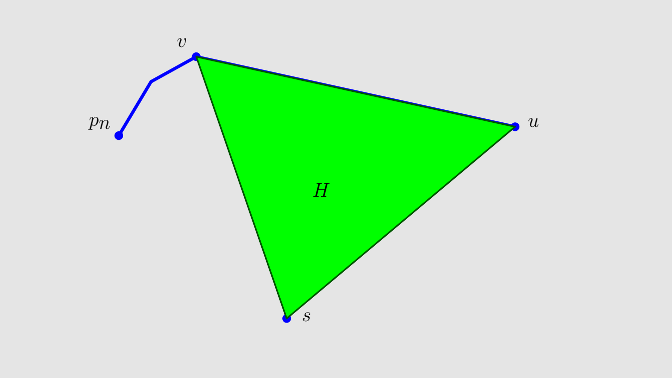
Construct triangle for next chain edge

Point and Polygon: Expansion Step



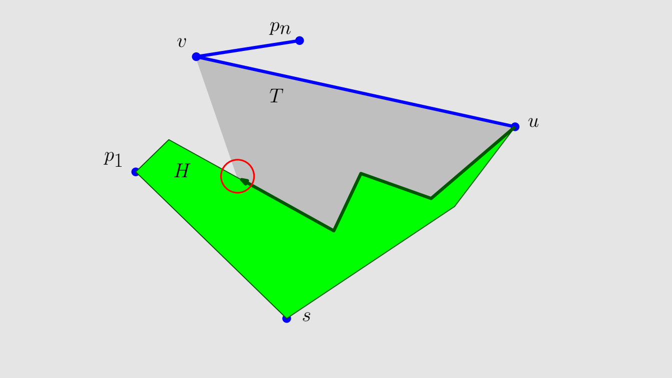
Case (i): hull lies within triangle

Point and Polygon: Expansion Step



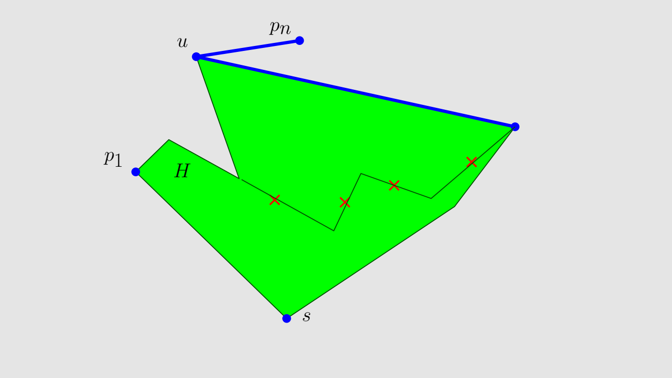
Case (i): hull lies within triangle

Point and Polygon: Expansion Step



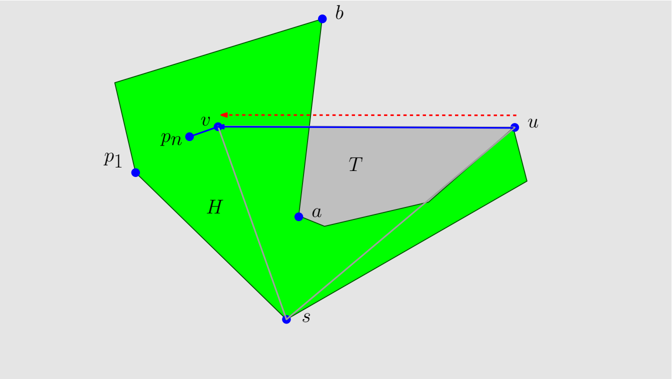
Case (ii): hull edge exits through side of triangle

Point and Polygon: Expansion Step



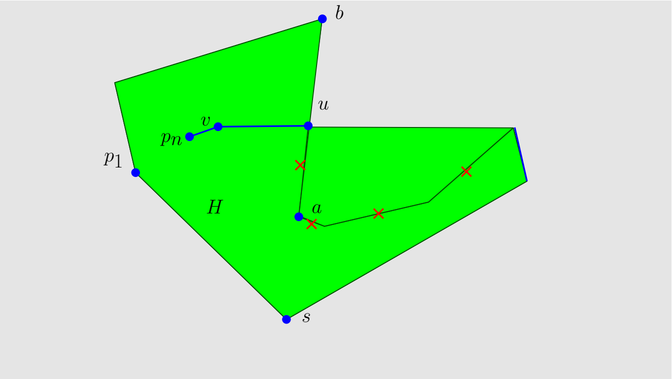
Delete edges interior to triangle

Point and Polygon: Expansion Step



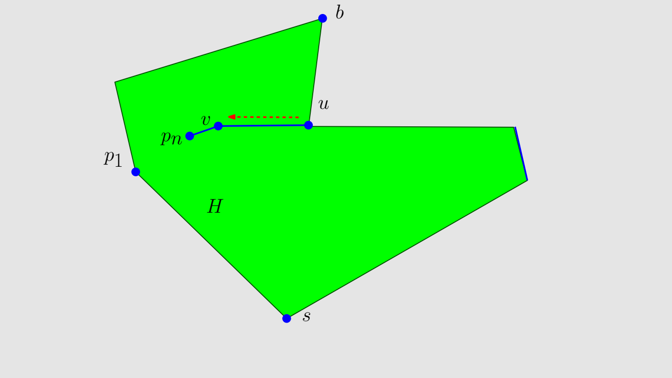
Case (iii): hull edge exits through top of triangle

Point and Polygon: Expansion Step



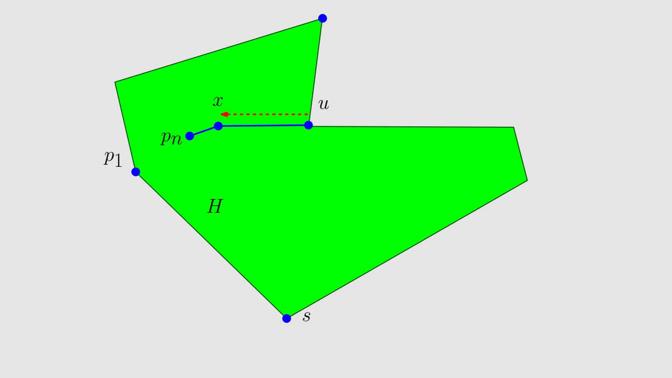
Delete edges interior to triangle

Point and Polygon: Expansion Step



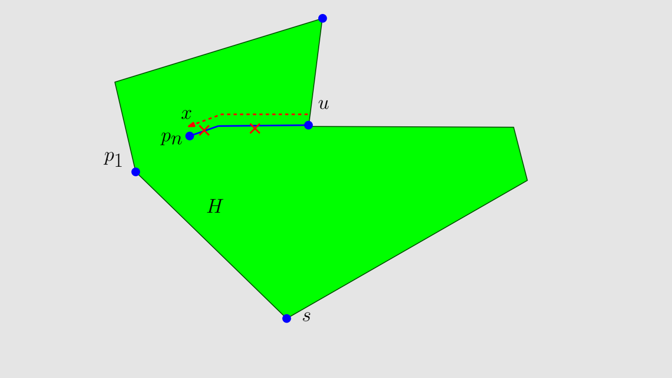
Next iteration will be interior step

Point and Polygon: Interior Step



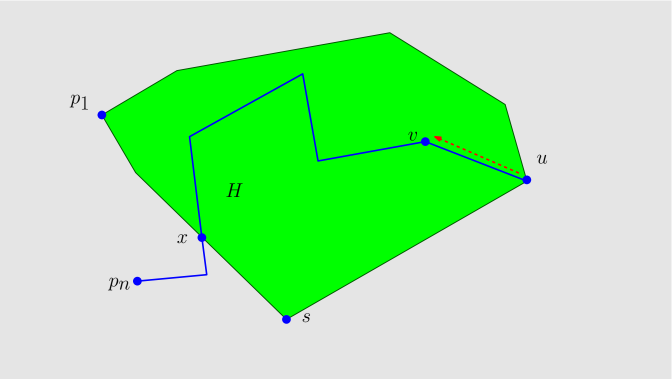
Interior step: advance along chain until one of two events

Point and Polygon: Interior Step



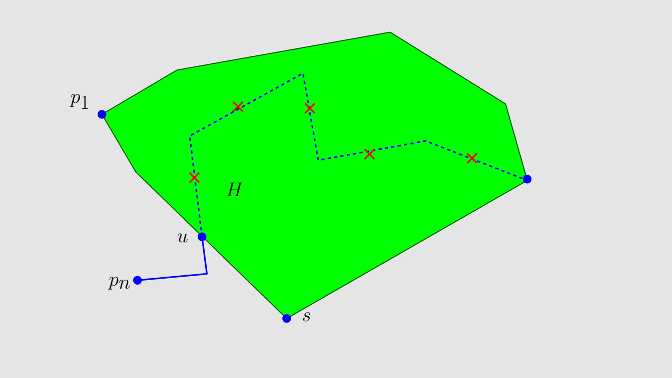
Case (i): end of chain reached; stop

Point and Polygon: Interior Step



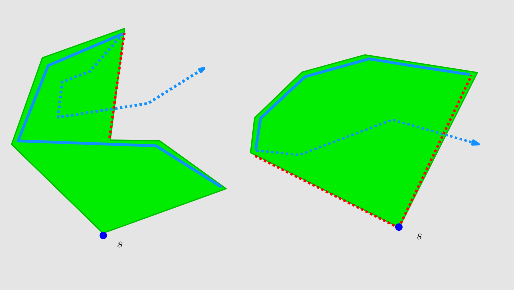
Case (ii): Chain emerges from hull

Point and Polygon: Interior Step



Next iteration is expansion step

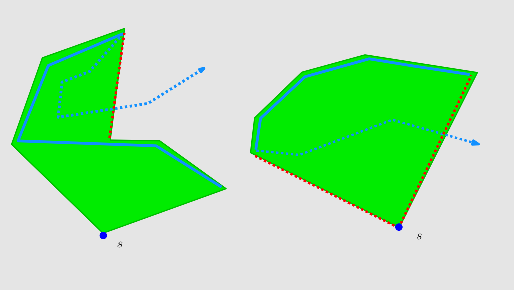
Point and Polygon: Interior Step



Lemma 10

The exit edge is the same as the entrance edge, if that edge is not incident with s; otherwise, it lies on one of the two edges incident with s.

Point and Polygon: Interior Step

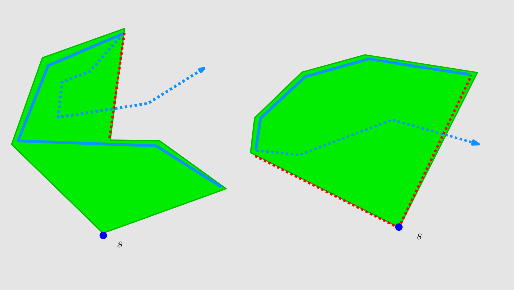


Lemma 10

The exit edge is the same as the entrance edge, if that edge is not incident with s; otherwise, it lies on one of the two edges incident with s.

- Each chain edge need be tested for crossing at most two hull edges

Point and Polygon: Interior Step



Lemma 10

The exit edge is the same as the entrance edge, if that edge is not incident with s; otherwise, it lies on one of the two edges incident with s.

- Each chain edge need be tested for crossing at most two hull edges
- $\Longrightarrow$ interior step runs in time linear in number of edges of chain processed

Point and Polygon: Running Time

- Each step takes $O(1)$ time; running time of algorithm is bounded by vertices processed

Point and Polygon: Running Time

- Each step takes $O(1)$ time; running time of algorithm is bounded by vertices processed
- These include vertices of chain, plus any vertices added to (dynamic) hull

Point and Polygon: Running Time

- Each step takes $O(1)$ time; running time of algorithm is bounded by vertices processed
- These include vertices of chain, plus any vertices added to (dynamic) hull
- Chain has $O(n)$ vertices

Point and Polygon: Running Time

- Each step takes $O(1)$ time; running time of algorithm is bounded by vertices processed
- These include vertices of chain, plus any vertices added to (dynamic) hull
- Chain has $O(n)$ vertices
- At most three vertices added to hull by addition of chain edge triangle

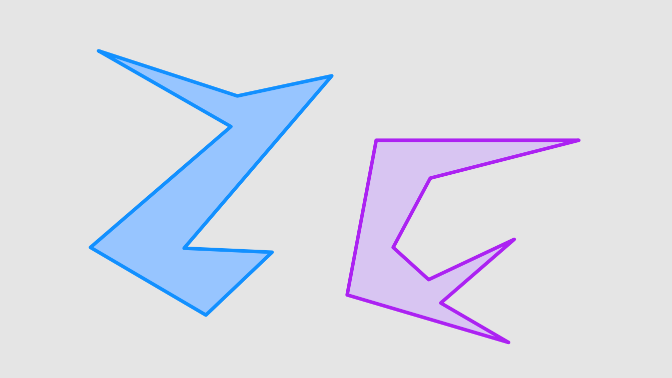
Point and Polygon: Running Time

- Each step takes $O(1)$ time; running time of algorithm is bounded by vertices processed
- These include vertices of chain, plus any vertices added to (dynamic) hull
- Chain has $O(n)$ vertices
- At most three vertices added to hull by addition of chain edge triangle

Theorem 11

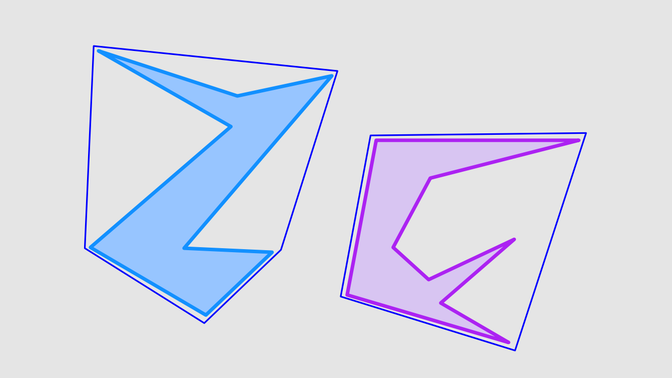
Algorithm runs in linear time

Possible Hull of Pair of Polygons



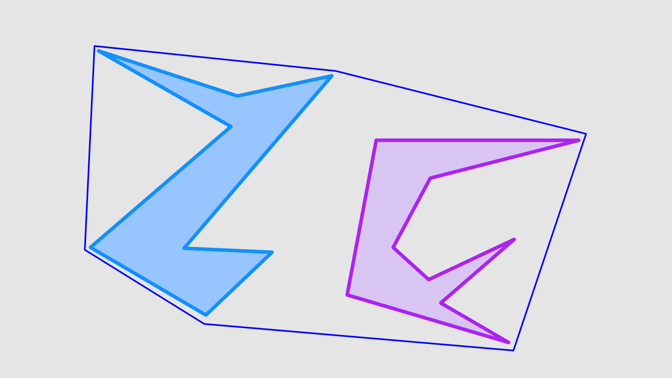
Start with two polygons

Possible Hull of Pair of Polygons



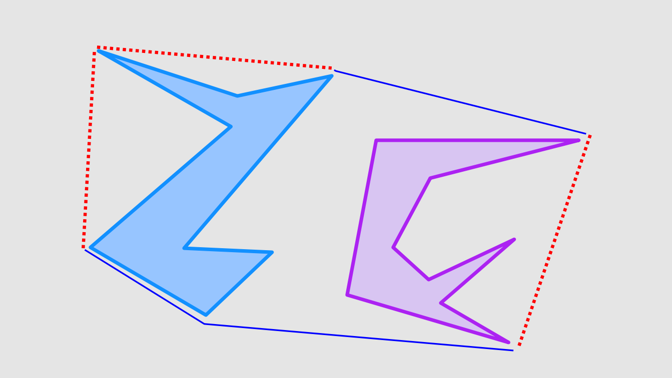
Construct convex hull of each

Possible Hull of Pair of Polygons



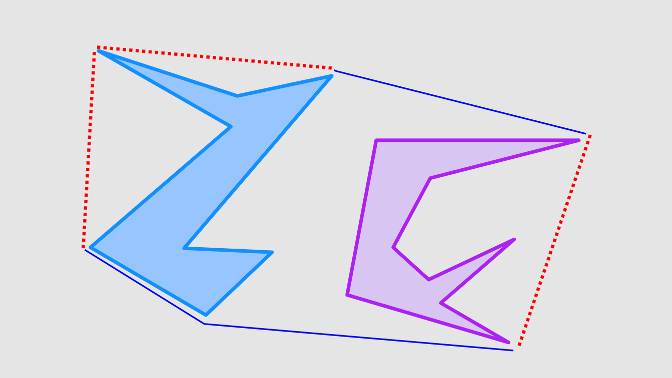
Construct convex hull of pair

Possible Hull of Pair of Polygons



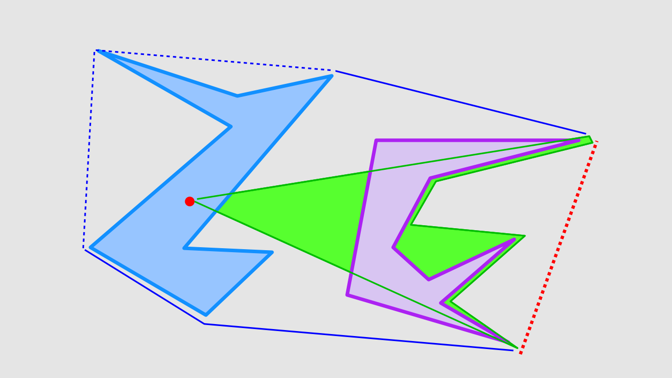
Identify pockets

Possible Hull of Pair of Polygons

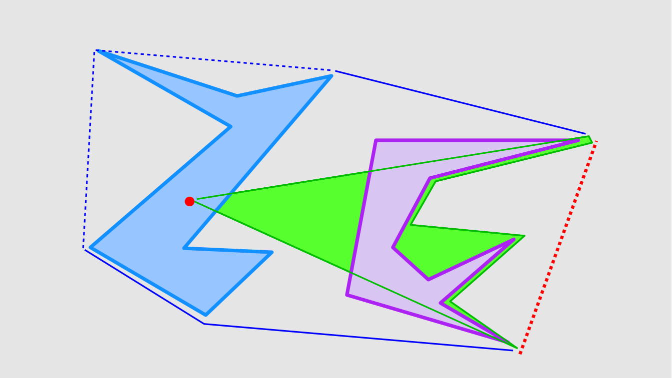


Two additional steps: Hull Contraction and Hull Expansion

Hull Contraction

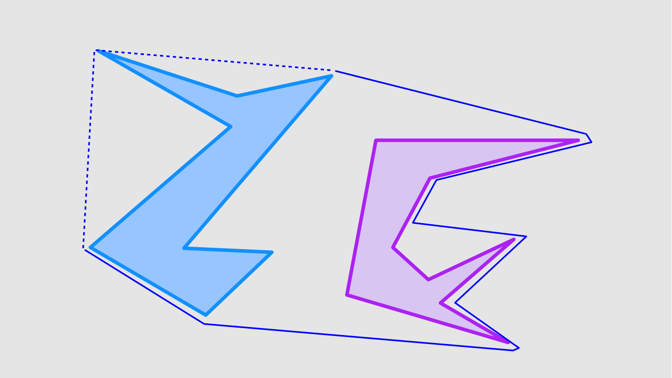


Hull Contraction



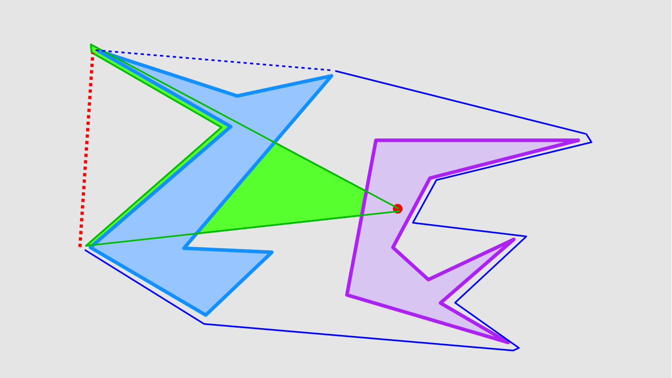
Replace pocket lid with possible hull of chain and point

Hull Contraction



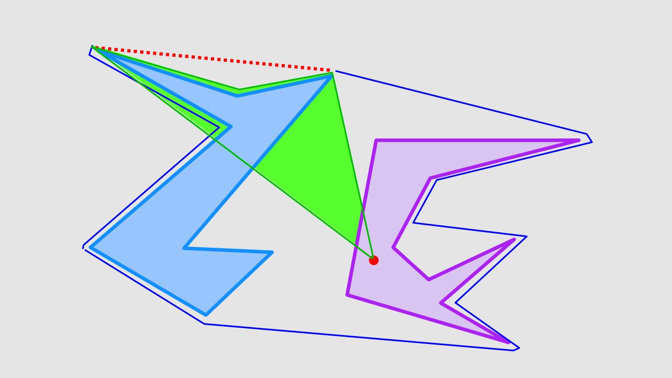
Replace pocket lid with possible hull of chain and point

Hull Contraction



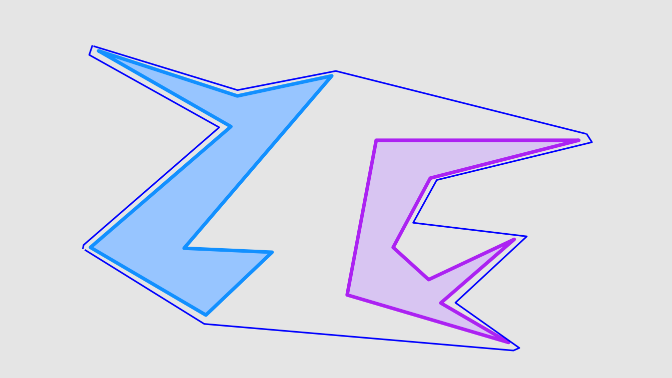
Replace pocket lid with possible hull of chain and point

Hull Contraction



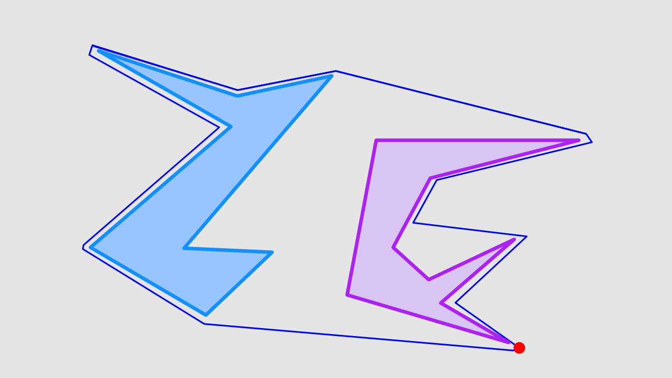
Replace pocket lid with possible hull of chain and point

Hull Contraction



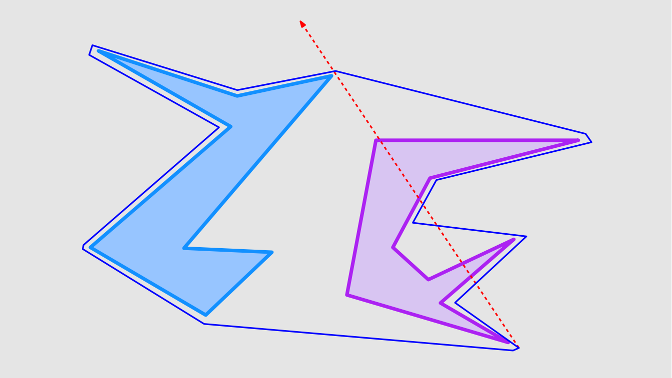
Replace pocket lid with possible hull of chain and point

Hull Expansion



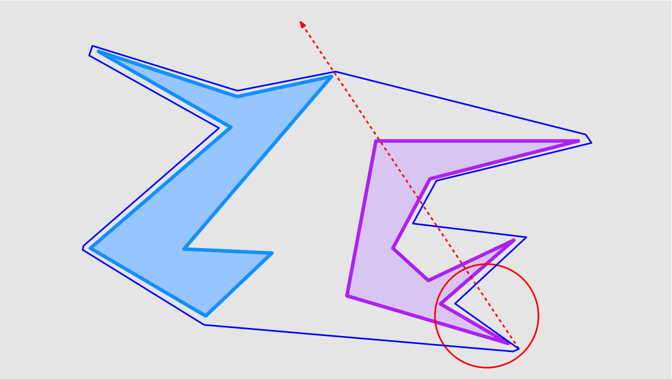
Hull Expansion: walk boundary ccw from any convex hull vertex

Hull Expansion



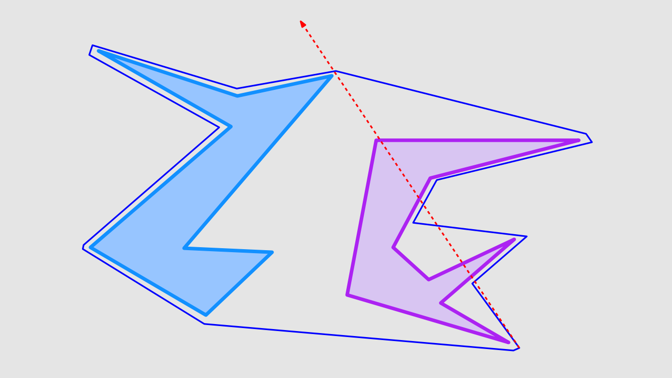
Construct line tangent to opposite polygon

Hull Expansion



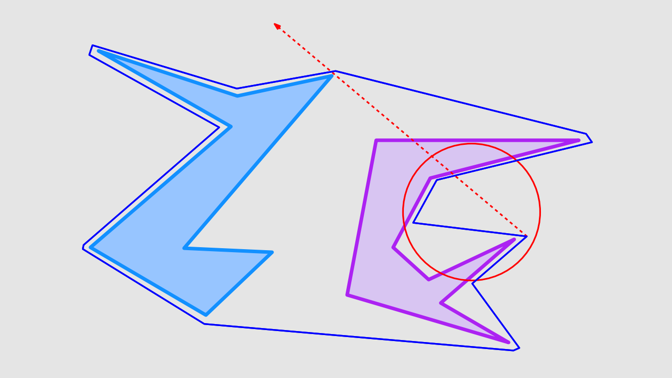
Expand by filling in pockets encountered

Hull Expansion



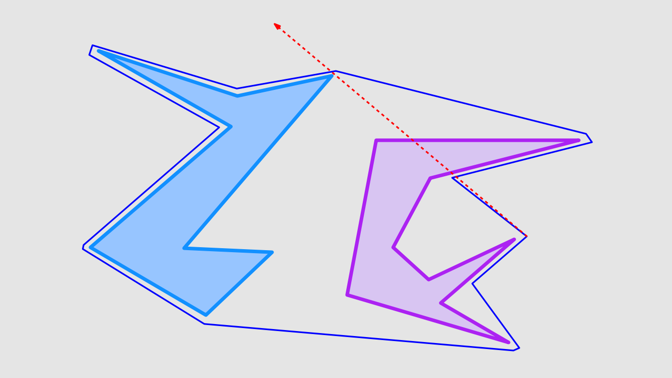
Expand by filling in pockets encountered

Hull Expansion



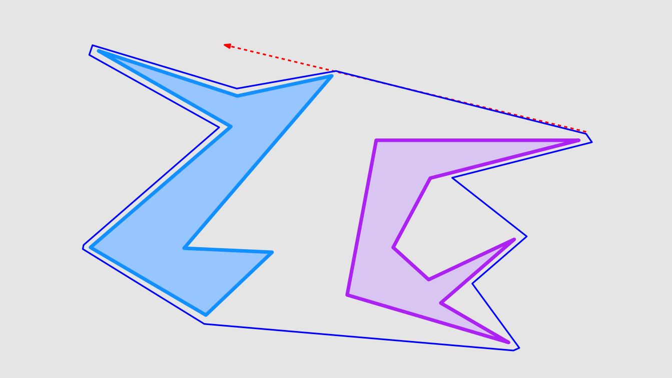
Repeat until returned to starting vertex

Hull Expansion



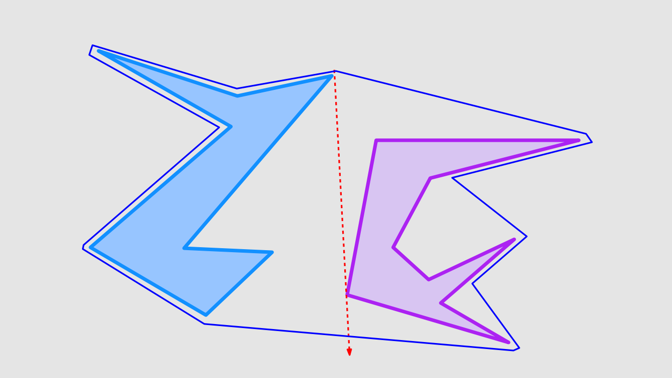
Repeat until returned to starting vertex

Hull Expansion



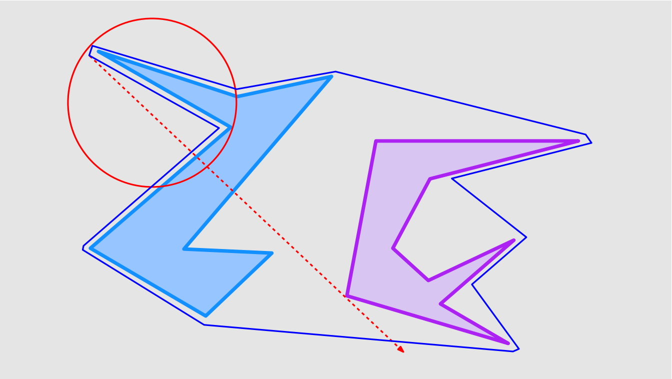
Repeat until returned to starting vertex

Hull Expansion



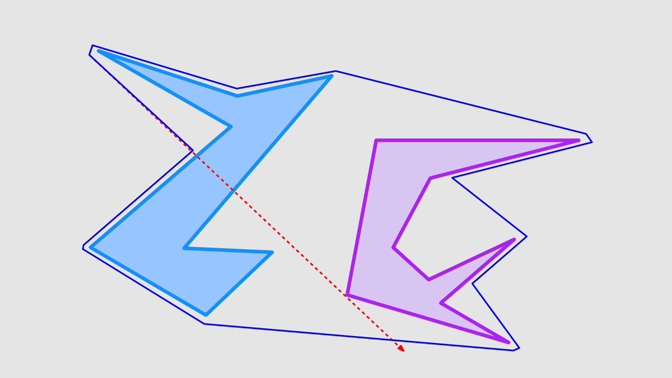
Repeat until returned to starting vertex

Hull Expansion



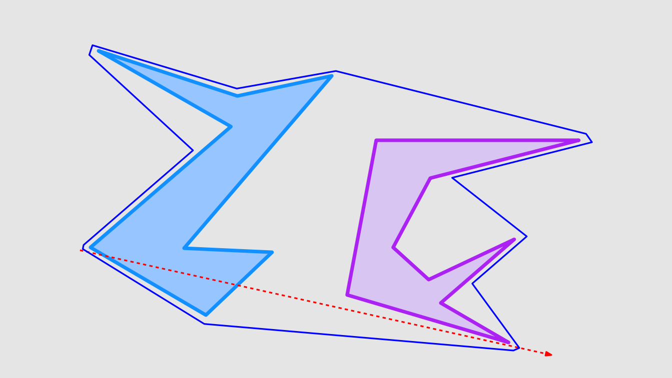
Repeat until returned to starting vertex

Hull Expansion



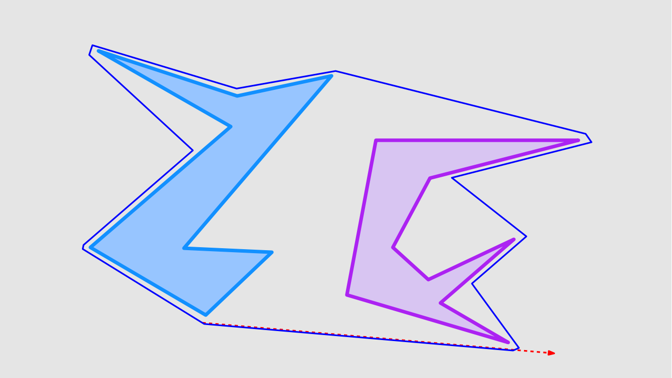
Repeat until returned to starting vertex

Hull Expansion



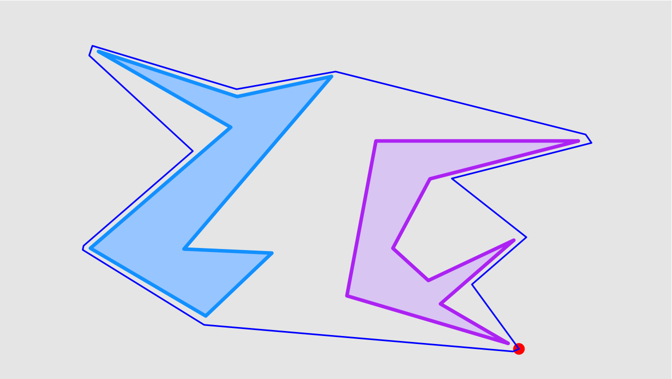
Repeat until returned to starting vertex

Hull Expansion



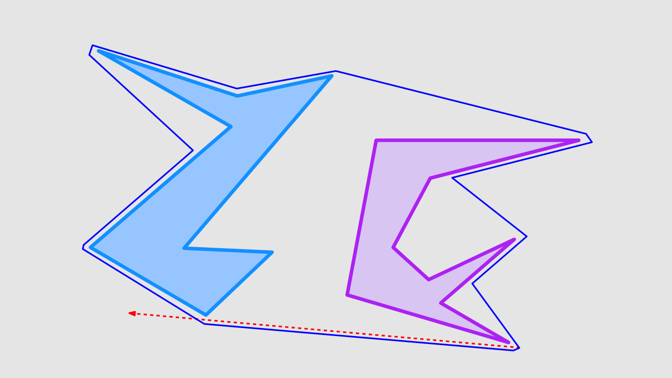
Repeat until returned to starting vertex

Hull Expansion



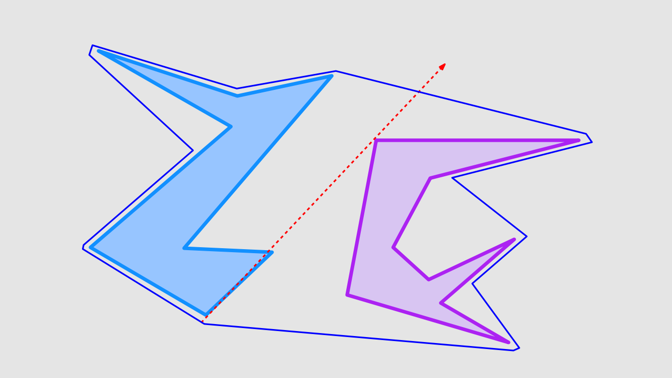
Repeat until returned to starting vertex

Hull Expansion



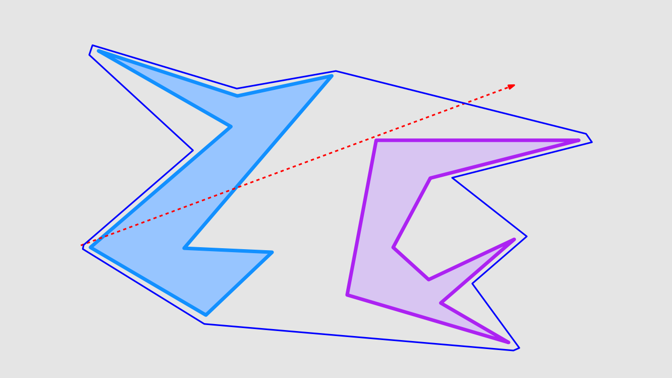
Perform symmetric procedure cw

Hull Expansion



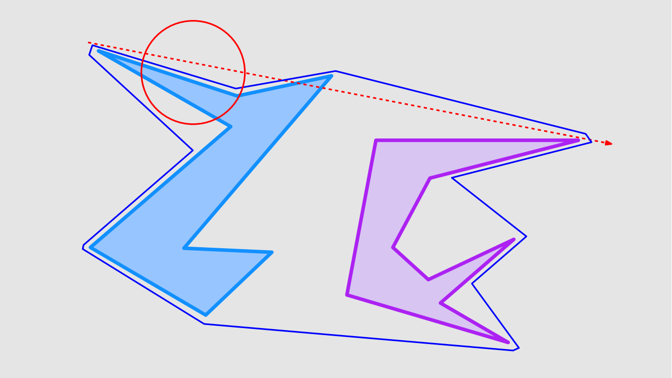
Perform symmetric procedure cw

Hull Expansion



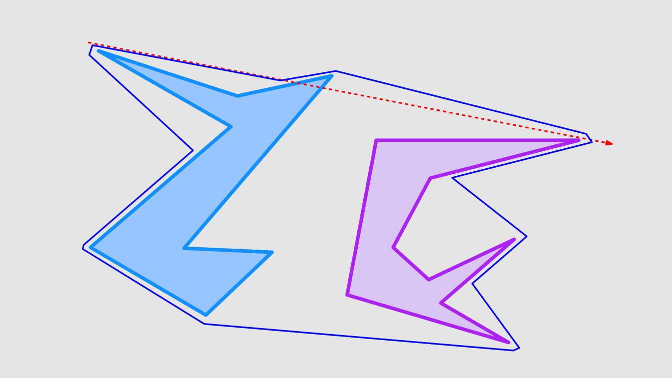
Perform symmetric procedure cw

Hull Expansion



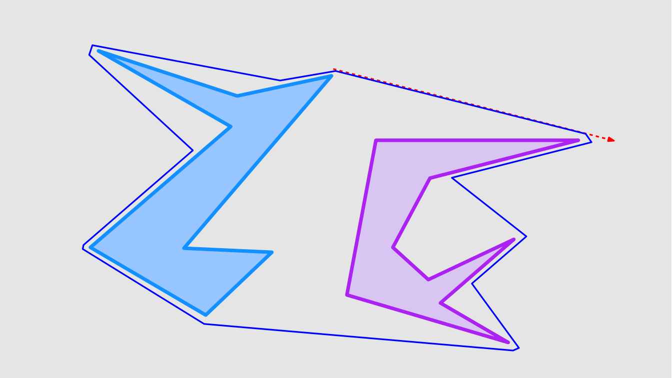
Perform symmetric procedure cw

Hull Expansion



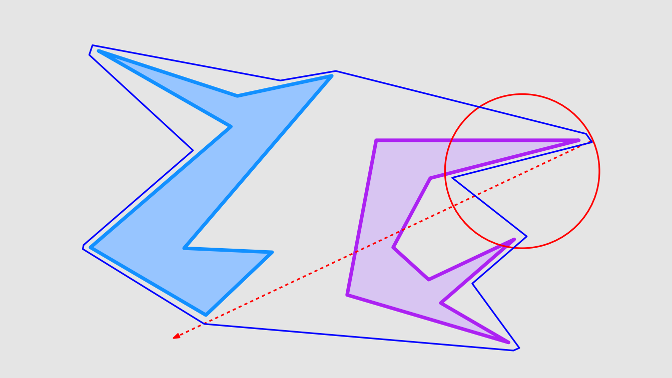
Perform symmetric procedure cw

Hull Expansion



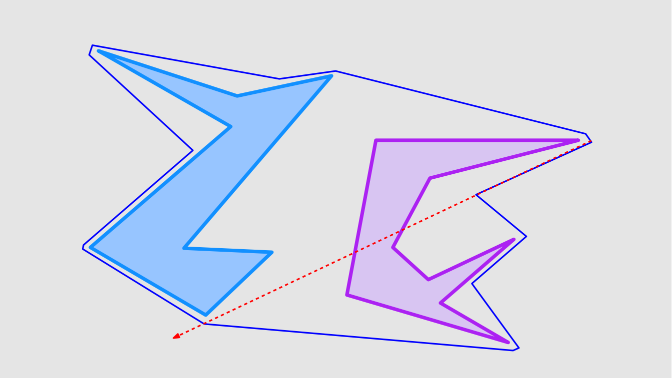
Perform symmetric procedure cw

Hull Expansion



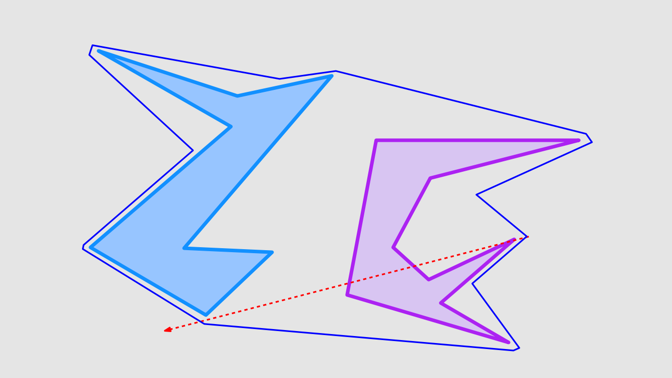
Perform symmetric procedure cw

Hull Expansion



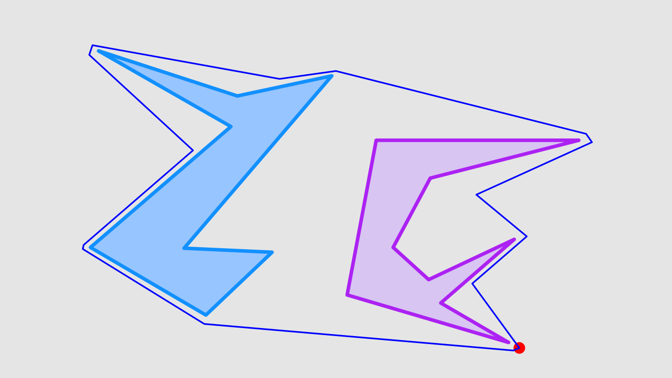
Perform symmetric procedure cw

Hull Expansion



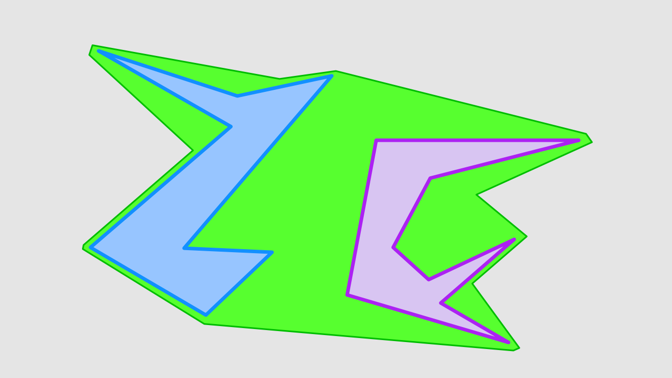
Perform symmetric procedure cw

Hull Expansion



Perform symmetric procedure cw

Hull Expansion



Possible hull

Pair of Polygons: Running Time

- Convex hull of each polygon: $O(n)$ time [Melkman, '87]

Pair of Polygons: Running Time

- Convex hull of each polygon: $O(n)$ time [Melkman, '87]
- Convex hull of hulls: $O(n)$ time, using rotating calipers [Toussaint, '83]

Pair of Polygons: Running Time

- Convex hull of each polygon: $O(n)$ time [Melkman, '87]
- Convex hull of hulls: $O(n)$ time, using rotating calipers [Toussaint, '83]
- Hull contraction: $O(n)$ time, using point / polygon algorithm

Pair of Polygons: Running Time

Expansion step:

- Adds only reflex vertices

Pair of Polygons: Running Time

Expansion step:

- Adds only reflex vertices
- Each can be charged to a distinct convex vertex

Pair of Polygons: Running Time

Expansion step:

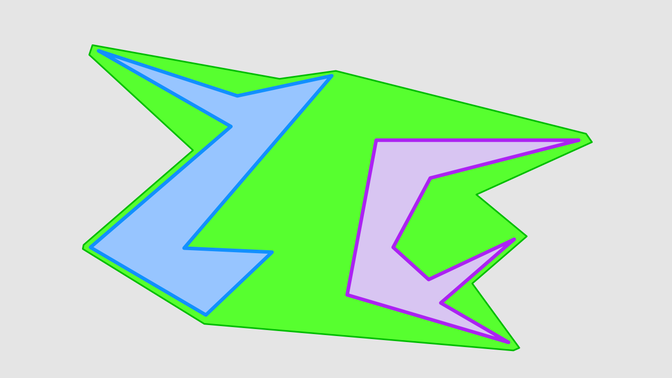
- Adds only reflex vertices
- Each can be charged to a distinct convex vertex
- Contraction step ensures tangents have monotonically increasing angles

Pair of Polygons: Running Time

Expansion step:

- Adds only reflex vertices
- Each can be charged to a distinct convex vertex
- Contraction step ensures tangents have monotonically increasing angles
- $\implies$ amortized $O(n)$ cost for constructing tangents

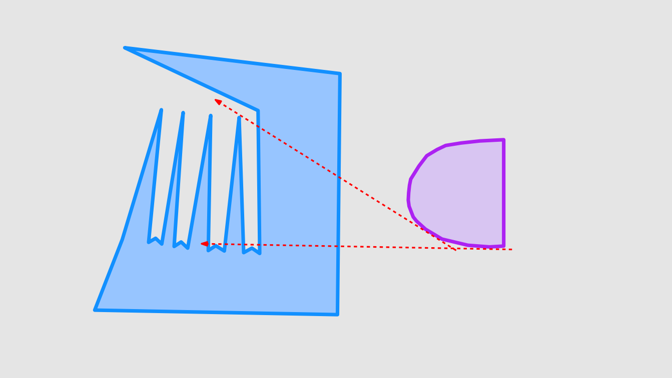
Pair of Polygons: Running Time



Theorem 12

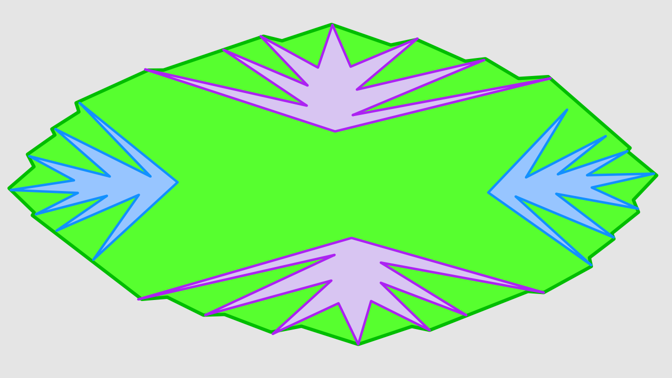
Possible hull of pair of polygons can be constructed in linear time.

Pair of Polygons: Running Time



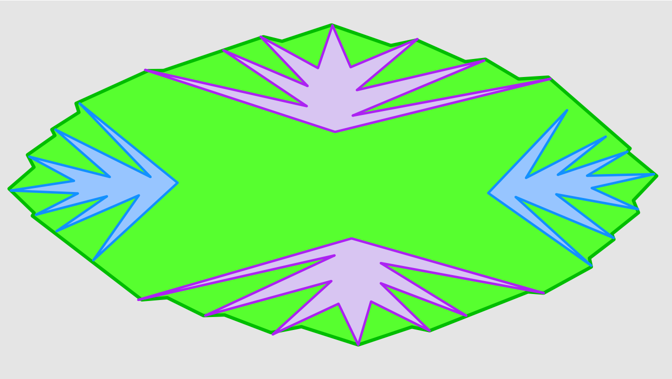
If contraction step omitted, monotonicity doesn't hold

Multiple Polygons



- Lemma 4 implies set of k polygons can be processed recursively

Multiple Polygons



- Lemma 4 implies set of k polygons can be processed recursively

Theorem 13

Possible hull of k polygons with n total vertices can be constructed in $O(n \log k)$ time.

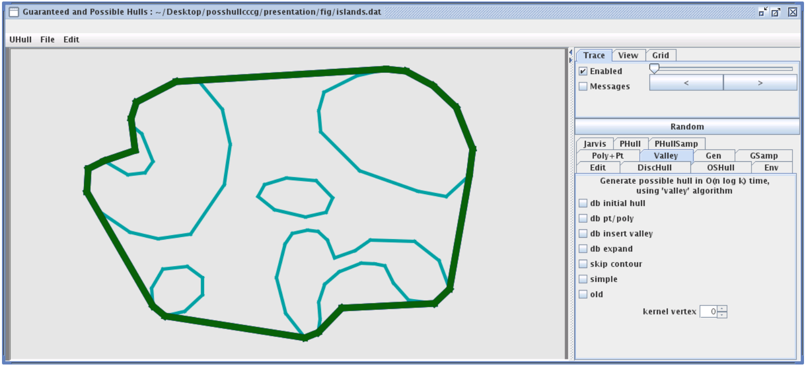
Multiple Polygons

- Worst-case optimal (consider k small triangles in convex position)

Multiple Polygons

- Worst-case optimal (consider k small triangles in convex position)
- Not output sensitive

Thank You!



Applet available at: www.cs.ubc.ca/~jpsember