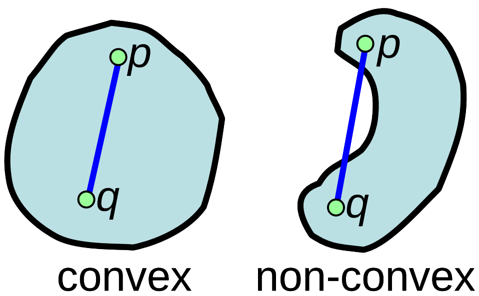
Convex Hull

Pierre Alliez

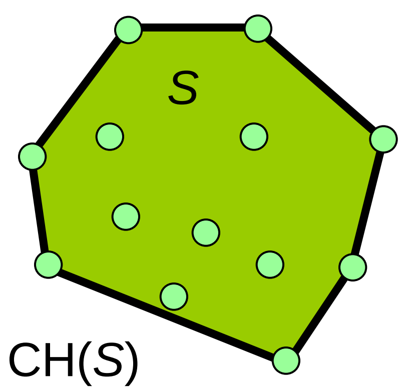
Convexity and Convex Hull

- A set S is *convex* if any pair of points $p,q \in S$ satisfy $pq \subseteq S$.



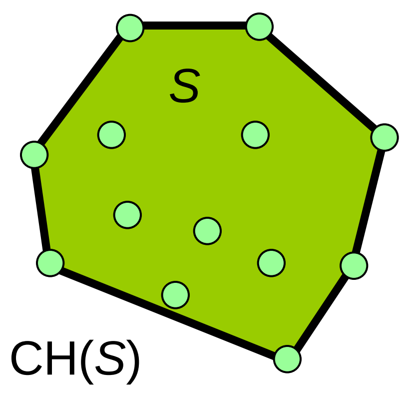
- The *convex hull* of a set S is:
  - The minimal convex set that contains S, i.e. any convex set C such that $S \subseteq C$ satisfies $\mathrm{CH}(S) \subseteq C$.
  - The intersection of all convex sets that contain S.
  - The set of all convex combinations of $p_i \in S$, i.e. all points of the form:

$$\sum_{i=1}^{n} \alpha_i p_i \ , \qquad \alpha_i \geq 0, \ \sum_{i=1}^{n} \alpha_i = 1$$



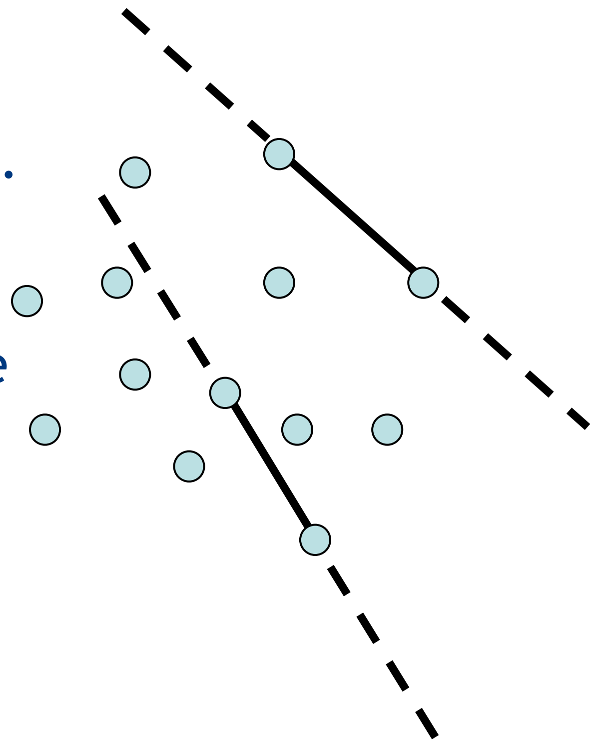
Convex Hulls – Some Facts

- The convex hull of a set is unique (up to colinearities).
- The boundary of the convex hull of a point set is a polygon on a subset of the points.



Convex Hull – Naive Algorithm

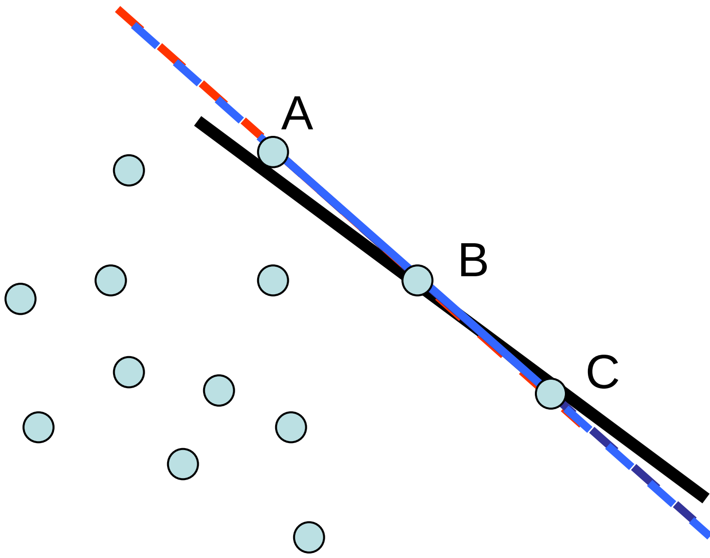
- Description:
  - For each pair of points construct its connecting segment and *supporting line*.
  - Find all the segments whose supporting lines divide the plane into two halves, such that one half plane contains *all* the other points.
  - Construct the convex hull out of these segments.



- Time complexity:
  - All pairs: $$O\left(\binom{n}{2}\right) = O\left(\frac{n(n-1)}{2}\right) = O(n^2)$$
  - Check all points for each pair: O(n)
  - Total: O(n^3)

Possible Pitfalls

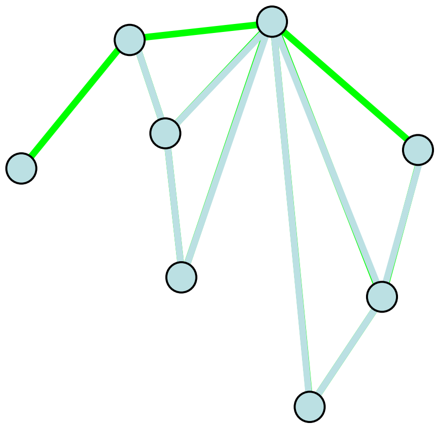
- Degenerate cases – e.g. 3 collinear points. Might harm the correctness of the algorithm. Segments AB, BC and AC will *all* be included in the convex hull.



- Numerical problems – We might conclude that *none* of the three segments belongs to the convex hull.

Convex Hull – Graham's Scan

- Algorithm:
  - Sort the points according to their *x* coordinates.
  - Construct the upper boundary by scanning the points in the sorted order and performing only "right turns".
  - Construct the lower boundary (with "left turns").
  - Concatenate the two boundaries.



- Time Complexity: O(*n*log*n*)
- May be implemented using a stack
- **Question:** How do we check for "right turn" ?

The Algorithm

- Sort the points in increasing order of x-coord:

  $$p_1, \ldots, p_n.$$

- Push(S,p_1); Push(S,p_2);
- For i = 3 to n do
  - While Size(S) $\geq$ 2 and Orient(p_i,top(S),second(S)) $\leq$ 0 do Pop(S);
  - Push(S,p_i);
- Print(S);

Graham's Scan – Time Complexity

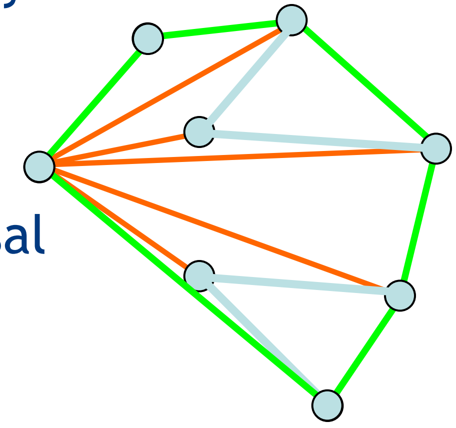
- Sorting – O(n log n)
- If D_i is number of points popped on processing p_i,

$$\text{time} = \sum_{i=1}^{n} (D_i + 1) = n + \sum_{i=1}^{n} D_i$$

- Each point is pushed on the stack only once.
- Once a point is popped – it cannot be popped again.
- Hence $$\sum_{i=1}^{n} D_i \leq n$$

Graham’s Scan– a Variant

- Algorithm:
  - Find one point, p_0, which must be on the convex hull.
  - Sort the other points by the *angle* of the rays to them from p_0.
  - **Question:** Is it necessary to compute the actual angles ?
  - Construct the convex hull using one traversal of the points.



- Time Complexity: O(*n* log *n*)

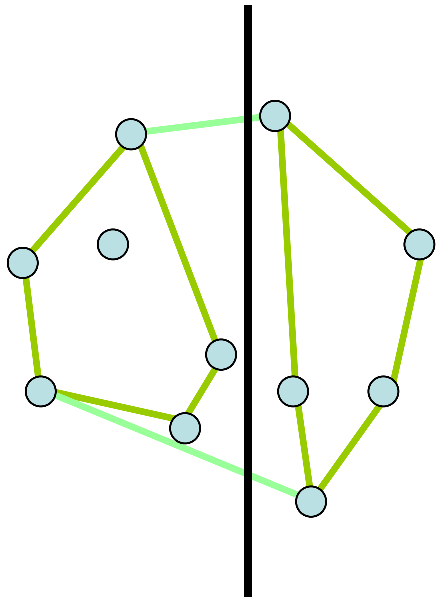
- **Question:** What are the pros and cons of this algorithm relative to the previous ?

Convex Hull - Divide and Conquer

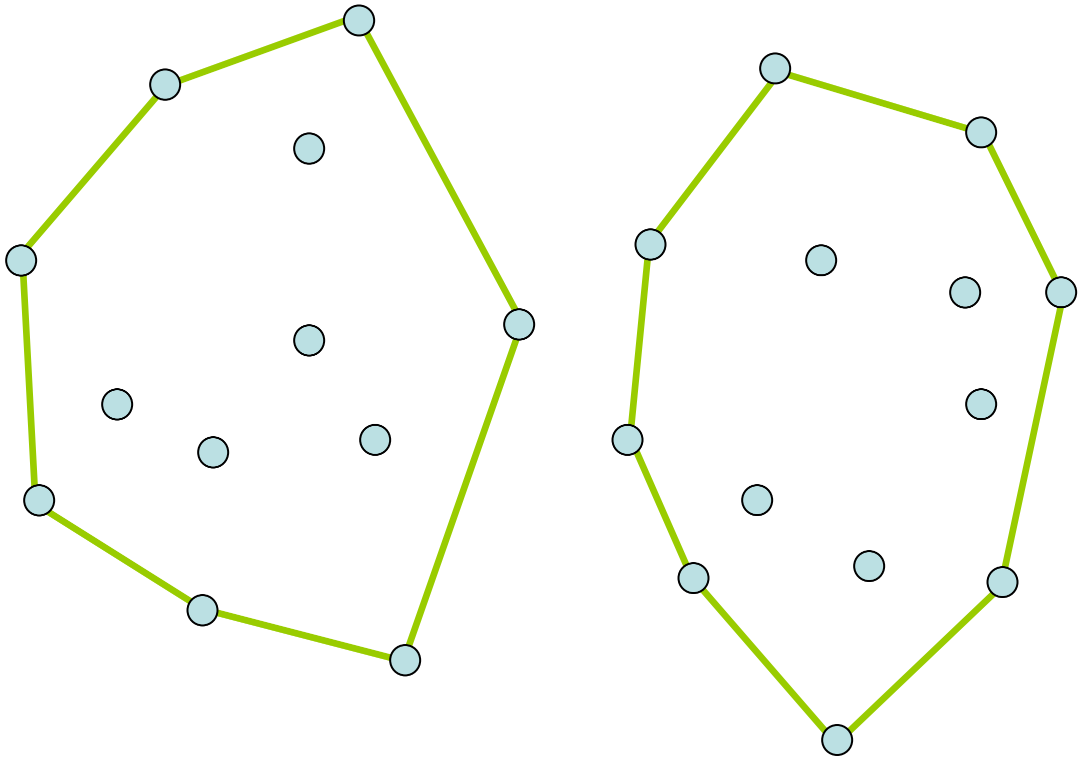
- Algorithm:
  - Find a point with a median *x* coordinate (time: O(*n*))
  - Compute the convex hull of each half (recursive execution)
  - Combine the two convex hulls by finding common *tangents*.

    This can be done in O(*n*).

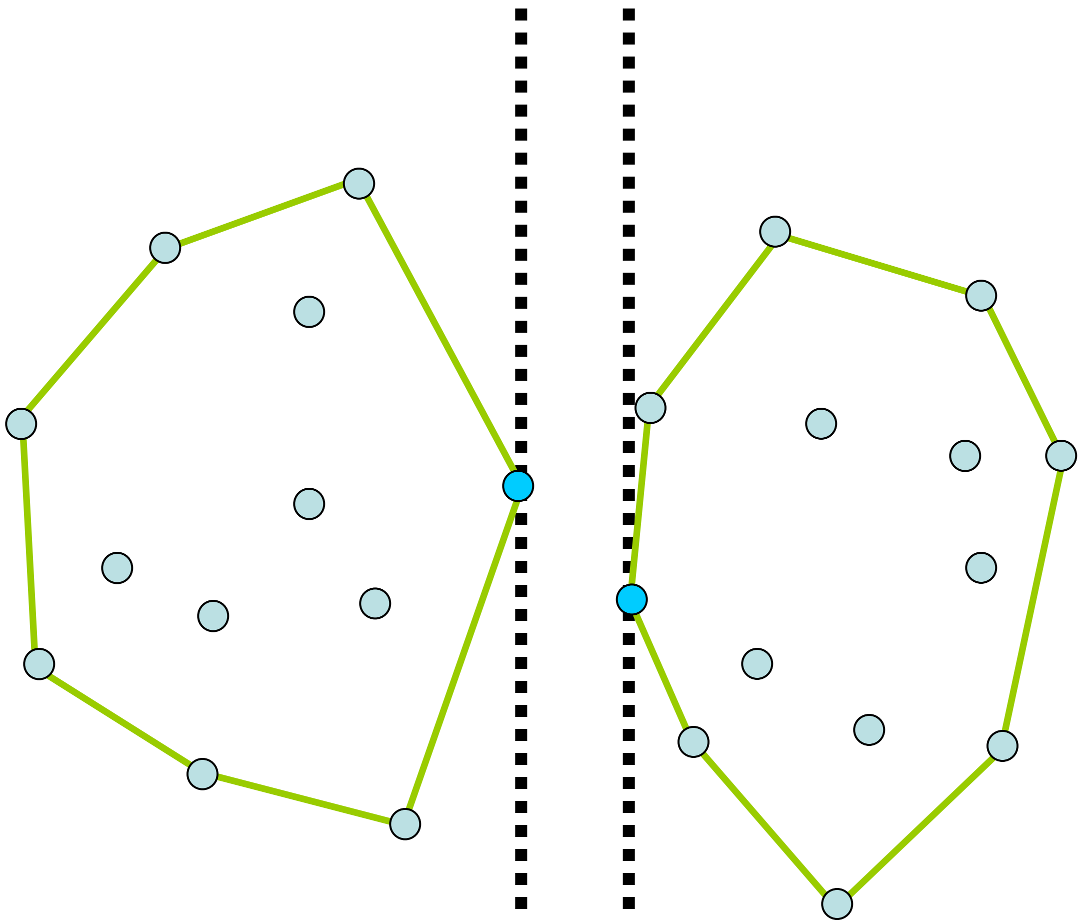
  - Complexity: O(*n*log*n*)



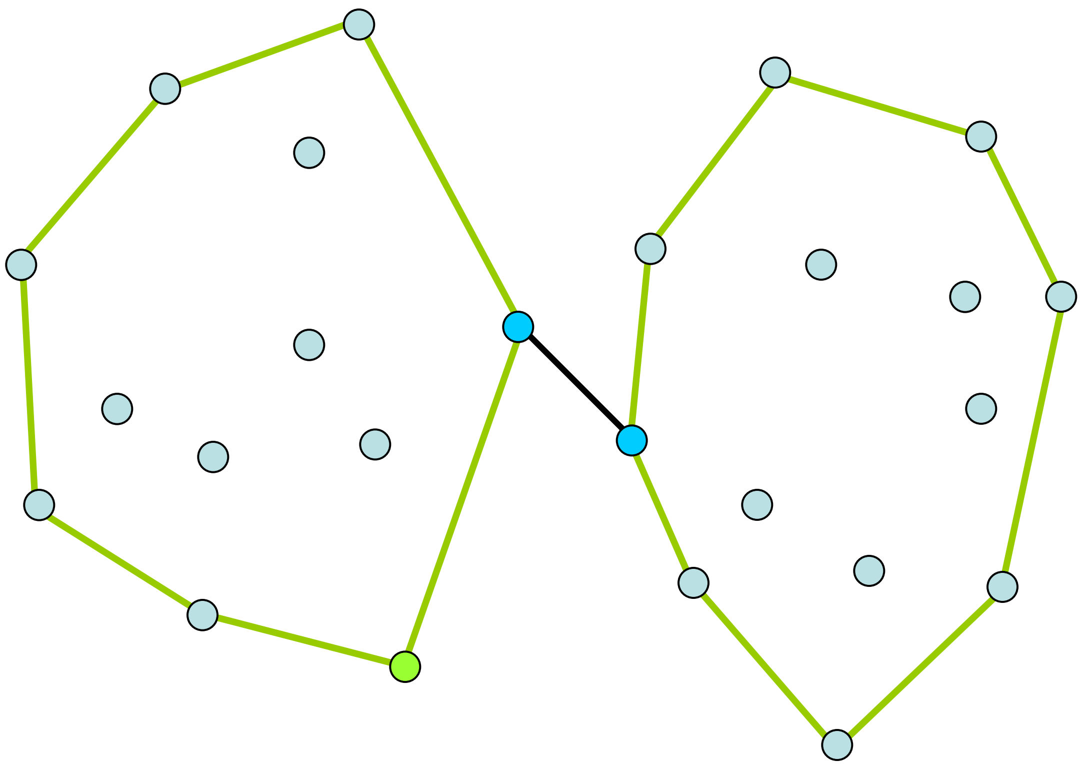
Finding Common Tangents



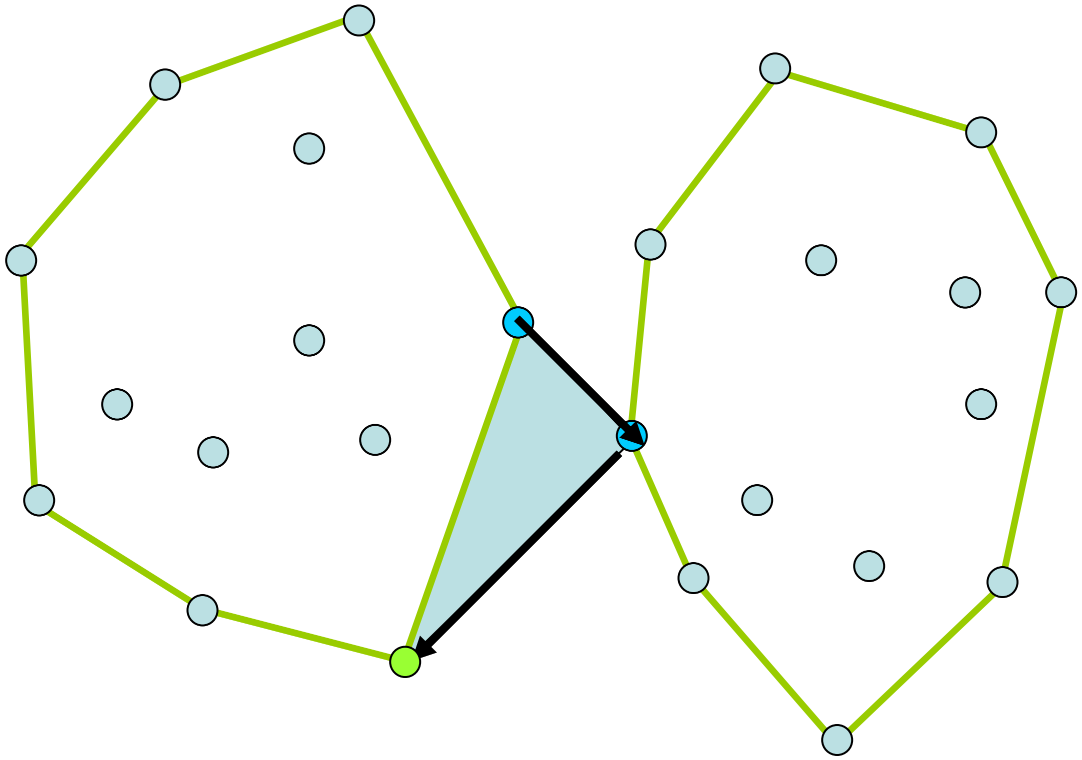
Finding Common Tangents



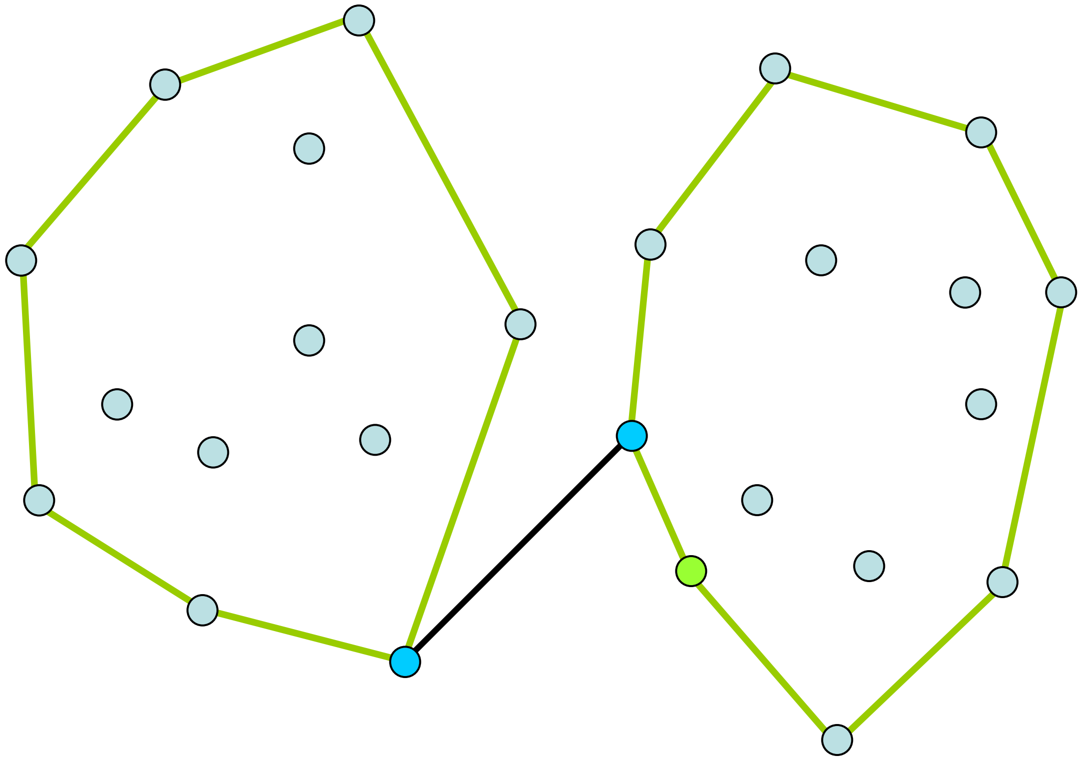
Finding Common Tangents



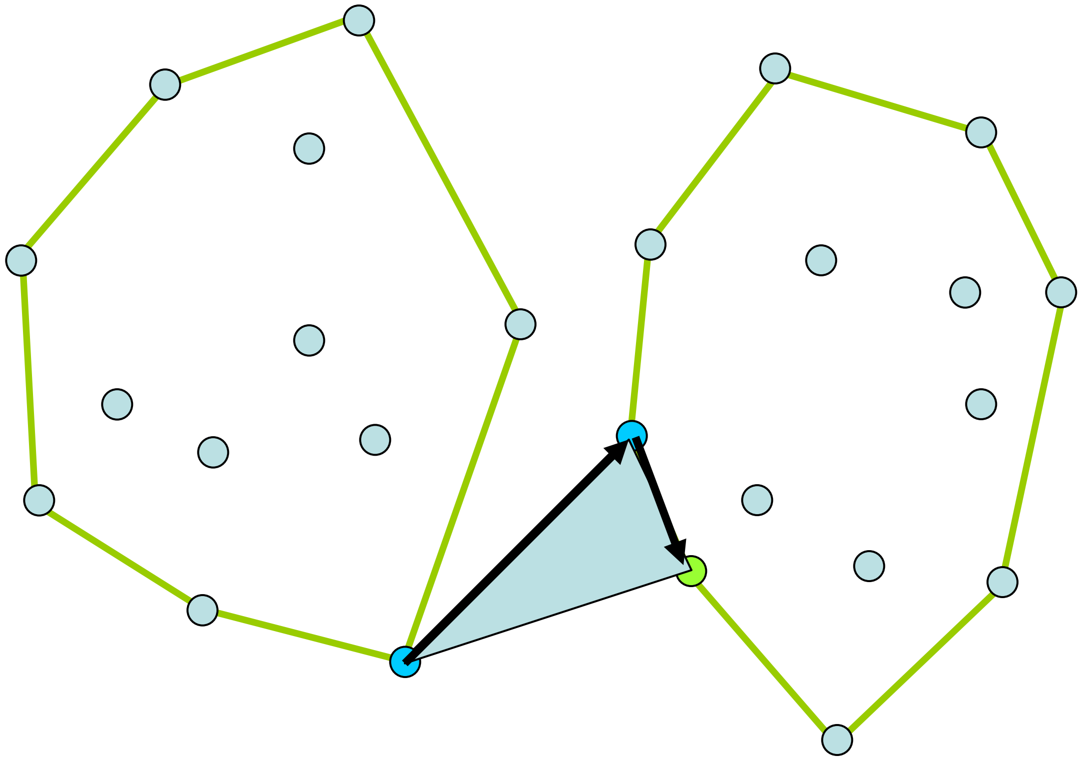
Finding Common Tangents



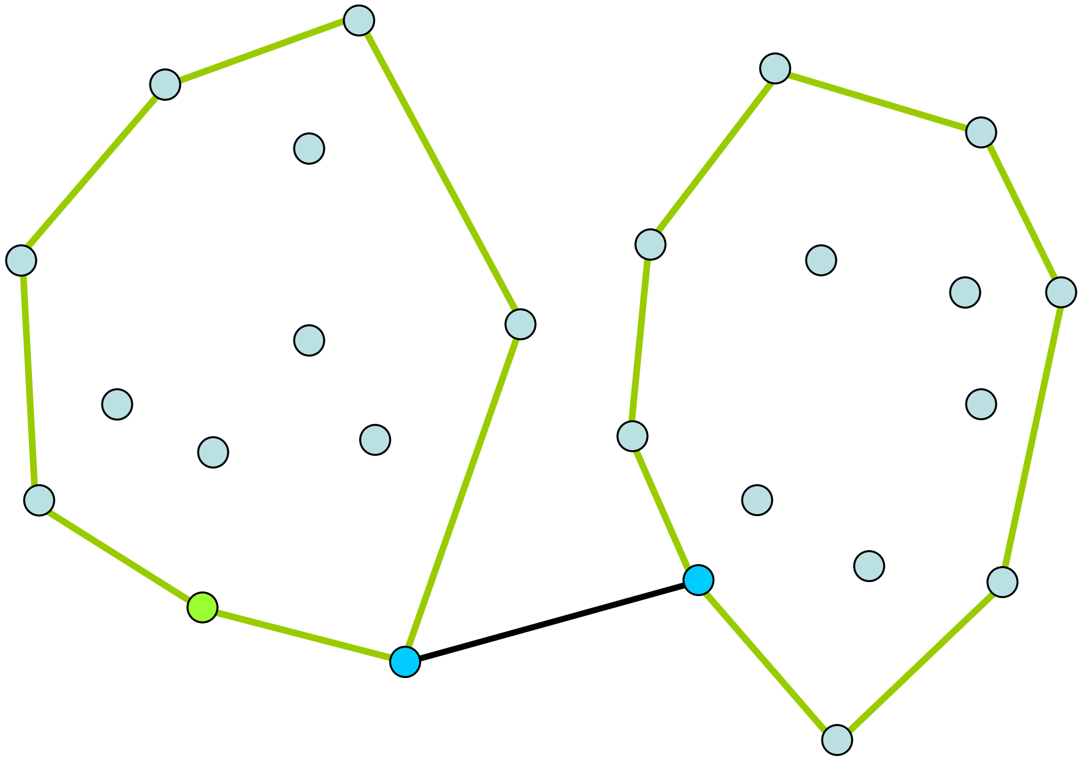
Finding Common Tangents



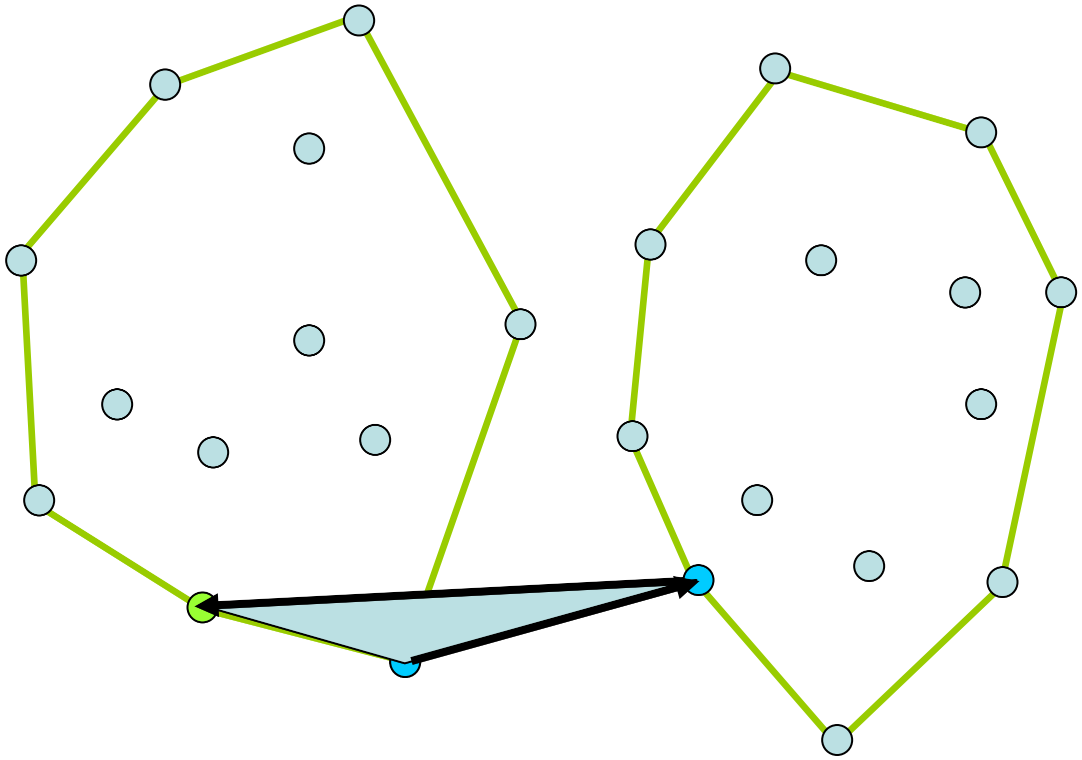
Finding Common Tangents



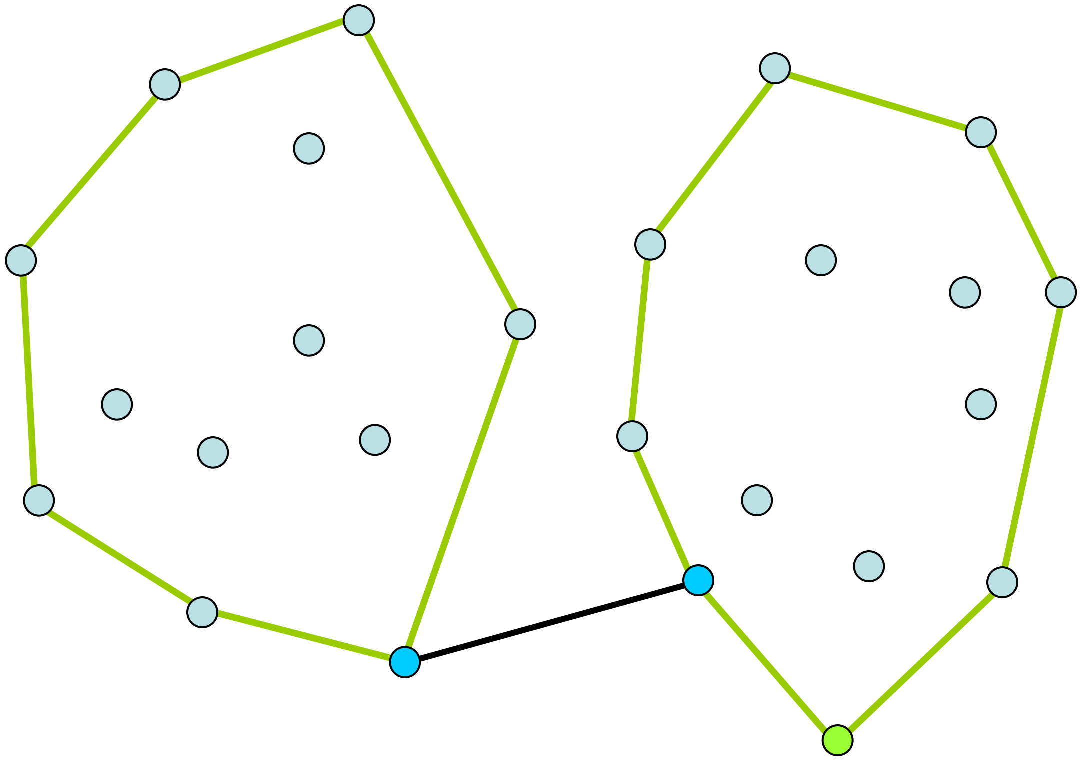
Finding Common Tangents



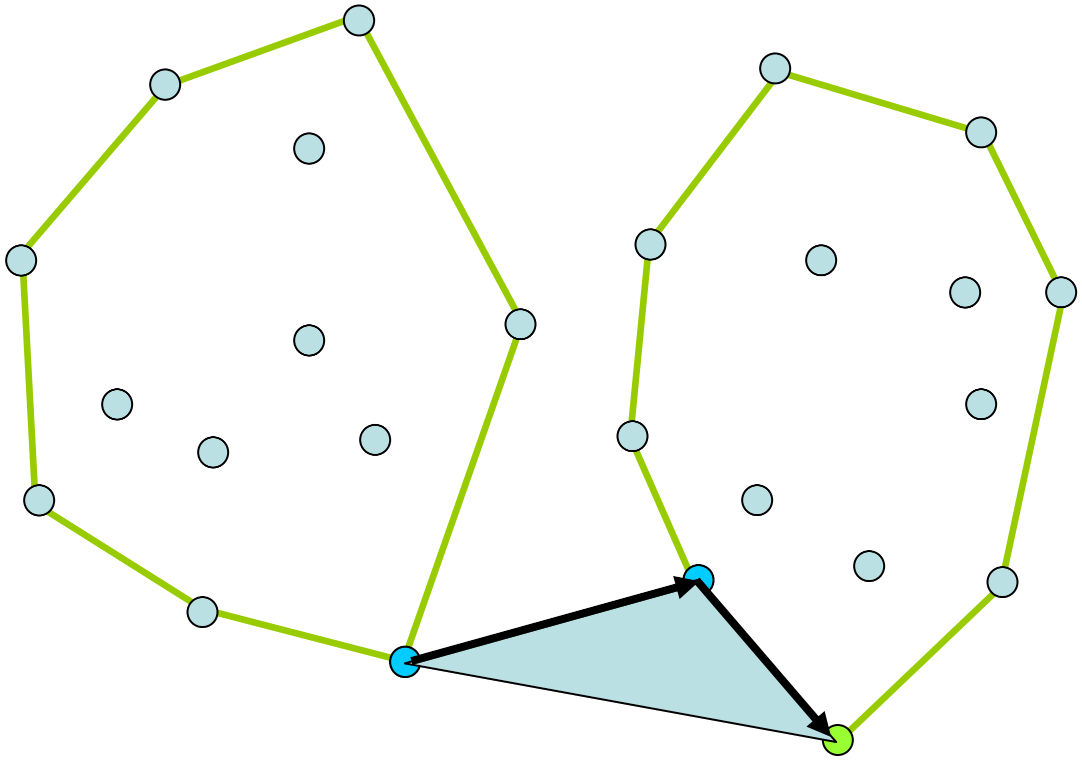
Finding Common Tangents



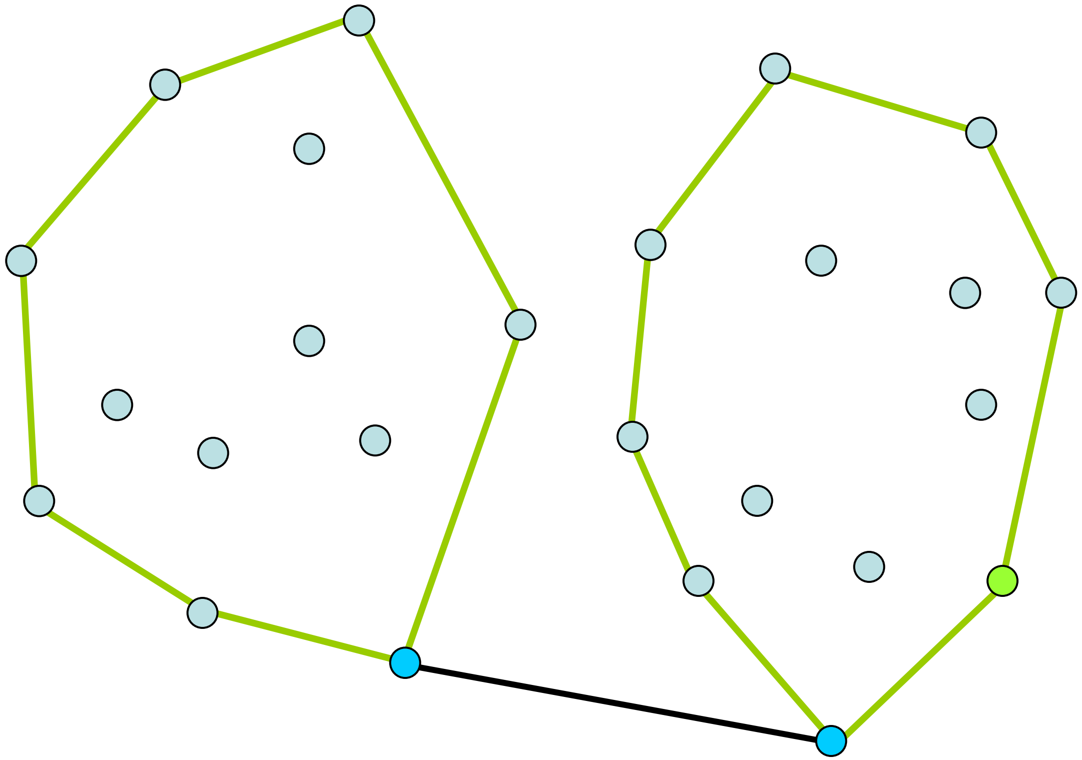
Finding Common Tangents



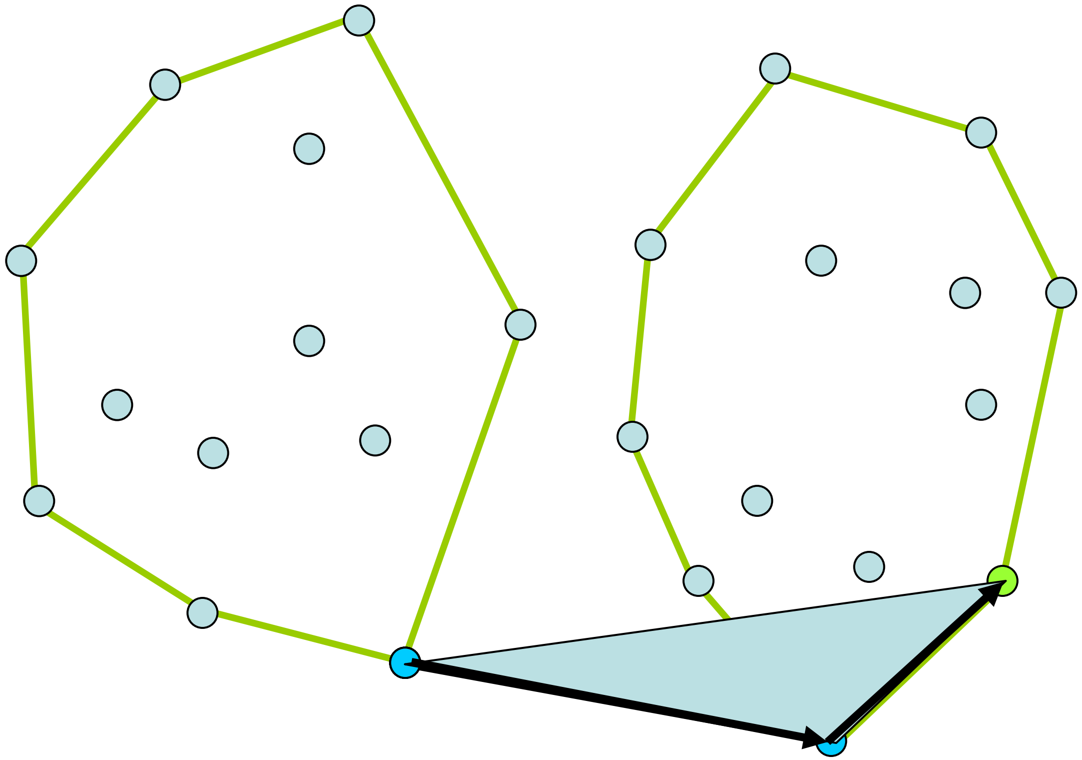
Finding Common Tangents



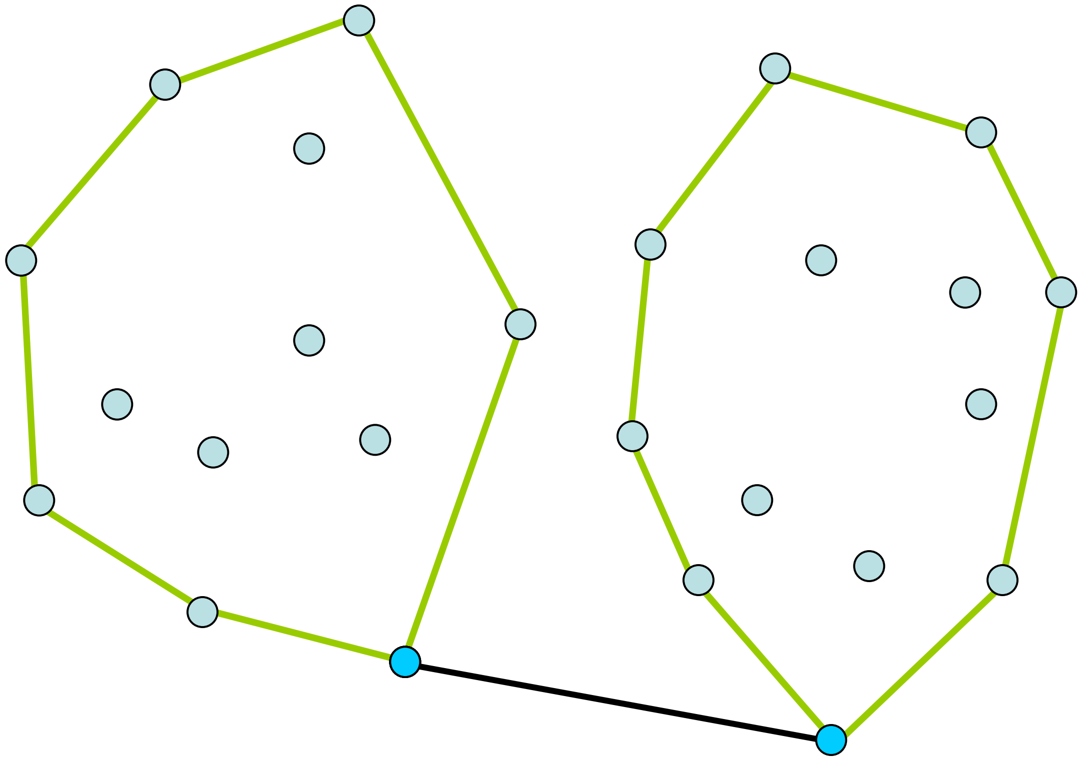
Finding Common Tangents



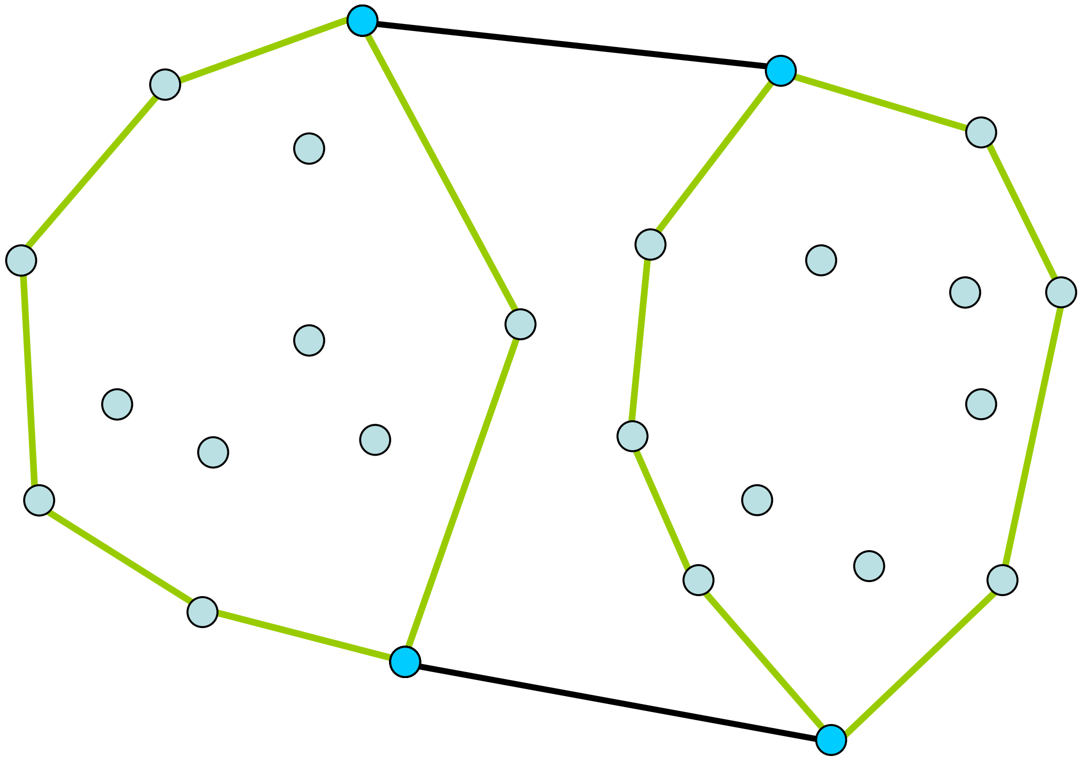
Finding Common Tangents



Finding Common Tangents



Finding Common Tangents



Finding Common Tangents

To find lower tangent:

- Find *a* - the rightmost point of H_A
- Find *b* – the leftmost point of H_B

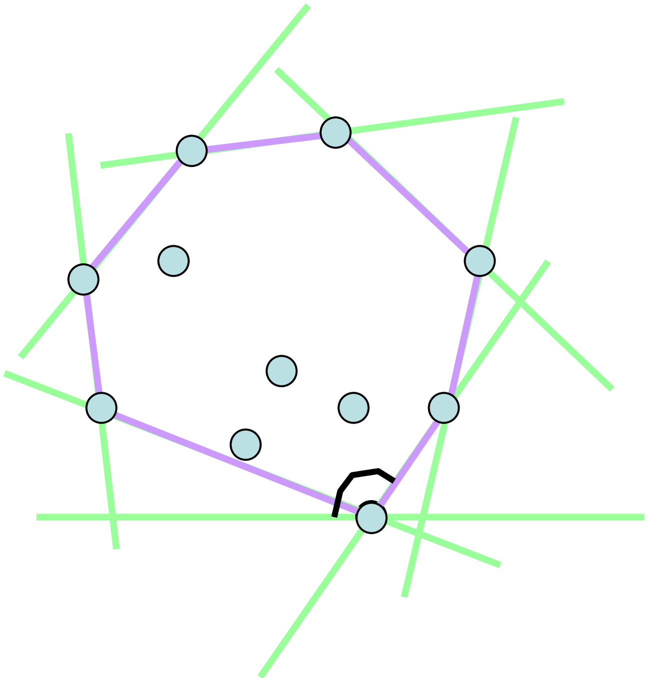
O(n)

- While *ab* is not a lower tangent for H_A and H_B, do:
  - If *ab* is not a lower tangent to H_A do $a = a$-1
  - If *ab* is not a lower tangent to H_B do $b = b$-1

Output-Sensitive Convex Hull Gift Wrapping

- Algorithm:
  - Find a point p_1 on the convex hull (e.g. the lowest point).
  - Rotate counterclockwise a line through p_1 until it touches one of the other points (start from a horizontal orientation).

    Question: How is this done ?



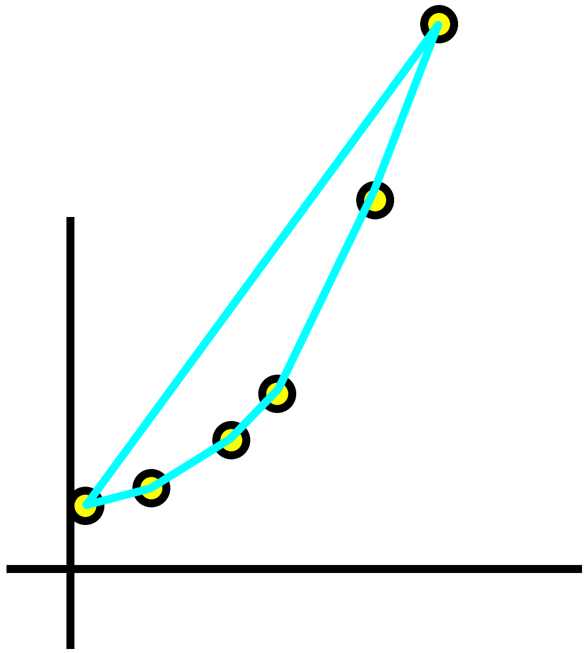
  - Repeat the last step for the new point.
  - Stop when p_1 is reached again.
- Time Complexity: O(nh), where n is the input size and h is the output (hull) size.

General Position

- When designing a geometric algorithm, we first make some simplifying assumptions, e.g.
  - No 3 collinear points.
  - No two points with the same *x* coordinate.
  - etc.
- Later, we consider the general case:
  - How should the algorithm react to degenerate cases ?
  - Will the correctness be preserved ?
  - Will the runtime remain the same ?

Lower Bound for Convex Hull

- A reduction from sorting to convex hull is:
  - Given n real values x_i, generate n 2D points on the graph of a convex function, e.g. (x_i, x_i^2).
  - Compute the (ordered) convex hull of the points.
  - The order of the convex hull points is the numerical order of the x_i.



- So CH=$\Omega(n \lg n)$