Computational Geometry

Workshop in Competitive Programming – 234900

“Let no man ignorant of geometry enter here”

Plato’s Academy in Athens

Agenda

- Basic Concepts
- Point in Polygon
- Convex Hull

Basic Concepts

Lines, Polygons, Orientation and Convexity

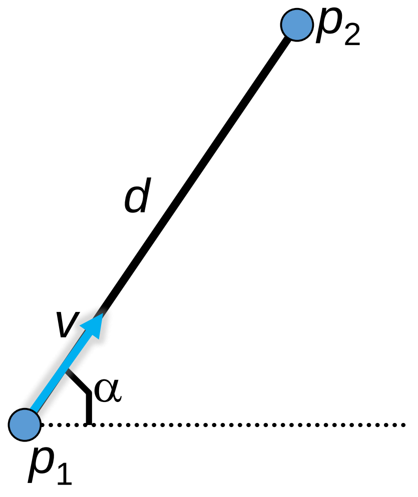
General Settings

- Usually 2D
- Simple geometric objects (e.g. points, lines and polygons)
- Forgot high school geometry? That’s alright

Representing Geometric Elements

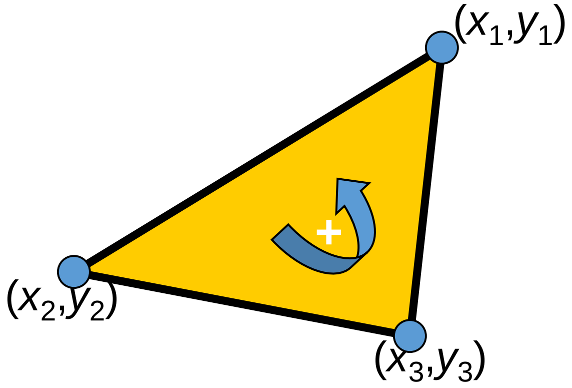
- Representation of a line segment by four real numbers:
  - Two endpoints (p_1 and p_2)
  - One endpoint (p_1), vector direction (v) and parameter interval length (d)

    (Question: where did the extra parameter come from?)
  - One endpoint (p_1), a slope (α), and length (d)
  - Other options…



Orientation

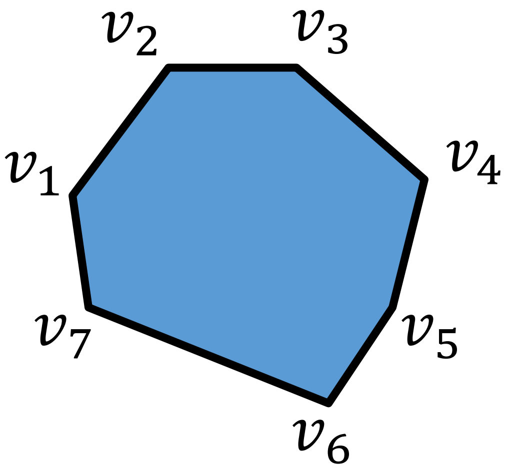
$$Area = \frac{1}{2}\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$



- The sign of the area indicates the orientation of the points.
- Positive area ≡ counterclockwise orientation ≡ left turn.
- Negative area ≡ clockwise orientation ≡ right turn.

- **Question:** How can this be used to determine whether a given point is "above" or "below" a given line?

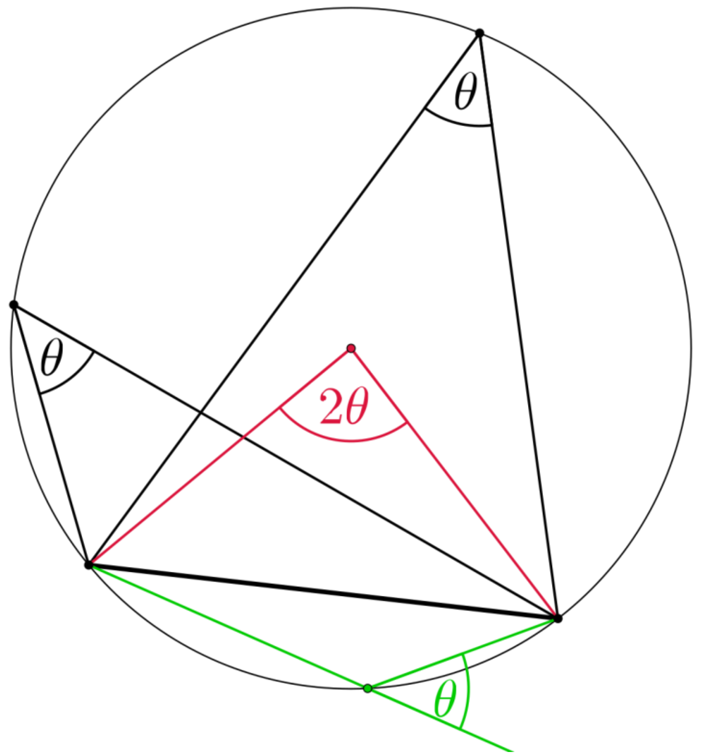
Polygons



- Represented as a list of (ordered) points
- We will deal with simple polygons
  - No intersections
  - No holes
- Useful properties:
  - Sum of angles $= 180 \cdot (n - 2)$
  - Regular polygon: A polygon for which the sides are all equal and the angles are all equal

Circles

- Represented as a center and a radius.
- Useful properties:
  - Three points uniquely determine a circle
  - Inscribed angle size is half the size of the inner angle.

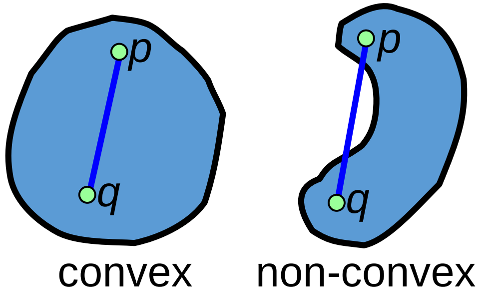


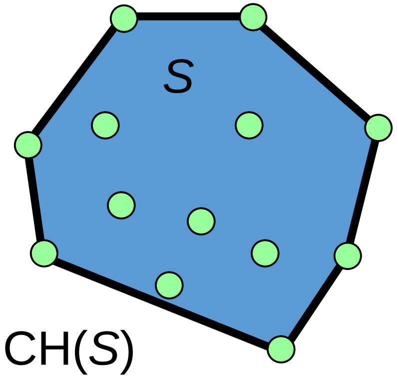
Formulas

- Distance between two points $= \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$
- Distance between $ax + by + c = 0$ (a line) to $(x_0, y_0) = \frac{|ax_0+by_0+c|}{\sqrt{a^2+b^2}}$
- Circle circumference $= 2\pi r$, and area $= \pi r^2$
- Angle between two vectors $\cos^{-1}(\frac{v_1 \cdot v_2}{|v_1||v_2|})$
- Area of a **simple** polygon $= \frac{1}{2}\sum_{i=0}^{n-1}(x_i y_{i+1} - x_{i+1} y_i)$ (Shoelace formula)
  - Where $(x_0, y_0) = (x_n, y_n)$
- Many more…

Convexity and Convex Hull

- Definition: A set S is *convex* if for any pair of points $p, q \in S$, the entire line segment $pq \subseteq S$.
- The *convex hull* (קְמוֹר) of a set *S* is the minimal convex set that contains *S*.
- Another (equivalent) definition: The intersection of **all** convex sets that contain *S*.



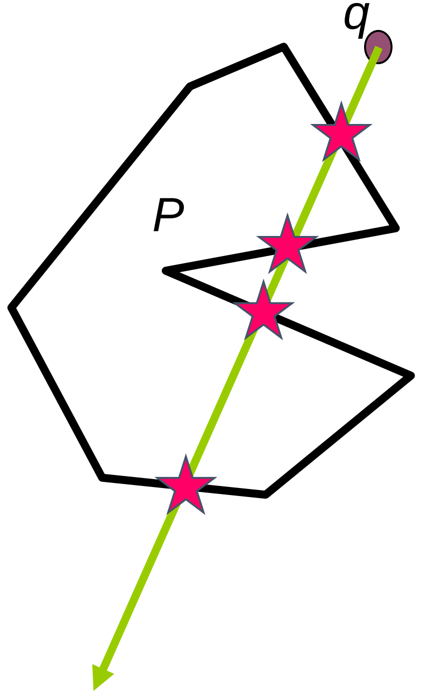


Point in Polygon

Non-Convex and Convex

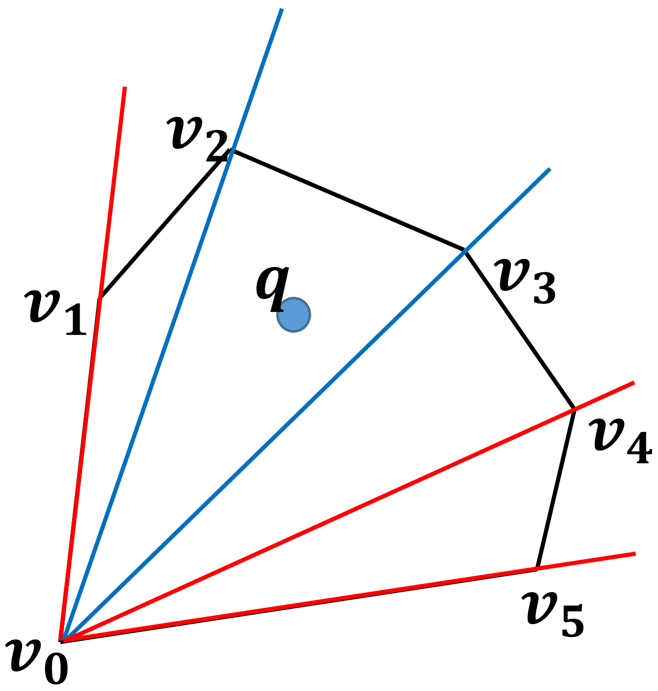
Point in Polygon

- Problem: Given a polygon P with n sides, and a point q, decide whether q lies inside P.
- Solution 1: count how many times a ray from q to infinity intersects the polygon. q lies inside P iff this number is odd.
- Time complexity: $\Theta(n)$
- Need to pay attention if the ray passes a vertex



Point in Convex Polygon

- Same problem, convex polygon
- Partition into wedges with vertex v_0
- **Binary search** to find the wedge q lies in
  - Binary search according to what?
- Check if q lies in the triangle
  - How?
- Time complexity: $O(\log n)$



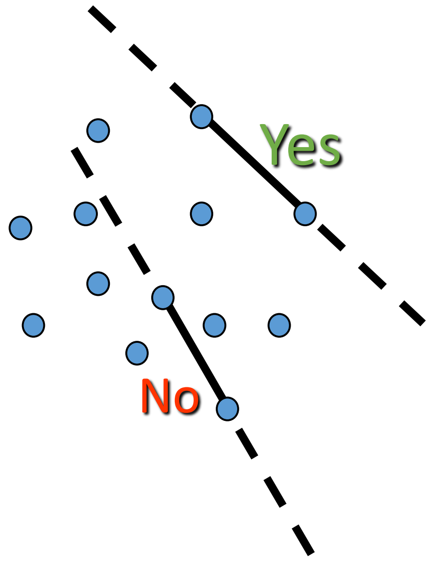
Convex Hull Algorithms

Naïve, Graham Scan, Gift Wrapping

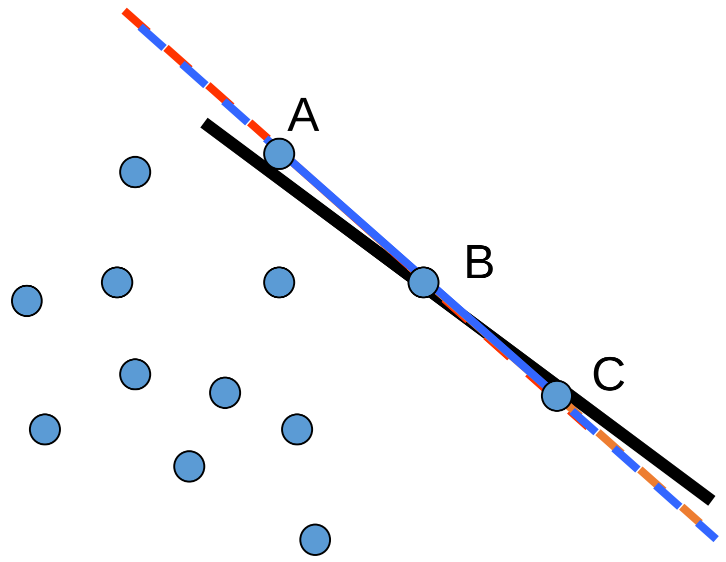
Convex Hull: Naïve Algorithm

- Description:
  - For each pair of points construct its *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 (for *n* points):
  - Number of point pairs: $\Theta(n^2)$
  - Check all points for each pair: $\Theta(n)$
  - Total: $\Theta(n^3)$



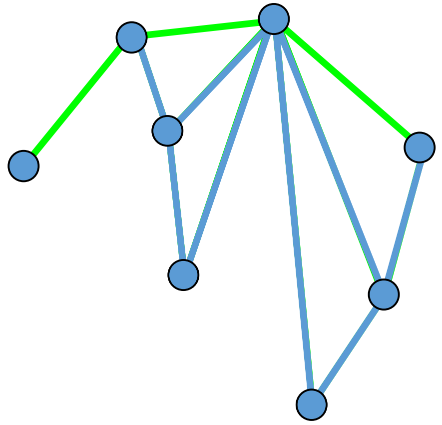
Possible Pitfalls



- Degenerate cases, e.g., 3 collinear points, may harm the correctness of the algorithm. All, or none, of the segments AB, BC and AC will be included in the convex hull.
- **Question**: How can we solve the problem?
- Numerical problems: We might conclude that *none* of the three segments (or a wrong pair of them) belongs to the convex hull.
- **Question:** How is collinearity detected?

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" (trim off "left turns").
  - Construct the lower boundary in the same manner.
  - Concatenate the two boundaries.
- Time Complexity: O(n log n) (only!)
- May be implemented using a stack



- **Question:** How do we check for a "right turn"?

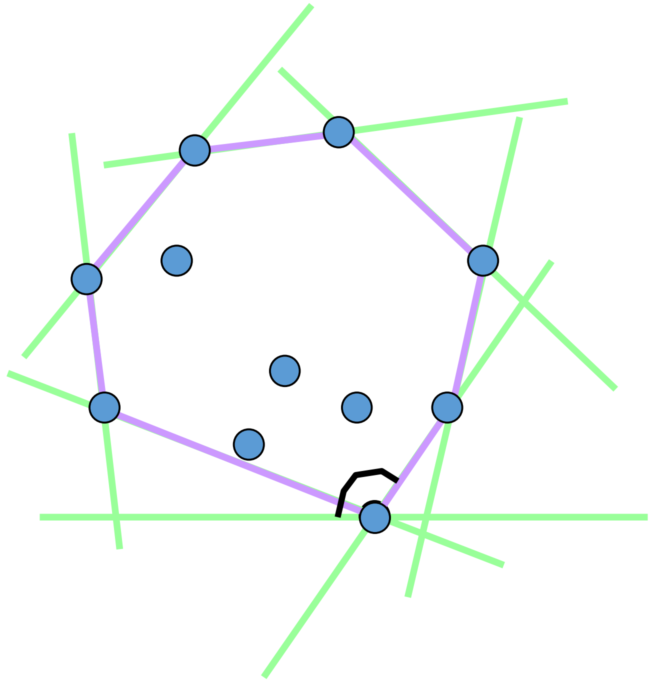
The Algorithm

- Input: Point set $\{p_i\}$.
- Sort the points in increasing order of x coordinates:

  $p_1, \ldots, p_n$.
- 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);
- Output S.

Convex Hull: Gift Wrapping

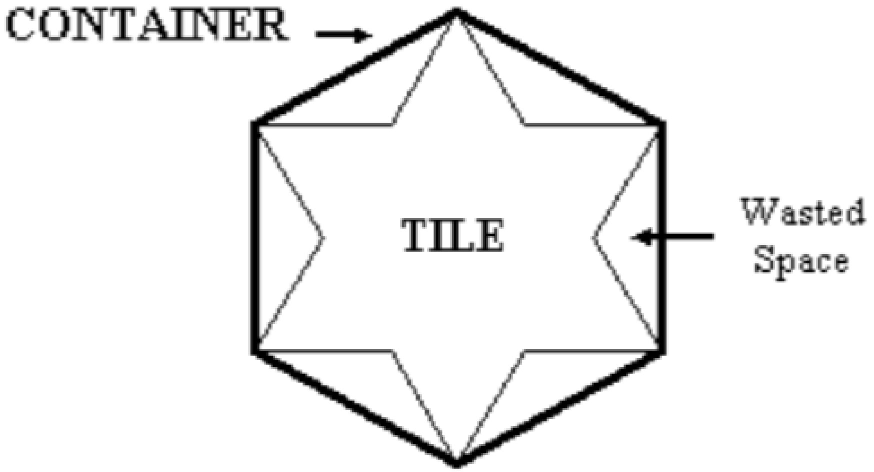
- Algorithm:
  - Find a point p_1 on the convex hull (e.g., the lowest point).
  - Rotate counterclockwise a line through p_1 until it touches one of the other points (start from a horizontal orientation).
  - **Question:** How is this done?
  - Repeat the last step for the new point.
  - Stop when p_1 is reached again.
- Time Complexity: O(nh), where n is the input size and h is the output (hull) size.
- Since $3 \leq h \leq n$, time is $\Omega(n)$ and $O(n^2)$.



Example problem

Useless Tile Packers

- A factory of tiles creates polygonal tiles and pack them in a convex container.
- Wasted space – empty area inside the container.
- Problem: What is the minimum possible percentage of wasted space?
- $n =$ Num of vertices
- $n \leq 100$



Tips

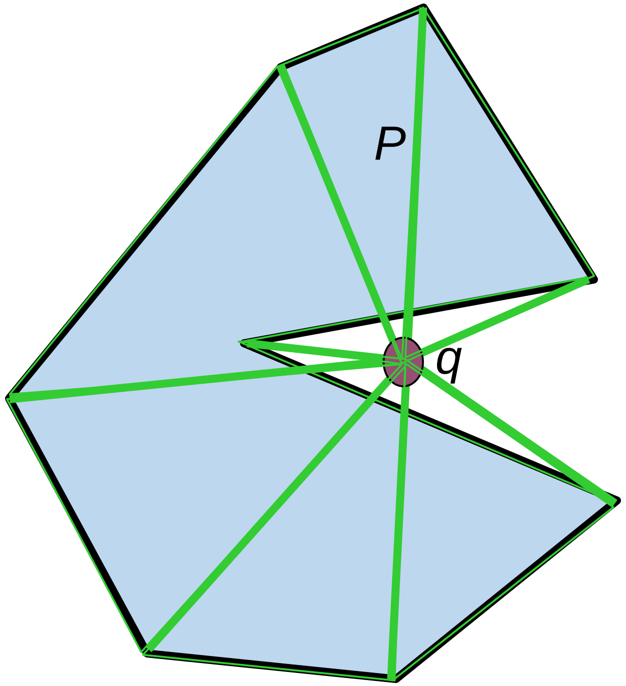
Tips

- Beware of precision problems.
  - For example to check equality: $abs(x - y) \leq \mathrm{EPS}$.
- Notice the input size. Sometimes the size is small Sand a brute force solution will work.
- Draw! A good drawing is worth a 1000 equations.

EXTRAS

Point in Polygon

- Solution 2: Sum up the angles $\alpha_i = \sphericalangle p_i q p_{i+1}$ for $i = 0,1,\ldots,n-1$.
  Sum = 2π if q lies inside P, otherwise Sum = 0.
- $\alpha_i = \sin^{-1}\left(\frac{\text{signed\_area}(p_i,q,p_{i+1})}{\|p_i - q\|\|p_{i+1} - q\|}\right)$
- Note that some angles are negative
- Time complexity: $\Theta(n)$



Lower Bound for Convex Hull

- A reduction from Sorting to convex hull:
  - Given n real values x_i, generate n points on the graph of a convex function, e.g., a parabola, (x_i, x_i^2).
  - Compute the polygon C, the convex hull of the points.
  - The order of the points on C is the same order as that of the x_i.
- Hence, Complexity(CH)=$\Omega(n \log n)$
- Due to the existence of $O(n \log n)$-time algorithms, Complexity(CH)= $\Theta(n \log n)$