

Algorithm Design and Analysis

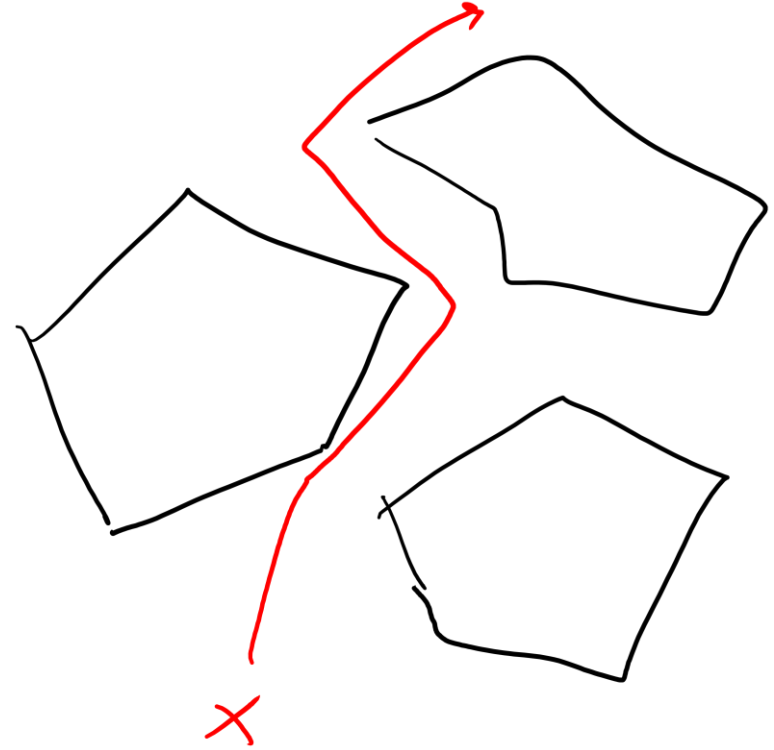
Computational Geometry (Fundamentals and the Convex Hull)

Goals for today

- Explore some fundamental tools for **computational geometry**
- Understand important tools/ideas such as:
 - **Dot and cross products**
 - The **line-side test**
- Define and solve the **convex hull** problem

Why geometry?

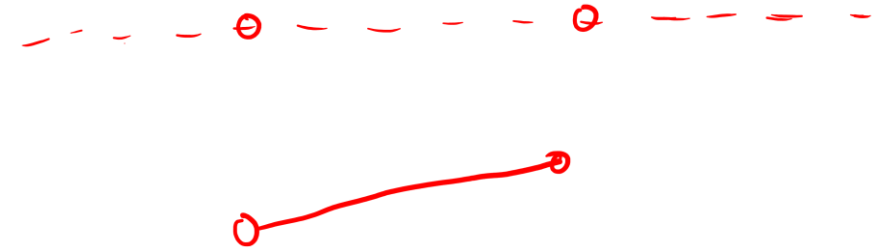
- Applications in robotics
- Applications to graphics
- Applications to algorithms (LPs!!)



Representation and Model

How might we represent some of the following ideas?

| | |
|---------------------|----------------------------------|
| Real number | Floating-point number |
| Point | A pair of floating-point numbers |
| Line | A pair of points |
| Line Segment | A pair of points |
| Triangle | A triple of points |



Concerns? Rounding Errors!

Solution? Be a theory class

Fundamental Objects & Operations

Representation (Point): A pair of real numbers

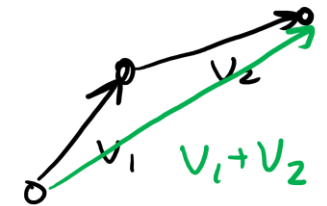
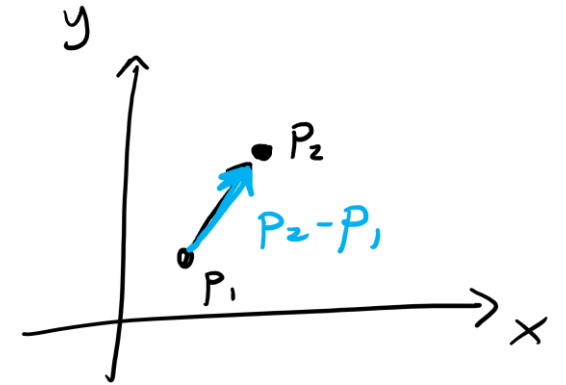
Representation (Vector): A pair of real numbers

We will use these interchangeably

Operation (Addition/subtraction):

$$(x_1, y_1) + (x_2, y_2) = (x_1 + x_2, y_1 + y_2)$$

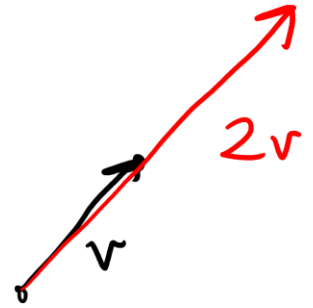
$$(x_1, y_1) - (x_2, y_2) = (x_1 - x_2, y_1 - y_2)$$



Fundamental Operations (continued)

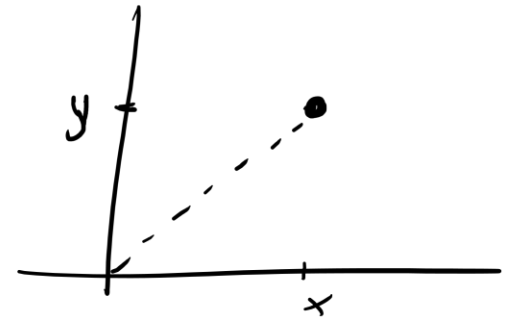
Operation (Scalar multiplication):

$$\alpha (x, y) = (\alpha x, \alpha y), \quad \alpha \in R$$



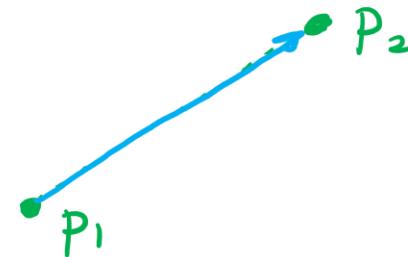
Operation (Length/magnitude):

$$||(x, y)|| = \sqrt{x^2 + y^2}$$



Application (Distance):

$$\text{dist}(p_1, p_2) = ||p_2 - p_1||$$



Fundamental Operations (continued)

Operation (The dot product):

$$(x_1, y_1) \cdot (x_2, y_2) = x_1x_2 + y_1y_2$$

Useful theorem (The dot product angle formula):

$$u \cdot v = \|u\| \|v\| \cos(\theta)$$

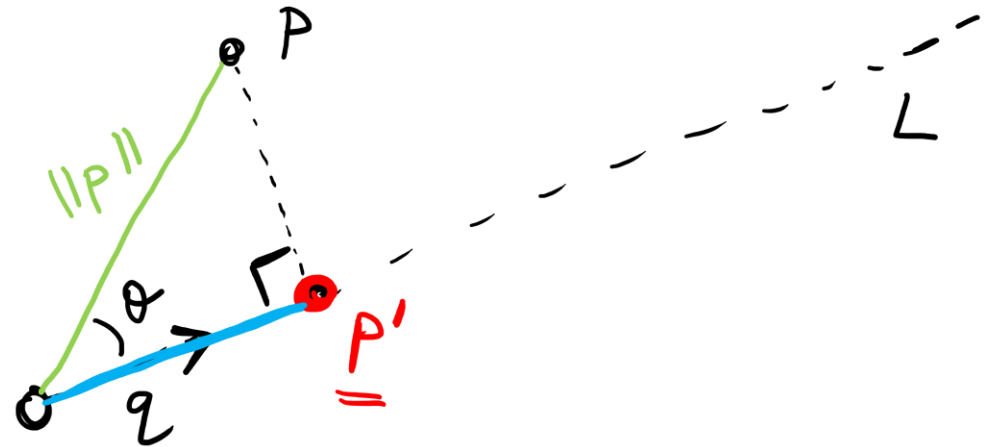
$$\theta = \cos^{-1} \left(\frac{u \cdot v}{\|u\| \|v\|} \right)$$

Application of the dot product

Application (Projection): Given a **point** p and a **line** L that goes through the origin in the direction of q (a unit vector), find the point p' on L that is closest to p

$$\begin{aligned} p \cdot q &= \|p\| \|q\| \cos(\theta) \\ &= \|p\| \cos(\theta) \\ &= \|p\| \cos(\theta) \end{aligned}$$

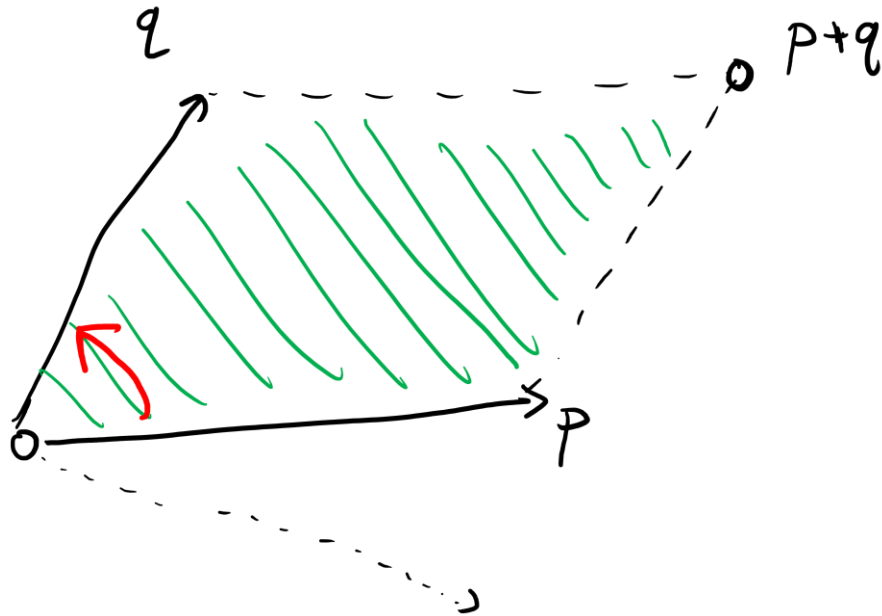
$$\underline{p'} = \underbrace{(p \cdot q)}_{\text{length}} \underbrace{q}_{\text{vector}}$$



Fundamental Operations (continued)

Operation (The cross product):

$$(x_1, y_1) \times (x_2, y_2) = \det \begin{pmatrix} x_1 & y_1 \\ x_2 & y_2 \end{pmatrix} = x_1 y_2 - x_2 y_1$$



$$|p \times q| = \text{AREA}$$

"Signed area"

- POSITIVE if q is to LEFT of p
- NEGATIVE if q is to right of p

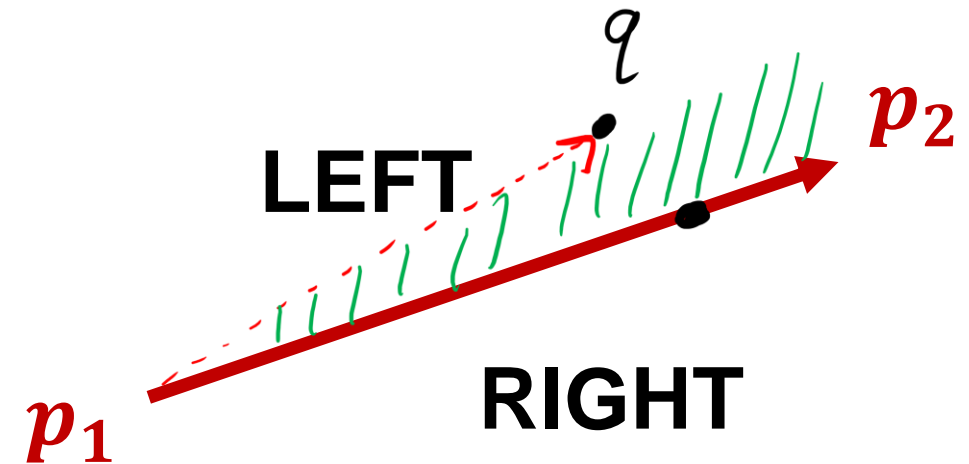
Line-side test (Important!)

Operation (Line-side test): Given points p_1, p_2, q , we want to know whether q is on the LEFT or RIGHT of the line from p_1 to p_2

$$V_1 = p_2 - p_1$$

$$V_2 = q - p_1$$

$$V_1 \times V_2 = \begin{cases} > 0 & \text{LEFT / CCW} \\ < 0 & \text{RIGHT / CW} \\ = 0 & \text{ON LINE} \end{cases}$$

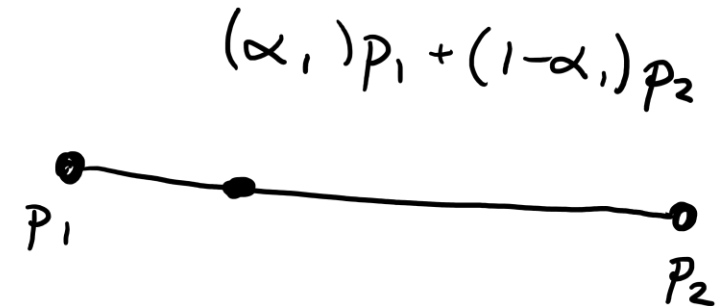
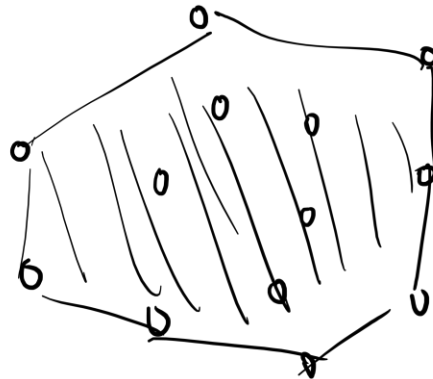
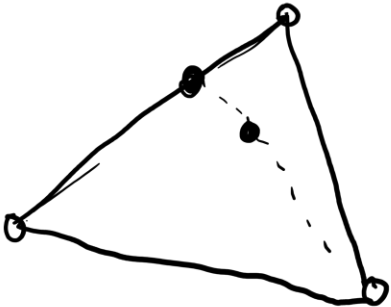


Convex Combinations

Definition (Convex combination): A *convex combination* of the points p_1, p_2, \dots, p_k is a point

$$p' = \sum_{i=1}^k \alpha_i p_i$$

such that $\sum \alpha_i = 1$ and $\alpha_i \geq 0$ for all i

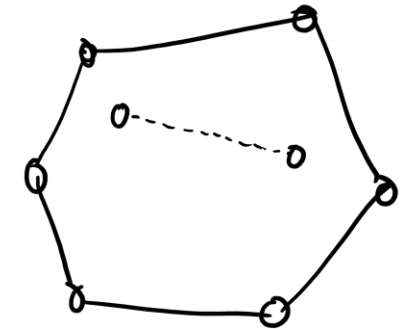
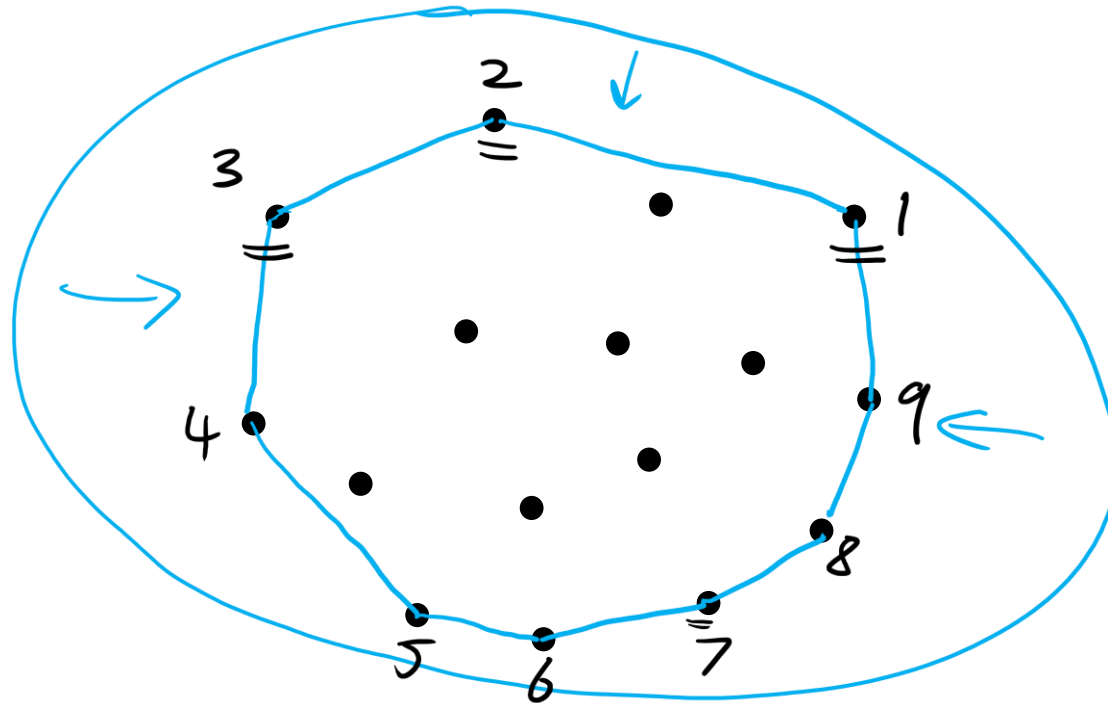
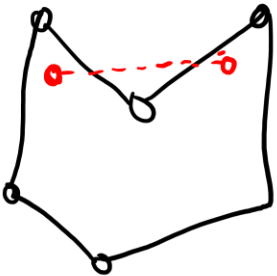


The Convex Hull

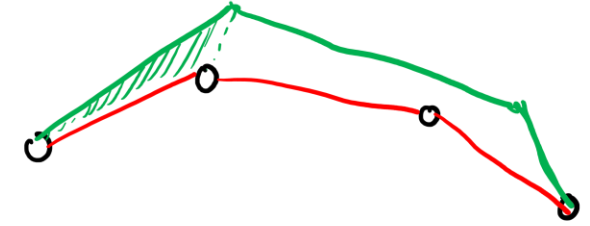
The Convex Hull

Definition (Convex hull): Given a set of points p_1, \dots, p_n , the **convex hull** is the smallest convex polygon containing all of them

Goal: output the vertices of the hull in counterclockwise order



An $O(n^3)$ -time algorithm

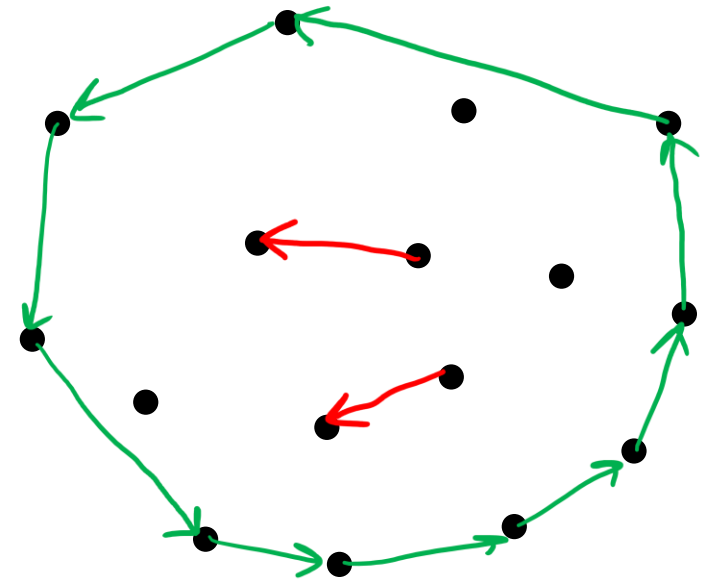


Observation (Hull edges): The edges of the convex hull must be pairs of points from the input

Claim (Hull edges): A segment (p_i, p_j) is on the convex hull if and only if...

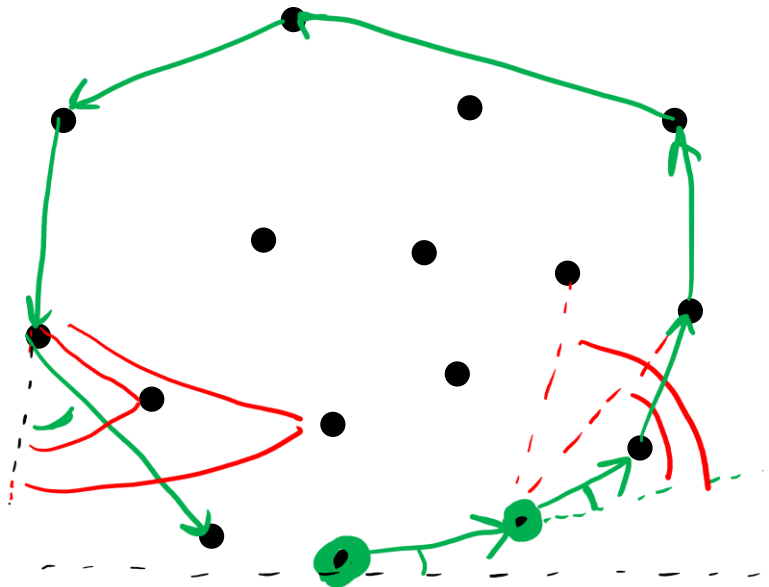
every other point is on the
LEFT of (p_i, p_j)

(Line-side test)



Better: An $O(n^2)$ -time algorithm

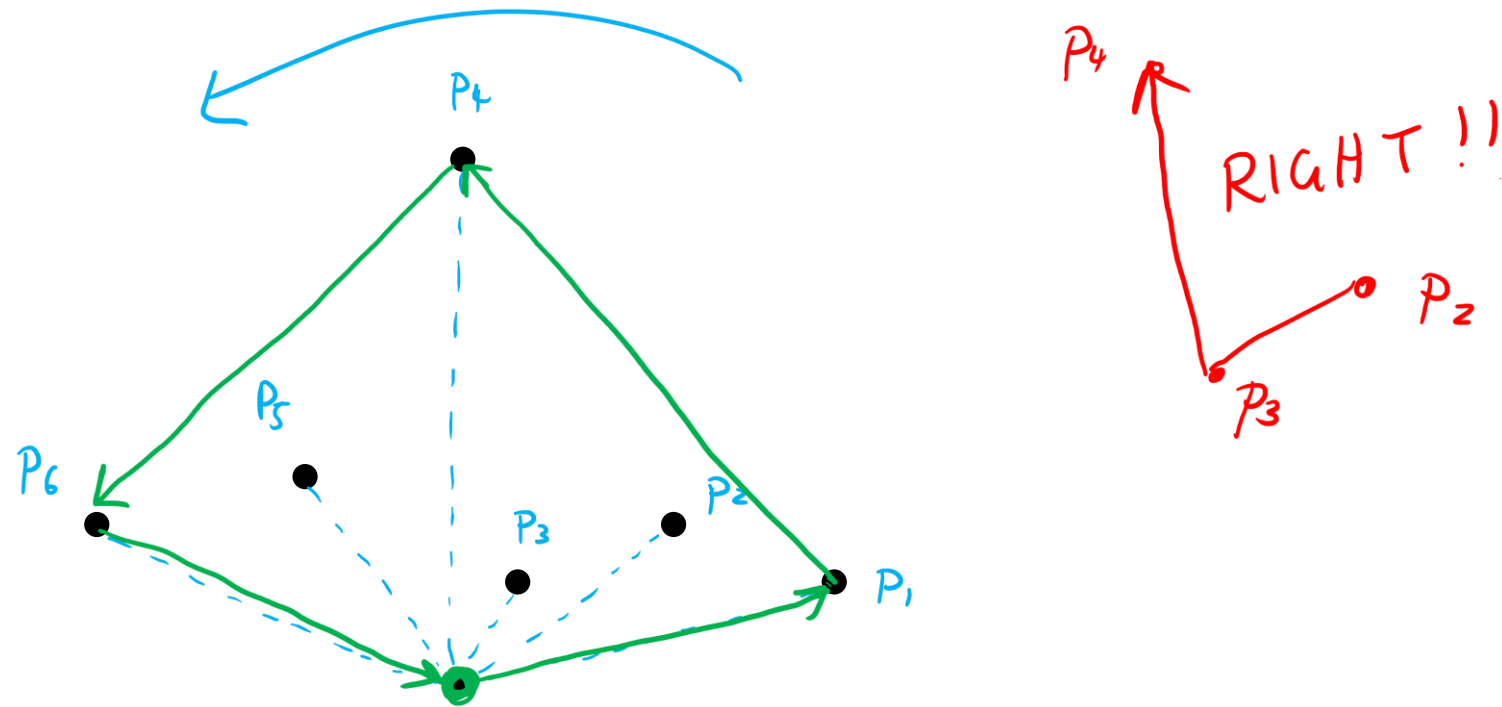
Observation (Order helps): The $O(n^3)$ -time algorithm found the hull edges in an arbitrary order... What if we try to find them in CCW order



Walk CCW around hull
making LEFT turns,
pick smallest angle

Graham Scan: An $O(n \log n)$ algorithm

Observation (Order helps again): We went from $O(n^3)$ to $O(n^2)$ by finding the edges in order... but we still processed the points in an arbitrary order. Can we order the points and do better?



Graham Scan: An $O(n \log n)$ algorithm

Algorithm (Graham Scan):

Find lowest point p_0

Sort points p_1, p_2, \dots counterclockwise by their angle with p_0

$H = [p_0, p_1]$

for each point $i = 2 \dots n - 1$

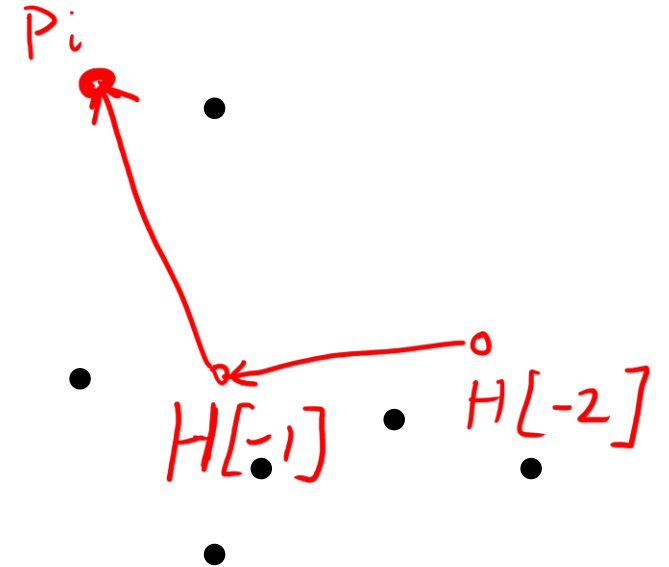
while $LST(H[-2], H[-1], p_i) == RIGHT$

$H.pop()$

$H.append(p_i)$

return H

$O(n \log n)$



Graham Scan: Complexity

Theorem: Graham Scan runs in $O(n \log n)$ time

Proof:

Sorting takes $O(n \log n)$

Each pt is only popped once $\rightarrow O(n)$

Lower Bound

Theorem: Any convex hull algorithm that uses line-side tests to find the hull requires $\Omega(n \log n)$ line-side tests (in a decision tree model)

Won't prove this

Take-home messages

- Computational geometry is all about using the right tools (and drawing good diagrams)
 - Dot product
 - Cross product
 - *Line-side test*
 - *Convex hull*