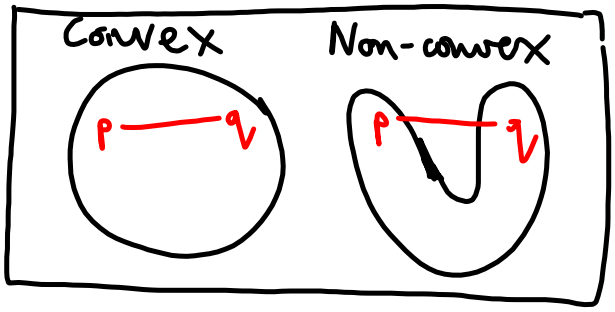
John Bourke bourkej@math.muni.cz

Geometrical algorithms

- Each week : 1 algorithm (computational geometry)
- Interconnected (eg. week 2 - sweep-line alg used later)
- E-Learning course: materials on IS
  Book : Computational geometry by deBerg...
- Exam at end

# Convex hull in the plane.

$K \subseteq \mathbb{R}^2$ (plane) is convex if for all $p, q \in K$ then the line segment $\overline{pq}$ is also in $K$.




$\overrightarrow{pq}$ consists of pts of form
$r = p + \lambda(q-p)$ for $\lambda \in [0,1]$
$\lambda = 0 \mapsto p$, $\lambda = 1 \mapsto q$
Equiv.
$r = (1-\lambda)p + \lambda q$ or
$r = \lambda p + (1-\lambda)q$

## Convex hull of a set P

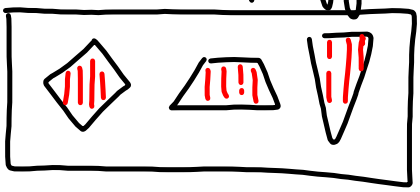
$CH(P) :=$ smallest convex set containing $P$

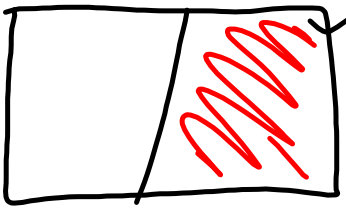
$$= \bigcap_{\substack{K \text{ convex} \\ P \subseteq K}} K$$

- Not computationally useful — infinitely many convex sets containing $P$.
- Goal: compute $CH(P)$ for $P$ finite.

P finite $\Rightarrow$ CH(P)

Convex polygons:

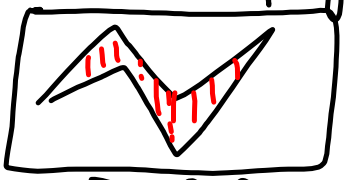




half-plane: points on 1 side of straight line

convex polygon: bounded by finitely many straight line segments.

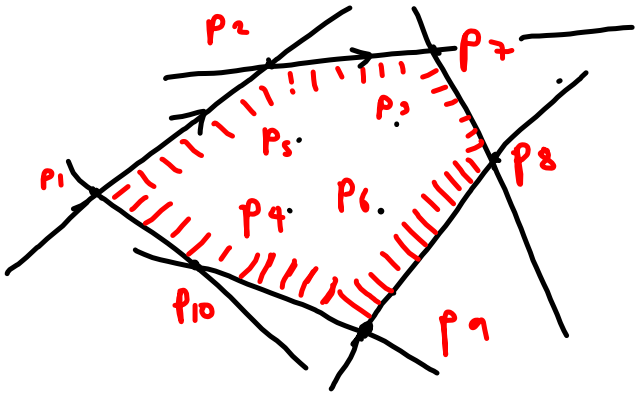
Non-convex polygons:



For P finite,

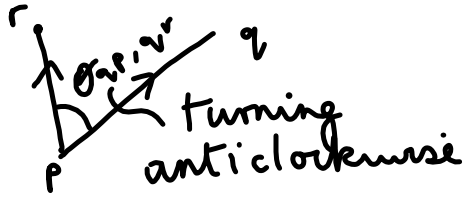
$CH(P) = \bigcap_{P \subseteq H} H$

a half-plane containing 2 points of P.




Idea: simple algorithm
- Search for directed line segments $\overrightarrow{pq}$ on convex hull in clockwise order:
eg $\overrightarrow{p_1p_2}$, $\overrightarrow{p_2p_7}$, ...
- Such a directed segment will lie on CH(P) if no point of P lies to its left.

When does a point r lie to the left of $\overrightarrow{pq}$?

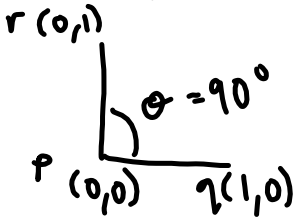



r lies to left of $\overrightarrow{pq}$

$\Leftrightarrow 0 < \theta_{qp,qr} < 180°$

This happens $\Leftrightarrow \det\begin{pmatrix} q_x - p_x & r_x - p_x \\ q_y - p_y & r_y - p_y \end{pmatrix} > 0$

eg.




$\det\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = 1$.

Algorithm: SlowConvexHull(P)

- For each pair $(p,q)$ test if no other point of $P$ lies to left of $\overrightarrow{pq}$.
- If so, add $\overrightarrow{pq}$ to our list $\mathcal{L}$.
- Sort clockwise (can be done in time $O(n \log n)$) ~ no of inputs

Complexity: - $n(n-1)$ pairs of distinct points
- For each pair, must check $n-2$ points (if they lie to left)

Complexity $O(n(n-1)(n-2) + n\log n) = O(n(n-1)(n-2)) = O(n^3)$.

## Complexity of algorithms:

- Algorithm $T$ det. function $T: \mathbb{N}_+ \longrightarrow \mathbb{R}^+$ – time
  ($\mathbb{N}_+$: no of inputs)

  $T(n)$ = max time it takes to run algorithm given $n$ inputs.
- Given $f, g: \mathbb{N}_+ \longrightarrow \mathbb{R}_+$ we write $f(x) = O(g(x))$ if $\exists\, c \in \mathbb{R}_+$ & $N \in \mathbb{N}_+$ s.t. $\forall n > N$

  $f(n) \leq c\, g(n)$
- Alg $T$ has complexity $O(g(n))$ if $T(n) = O(g(n))$
- $O(n), O(\log n), O(n^2), O(n^3), O(n \log n), \ldots$ $O(n^2 + 3n + 4) = O(n^2)$

Faster algorithm : Graham's scan $O(n \log n)$

- Order points in $P$ lexicographically
  (left to right, top to bottom)
  $p < q \iff p_x < q_x$ or $(p_x = q_x \ \& \ p_y > q_y)$
- Obtain ordered sequence $p_1 < p_2 < \dots < p_n$.


- $p_1, p_n$ lie on the convex hull
- Convex hull splits into an upper and a lower part.
- Search for upper hull & then lower.

- Let $\alpha_i$ = upper convex hull for $\{p_1, p_2, \ldots, p_{i+1}\}$.
- $\alpha_2 = (p_1, p_2)$
- Given $\alpha_i$ construct $\alpha_{i+1}$ : add $p_{i+1}$ which must belong to $\alpha_{i+1}$.
- Consider last 3 points in $\alpha_{i+1}$: $(p_j, p_i, p_{i+1})$. Say that they form a right turn if $p_{i+1}$ lies to right of line segment $\overrightarrow{p_j p_i}$ (ie. $\det(\,) < 0$)


- If they form a right turn, we stop.

- If they do not form a right turn

eg. 


delete the middle point $p_i$ from $L_{i+1}$.

Then look at last 3 points in $L_{i+1}$ and repeat this step until:

- last 3 points form a right turn or
- only two points remain in $L_{i+1}$.

- In this way, we obtain $L_{upper} = L_n$

- Vertices of lower hull calculated similarly:
  calc. lower hulls of sets $\{p_{n-1}, \dots, p_{n-1}, p_n\}$ for $i = 1$ to $i = n-1$.
  ie. $\{p_{n-1}, p_n\}$, $\{p_{n-2}, p_{n-1}, p_n\}$, $\dots$ $\{p_1, \dots, p_n\}$
- Finally append $d_{lower}$ to $d_{upper}$ (firstly deleting $p_1, p_n$ from $d_{lower}$)

- Time complexity : $O(n \log n)$ eg. mergesort
- Order n points lexicograph. takes time $O(n \log n)$
- On upper hull, a point is added & removed at most once - at most 2n actions.
- On lower hull - at most 2n.
- Append lists - constant.

$$O(4n + n \log n) = O(n \log n).$$

other algorithm : Gift wrapping
(Output sensitive)

- Complexity $O(nk)$
  - $n$: number of pts
  - $k$: no of vertices on convex hull
- Useful if $k$ small relative to $n$ as $O(kn) << O(n \log n)$ (see animation!)