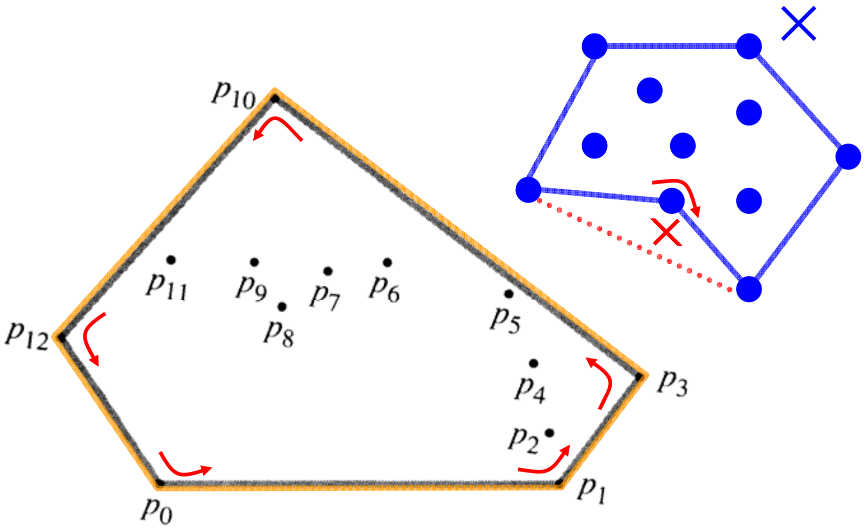
------------------------------ 33 ------------------------------
Computational Geometry

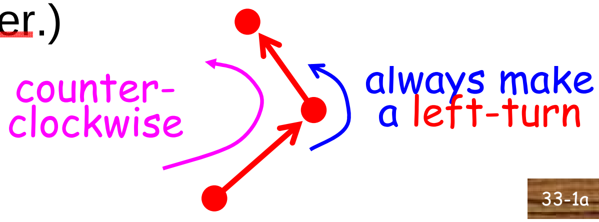
33.3 Finding the convex hull

The *convex hull* of a set *Q* of points is the smallest convex polygon *P* for which each point in *Q* is either on the boundary of *P* or in its interior.

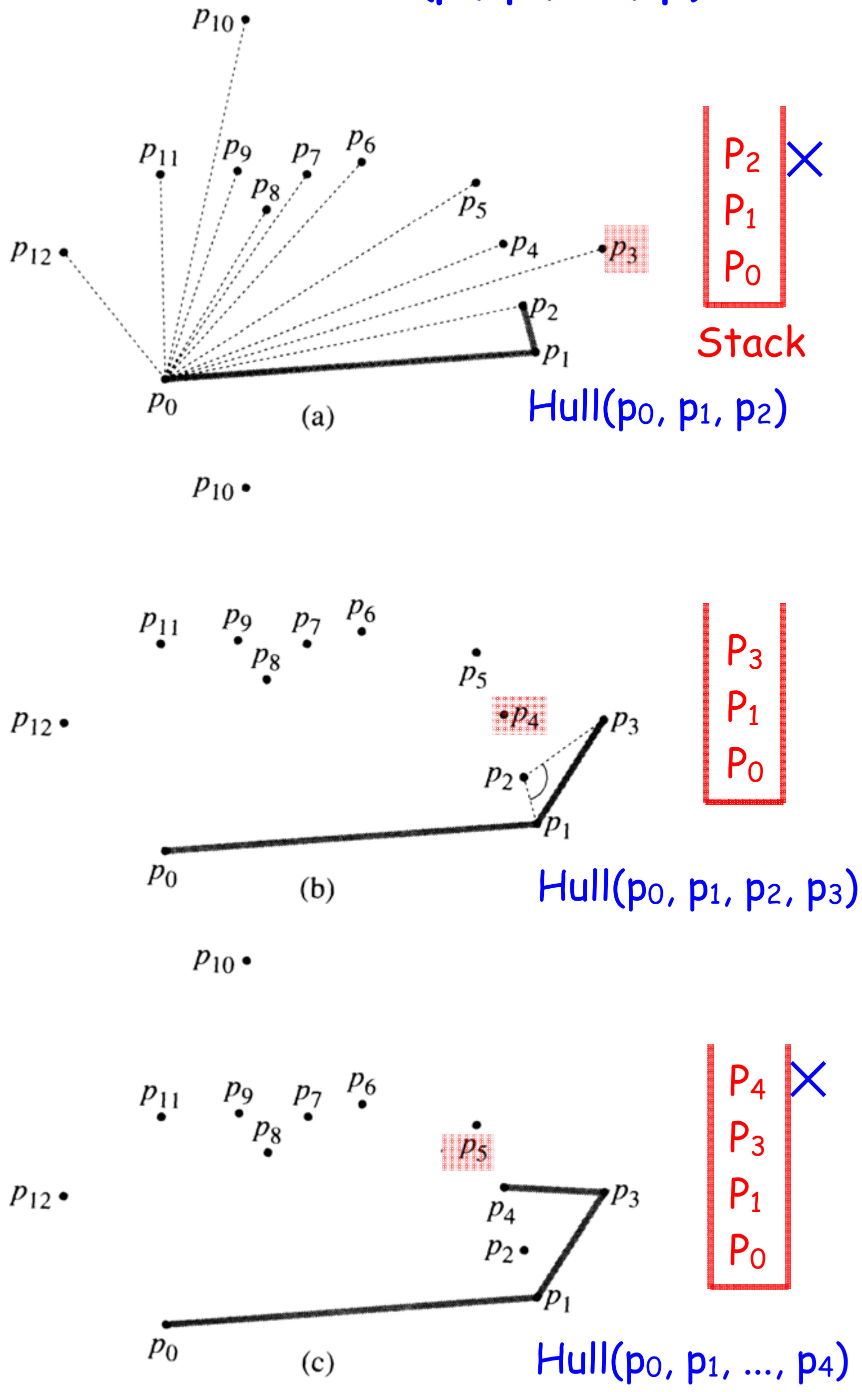


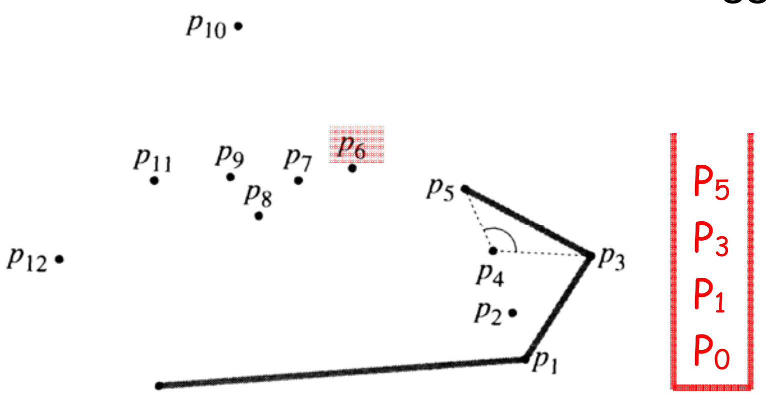
* A simple $O(n^3)$ solution: if $\overline{ab}$ is a convex edges, all other points are at the same side.

Graham's scan (Output the vertices in counterclockwise order.)



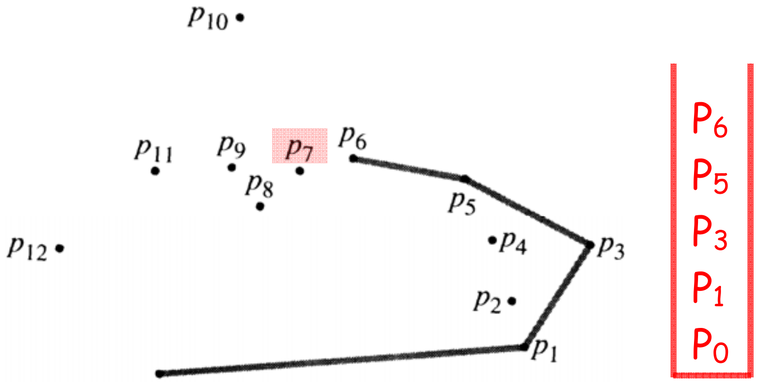
for i = 3 to n do
Hull(p_0, p_1, . . . , p_i)





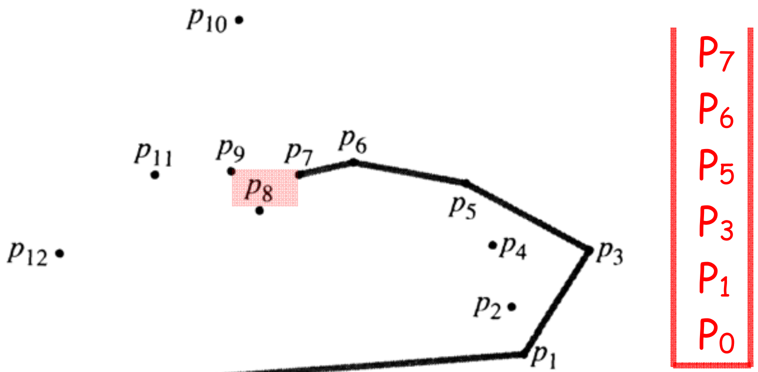
(d)

Hull(p_0, p_1, …, p_5)



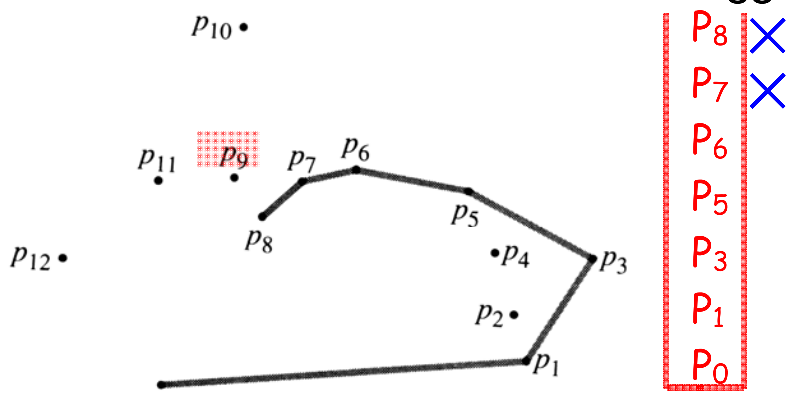
(e)

Hull(p_0, p_1, …, p_6)

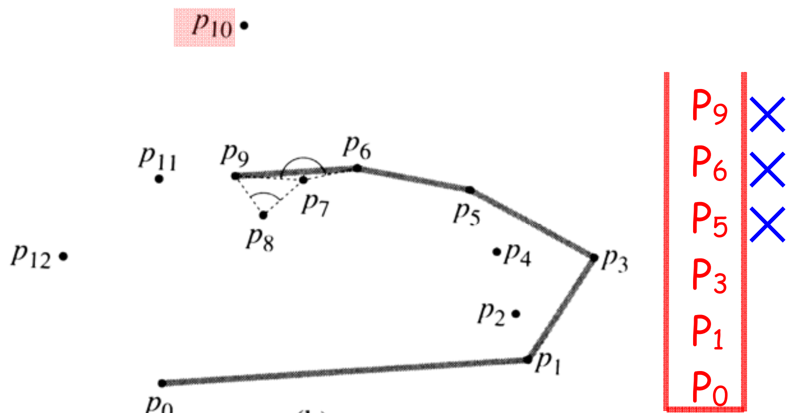


(f)

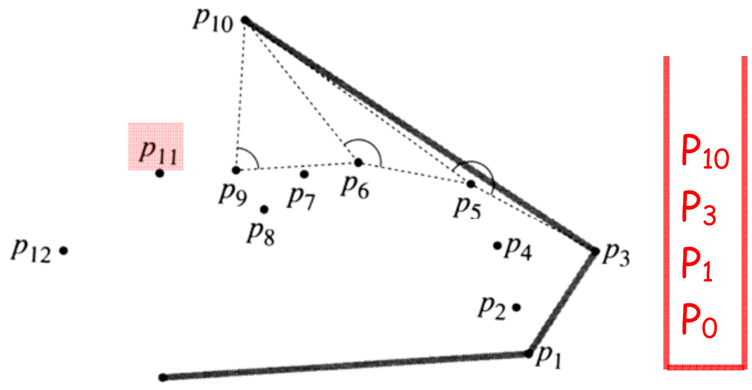
Hull(p_0, p_1, …, p_7)



(g)

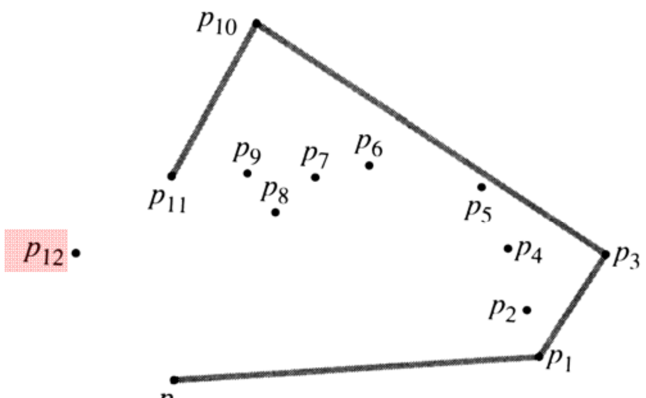


(h)

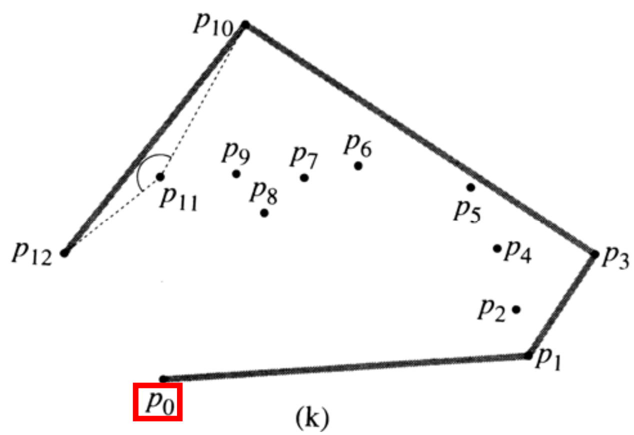


(i)

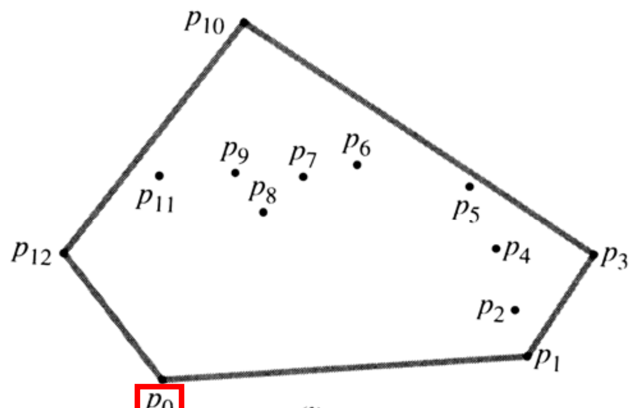
[redacted]



(j)



(k)



(l)

The end!

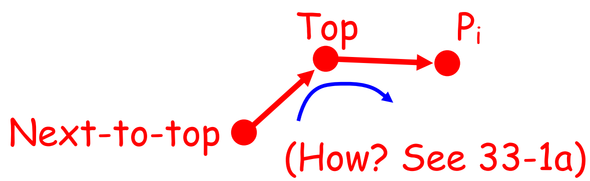
GRAHAM-SCAN(Q)

```
O(n)     1  let p0 be the point in Q with the minimum y-coordinate,
               or the leftmost such point in case of a tie
O(nlg n) 2  let <p1, p2, ..., pm> be the remaining points in Q,
               sorted by polar angle in counterclockwise order around p0
               (if more than one point has the same angle, remove all but
               the one that is farthest from p0)
         3  let S be an empty stack
         4  PUSH(p0, S)
         5  PUSH(p1, S)
O(n)     6  PUSH(p2, S)
         7  for i = 3 to m
         8      while the angle formed by points NEXT-TO-TOP(S),
                      TOP(S), and pi makes a nonleft turn
         9          POP(S)
        10      PUSH(pi, S)
        11  return S
```

(How? See 33-1b)

function compare(p_0, a, b)
$\begin{cases} 0: \theta_a \le \theta_b \\ 1: \theta_a > \theta_b \end{cases}$

multi-pop + push (amortized)



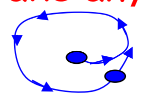
Time complexity:

Line 1: $O(n)$ find P_0
Line 2: ~~$O(n\lg n)$~~ $P_1, P_2, P_3, \ldots, P_m$ $(m \le n-1)$
Line 3~6: $O(1)$
Line 8~9: $O(n)$ Amortized
(at most n PUSH $\rightarrow$ at most n POP)
Line 10: $O(n)$

$\Rightarrow O(n\lg n)$

* if the sorted order is given : $O(n)$

* take any as p_0:



33-6x

* **Correctness**: See textbook.

33-7x

Jarvis's March (using a technique known as ***package wrapping*** or ***gift wrapping***)
(Find a correct edge each time !)

Step 1: Find the lowest point x and the highest point y.

Step 2: Compute the right chain
($p_0=x$, p_1, …, $p_k=y$).

Step 3: Compute the left chain
($p_k=y$, p_{k+1}, …, $p_h=x$).

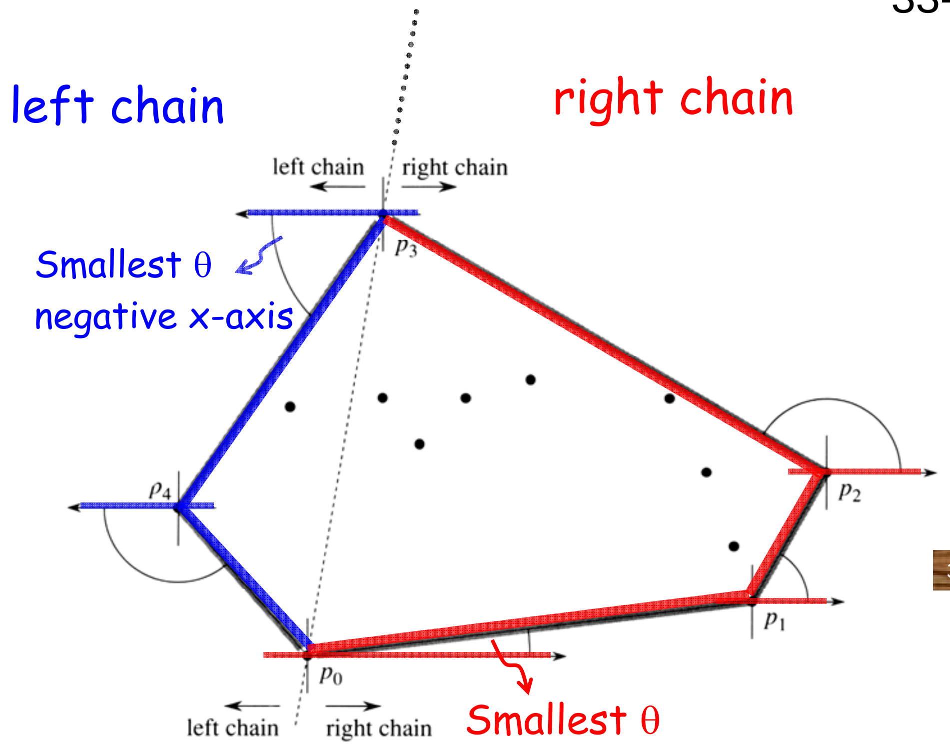
* In $O(1)$ time, we can compare the polar angles of two points. (How ??? See Section 33.1)
function compare(p_0, a, b): $\begin{cases} 0: \theta_a \le \theta_b \\ 1: \theta_a > \theta_b \end{cases}$

* In $O(n)$ time, we can determine the point with smallest (or largest) polar angle with respect to a given point. (How? See 33-1b)

* Since each computation of p_i take $O(n)$ time, $T(n)=O(nh)$. → # of convex hull vertices

* In the worst case, $h=n$ and thus $T(n)=O(n^2)$.

* Jarvis's march is better than Graham's scan if $h=o(\lg n)$.



33-8x

Figure 33.9 The operation of Jarvis's march. The first vertex chosen is the lowest point p next vertex, p_1, has the smallest polar angle of any point with respect to p_0. Then, p_2 smallest polar angle with respect to p_1. The right chain goes as high as the highest point p_3. the left chain is constructed by finding smallest polar angles with respect to the negative x-ax

Homework: None.

conclude