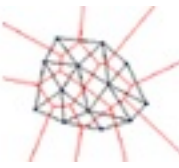
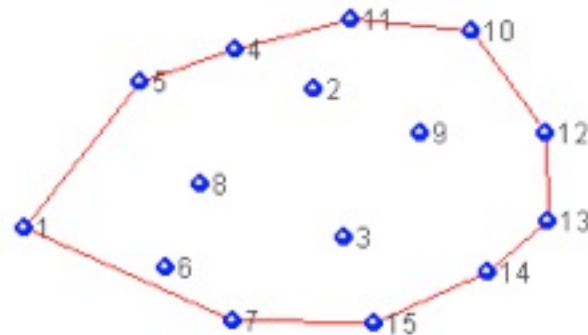


Convex hull

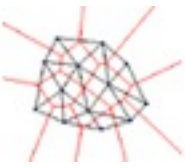


- Definición, aplicaciones y propiedades
- Algoritmos triviales
- Gift wrapping
- Quick hull
- Algoritmo de Graham
- Algoritmo incremental

Definición

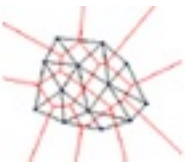


- Dado un conjunto S de puntos, su convex hull es el polígono convexo más pequeño que incluye a todos los puntos de S



- Planificación de movimientos sin colisiones
- Optimización: investigación operativa
- Análisis de forma

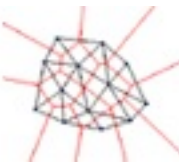
Conjunto convexo



- Un conjunto S es convexo, si, para cualquier $x, y \in S$

se cumple que $\overline{xy} \in S$

Combinación convexa



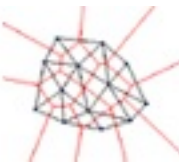
- Una combinación convexa de los puntos x_1, \dots, x_n es un punto que cumple

$$\alpha_1 x_1 + \dots + \alpha_n x_n$$

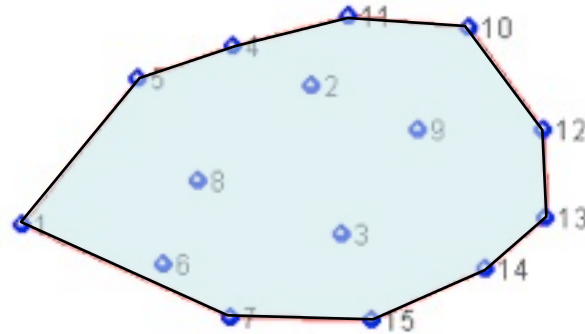
y en donde $\alpha_i > 0$ y

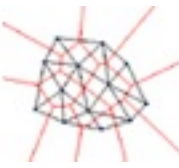
$$\alpha_1 + \dots + \alpha_n = 1$$

Combinación convexa y RC

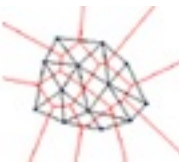


- El recubrimiento convexo de x_1, \dots, x_n es el conjunto de todas sus combinaciones convexas





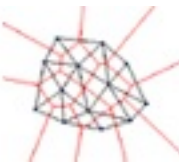
- El RC de S es la unión de todos los triángulos determinados por puntos de S
- El RC de S es la intersección de todos los semiespacios que contienen S
- Un punto de S es un vértice del RC si no existe ningún triángulo determinado por puntos de S que lo incluya



- Determinación de puntos extremos

1. Para cada i hacer
2. Para cada $j \neq i$ hacer
3. Para cada $k \neq i \neq j$ hacer
4. Para cada $h \neq i \neq k \neq j$ hacer
5. Si P_h está a la izqda de (P_i, P_j) y
6. P_h está a la izqda de (P_j, P_k) y
7. P_h está a la izqda de (P_k, P_i)
8. entonces P_h no es extremo

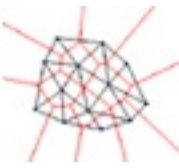
▪ **Complejidad: $O(n^4)$**



- Determinación de aristas extremas
 1. Para cada i hacer
 2. Para cada $j \neq i$ hacer
 3. Para cada $k \neq i \neq j$ hacer
 4. Si P_k no está a la izqda de (P_i, P_j)
 5. entonces (P_i, P_j) no es extremo

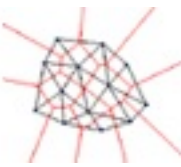
▪ Complejidad: $O(n^3)$

Gift Wrapping

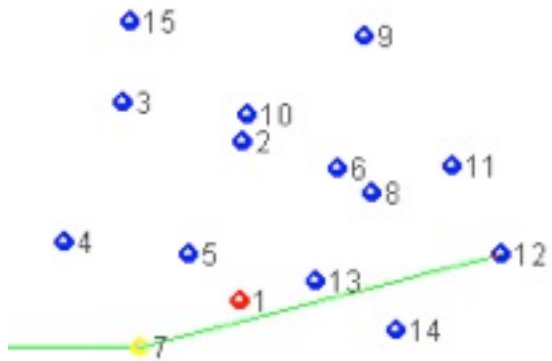


1. Encontrar punto p más pequeño en coord. y , sea i_0 su índice
2. $i := i_0$
3. Repetir
4. Para cada $j \neq i$ hacer
5. Calcular el ángulo en sentido antihorario entre P_j y la arista anterior del RC
6. Sea k el índice del punto con ángulo menor
7. Marcar (P_i, P_k) como una arista del RC
8. $i := k$
9. Hasta que $i = i_0$

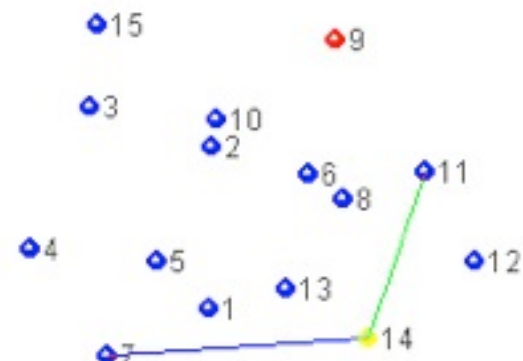
Gift Wrapping (ejemplo)



1



2



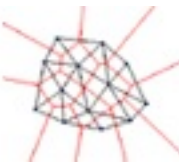
3



4



QuickHull



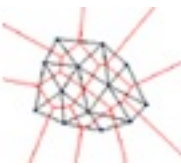
QuickHull(a, b, S)

1. Si $S = \{a, b\}$ entonces devolver (a, b)
2. sino
3. $c :=$ índice del punto con máxima distancia a (a, b)
4. $A :=$ puntos a la derecha de (a, c)
5. $B :=$ puntos a la derecha de (c, b)
6. devolver
concatenar(QuickHull(a, c, A), QuickHull(c, b, B))

ConvexHull(S)

1. $a :=$ punto más a la derecha de S
2. $b :=$ punto más a la izquierda de S
3. devolver concatenar(QuickHull(a, b, S), QuickHull(b, a, S))

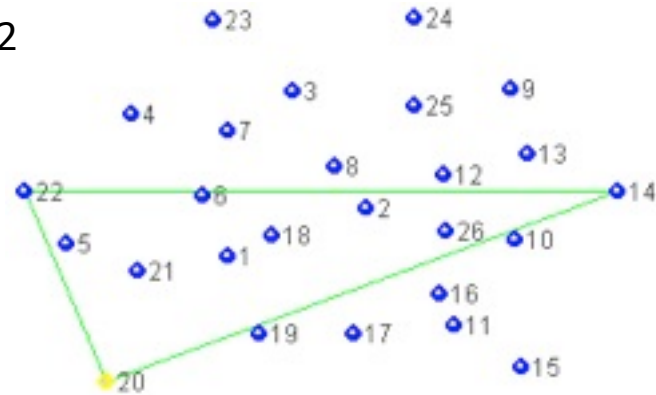
QuickHull



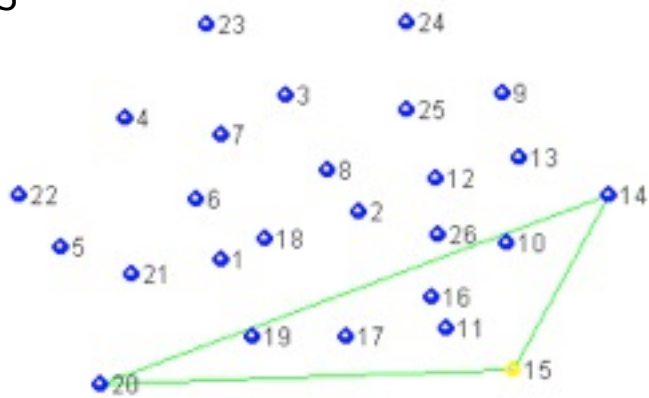
1



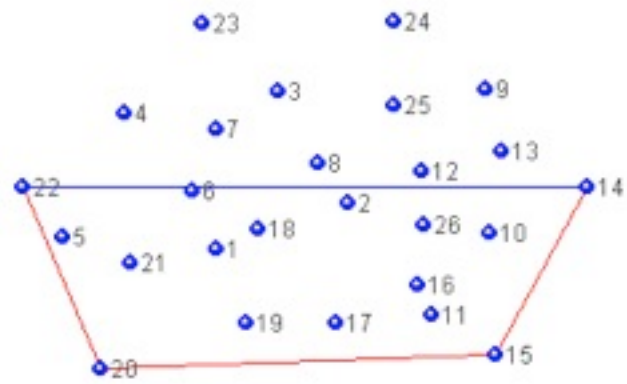
2



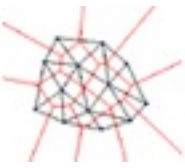
3



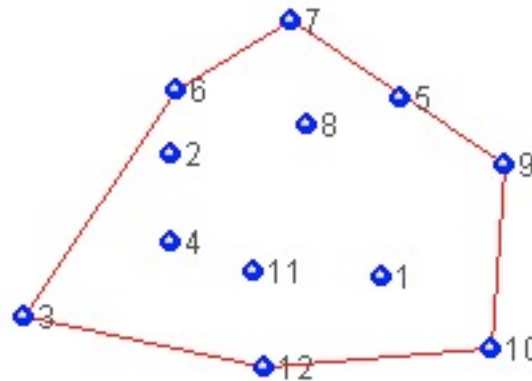
4



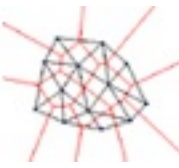
Propiedad del RC



- Si un punto no es un vértice del RC, entonces es interno a algún triángulo (O_pq), donde O es un punto cualquiera interior al RC, y p y q son vértices consecutivos del RC

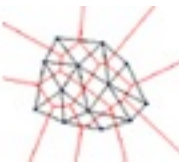


Scan de Graham



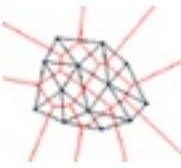
- Ordenación de los puntos alrededor de un punto interior
- Barrido de forma ordenada, eliminando los puntos internos, que cumplen la condición anterior

Algoritmo Scan de Graham

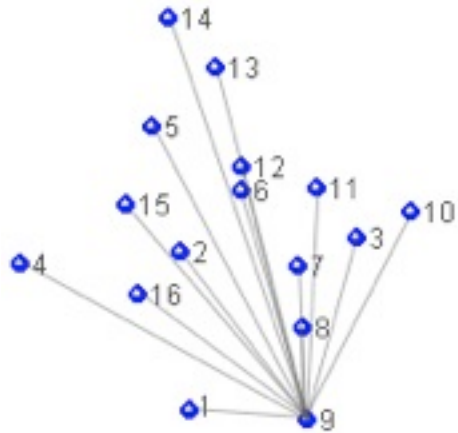


1. Encontrar el punto inferior más a la derecha (P_0)
2. Ordenar todos los puntos angularmente alrededor de P_0 ; aquellos puntos que estén en el mismo ángulo colocar primero el más cercano a P_0 : etiquetarlos P_1, \dots, P_{n-1}
3. Pila $S := (P_{n-1}, P_0)$
4. $t :=$ tope de S (P_0)
5. $i := 1$
6. Mientras que $i < n$ hacer
7. Si P_i está estrictamente a la izqda de (P_{t-1}, P_t)
8. entonces $\text{Push}(S, i)$; $i := i + 1$
9. sino $\text{Pop}(S)$

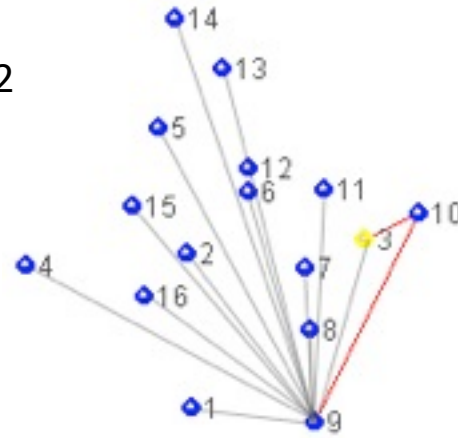
Ejemplo



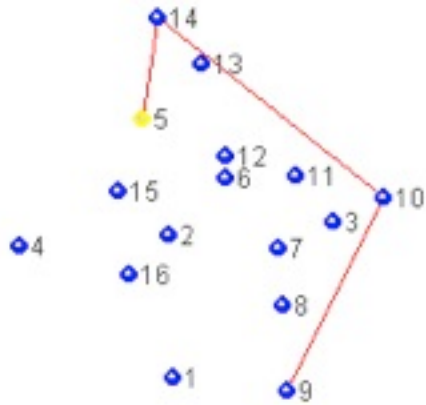
1



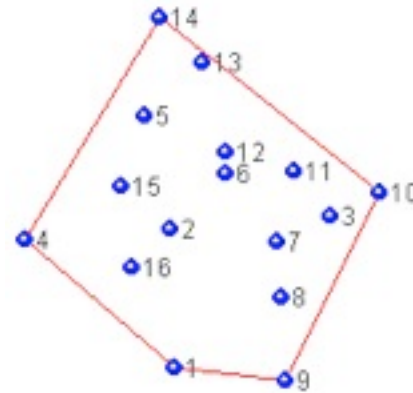
2



3



4





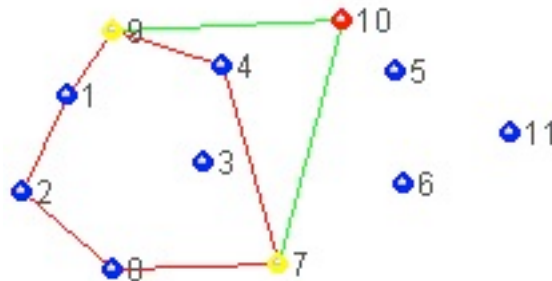
- Interés en procesar los puntos que han llegado hasta el momento, manteniendo la estructura actualizada hasta el último punto recibido
- La complejidad del algoritmo incremental es $nO(u)$, siendo $O(u)$ la complejidad de actualización y n el número de puntos

Algoritmo RC incremental

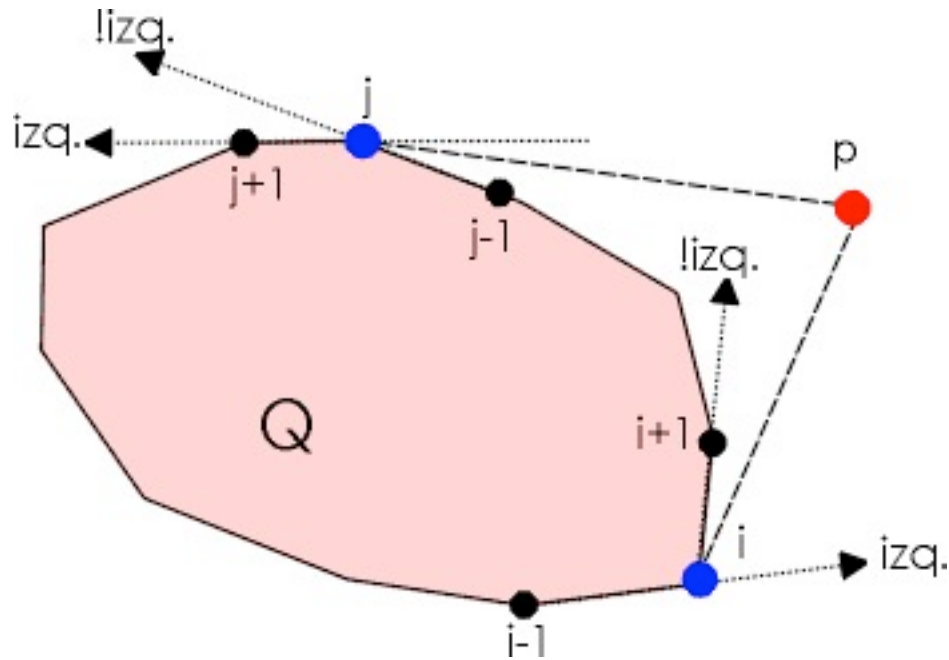
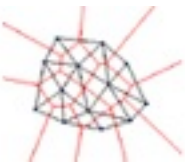


Incremental(RC,p)

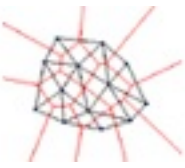
1. Si p está dentro de RC
2. entonces devolver RC
3. sino
4. $(Pt1, Pt2) := \text{puntosTangentes}(RC, p)$
5. Eliminar aristas en RC de $Pt1$ a $Pt2$
6. Conectar $Pt1$ con p y p con $Pt2$



Puntos tangentes



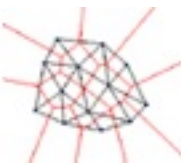
Aplicación a un mapa de bits



- Un ejemplo de aplicación: delimitación de objetos



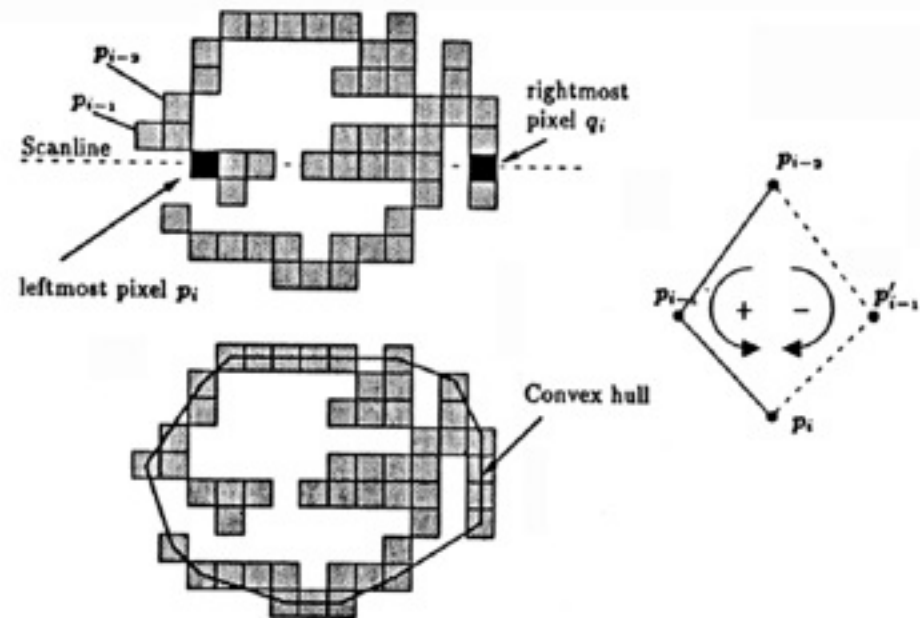
Algoritmo para mapa de bits



- Se realiza un barrido de arriba a abajo, seleccionando los puntos más a la derecha e izquierda de cada línea. Eliminar p_i si

$$D_i = \begin{cases} \leq 0 & : p_i \text{ en el lado izquierdo} \\ \geq 0 & : p_i \text{ en el lado derecho} \end{cases}$$

$$D_i = \begin{vmatrix} x_{i-2} & y_{i-2} & 1 \\ x_{i-1} & y_{i-1} & 1 \\ x_i & y_i & 1 \end{vmatrix}$$



Extensión para 3 dimensiones

