Introducción a la Geometría Computacional

Análisis de Algoritmos

Geometría Computacional

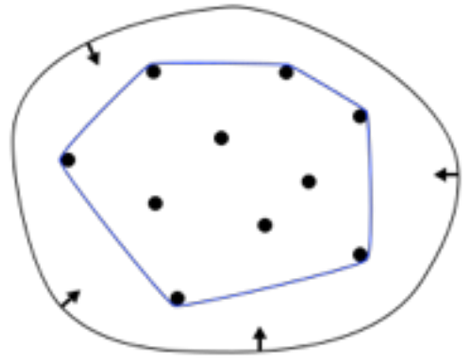
- La Geometría Computacional surgió a finales de los 70s del área de diseño y análisis de algoritmos.
- Estudio sistemático de algoritmos y estructuras de datos de objetos geométricos, con un enfoque en algoritmos exactos (deterministas).
- Reúne problemas matemáticos y de ciencias de la computación.
- Aplicaciones a numerosas áreas como: gráficos por computadora, sistemas de información geográficos, robótica, visión por computadora, etc.
- Una buena solución al problema de Geometría Computacional requiere de una buena comprensión de la estructura geométrica del problema y conocimiento profundo de algorítmica y estructuras de datos.

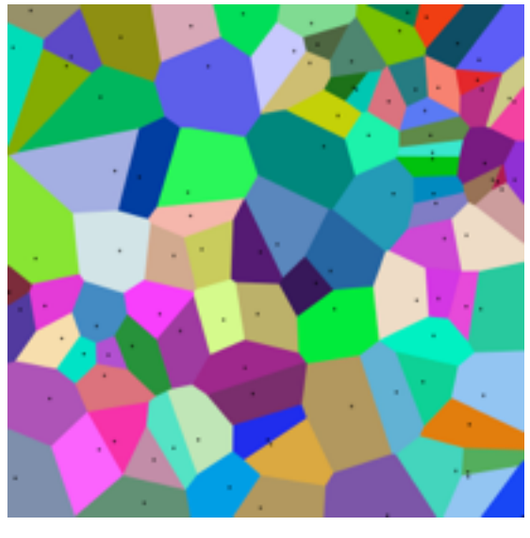
Geometría Computacional

- Estudiaremos algunos algoritmos de geometría computacional combinatoria o geometría algorítmica:
  - Considerar a los objetos como entidades discretas.
  - Entrada:
    - descripción de un conjunto de objetos geométricos: puntos, segmentos de recta, vértices de polígonos...
  - Salida:
    - respuesta al problema, por ejemplo si un par de segmentos de recta intersecan, cuántas intersecciones hay... o un nuevo objeto geométrico.

Clases de problemas

- Problemas estáticos:
  - Intersección de segmentos de recta:
    - Dado un conjunto de segmentos de recta, encontrar las intersecciones entre ellos.
  - Convex hull (envolvente convexo)
    - Dado un conjunto de puntos, encontrar el poliedro o polígono más pequeño que los contenga.
  - Triangulación de Polígonos, Triangulación de Delaunay.
  - Diagrama de Voronoi.
  - Algoritmos de visibilidad





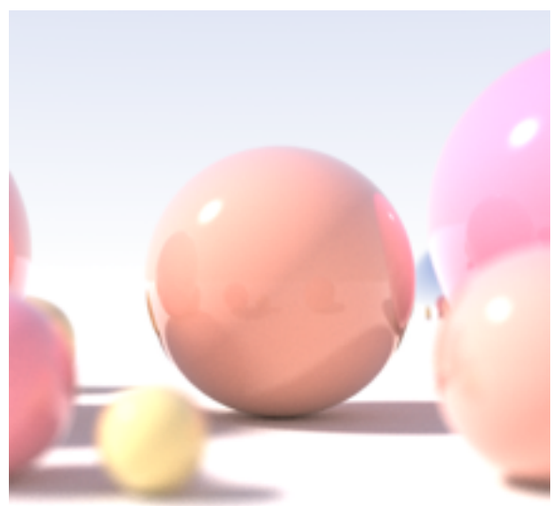
Problemas estáticos

- La complejidad computacional para este tipo de problemas se estima generalmente con el tiempo de cálculo y el espacio (memoria) requerida para resolver el problema.

Clases de problemas

- Problemas de búsqueda geométrica
  - Búsqueda de rango:
    - Dada una región del espacio, pre-procesar un conjunto de puntos para contar de manera eficiente cuántos hay en la región.

  - Localización de un punto:
    - Dada una partición del espacio en celdas, calcular eficientemente en qué celda se localiza el punto.



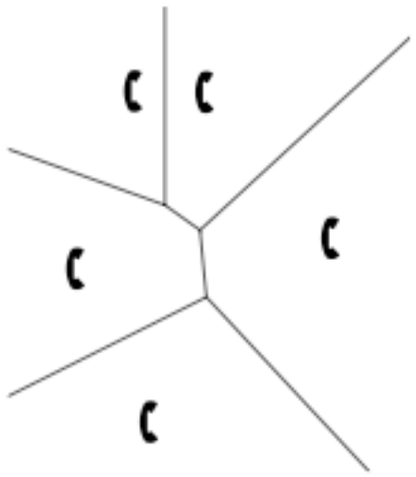
  - Vecino más cercano.
  - Trazado de rayos

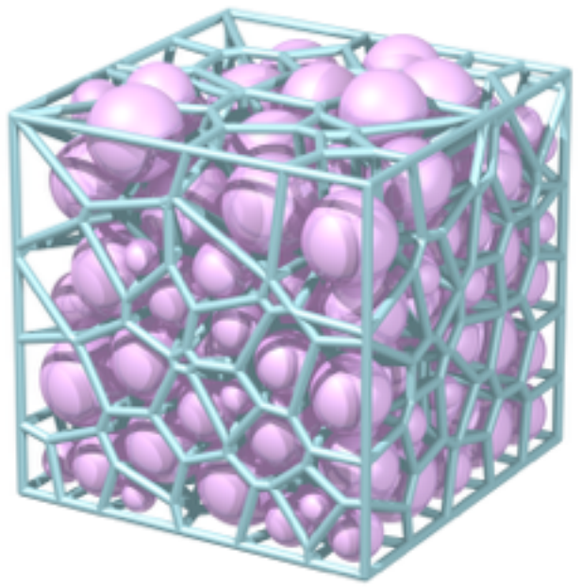
Problemas de búsqueda geométrica

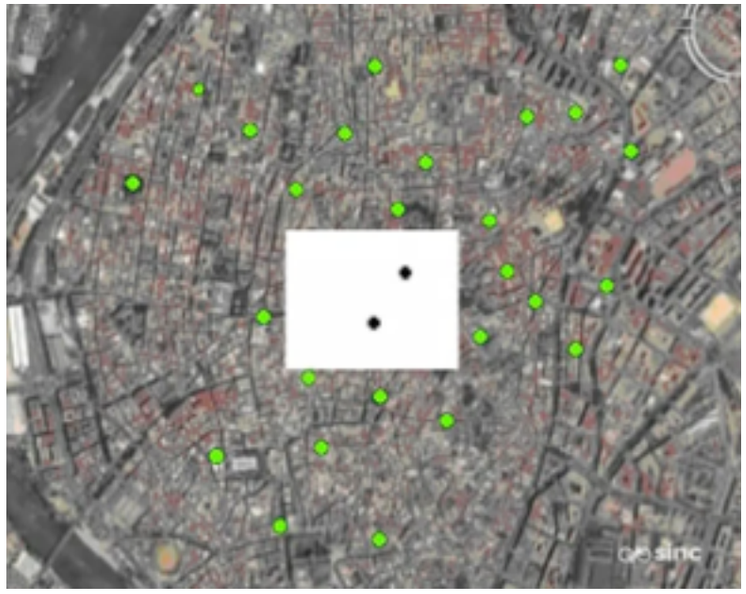
- Se genera un espacio de búsqueda (exploración) y luego se busca una solución en este.
- Si el espacio de búsqueda está fijo, la complejidad computacional se estima con el tiempo y el espacio requerido para construir la estructura de datos donde se realizará la búsqueda además del tiempo y la memoria que toma encontrar la solución.

Ejemplo 1: Diagrama de Voronoi

Ejemplo: Aplicación del Diagrama de Voronoi, metro de Sevilla

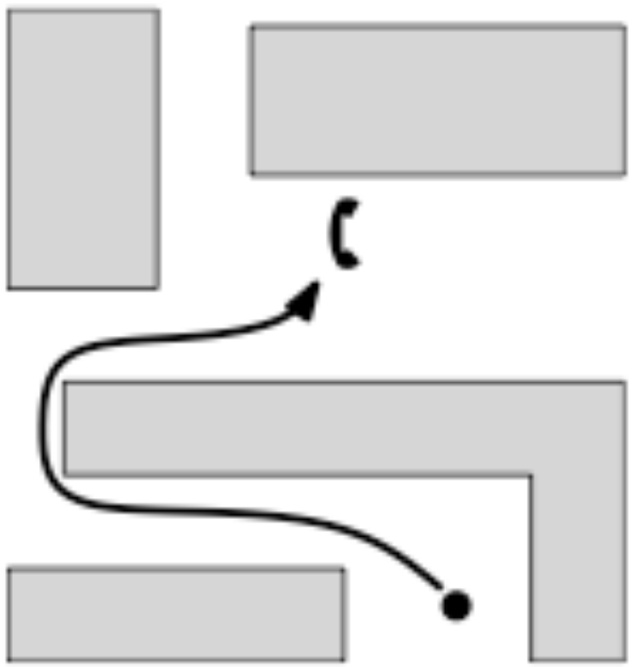


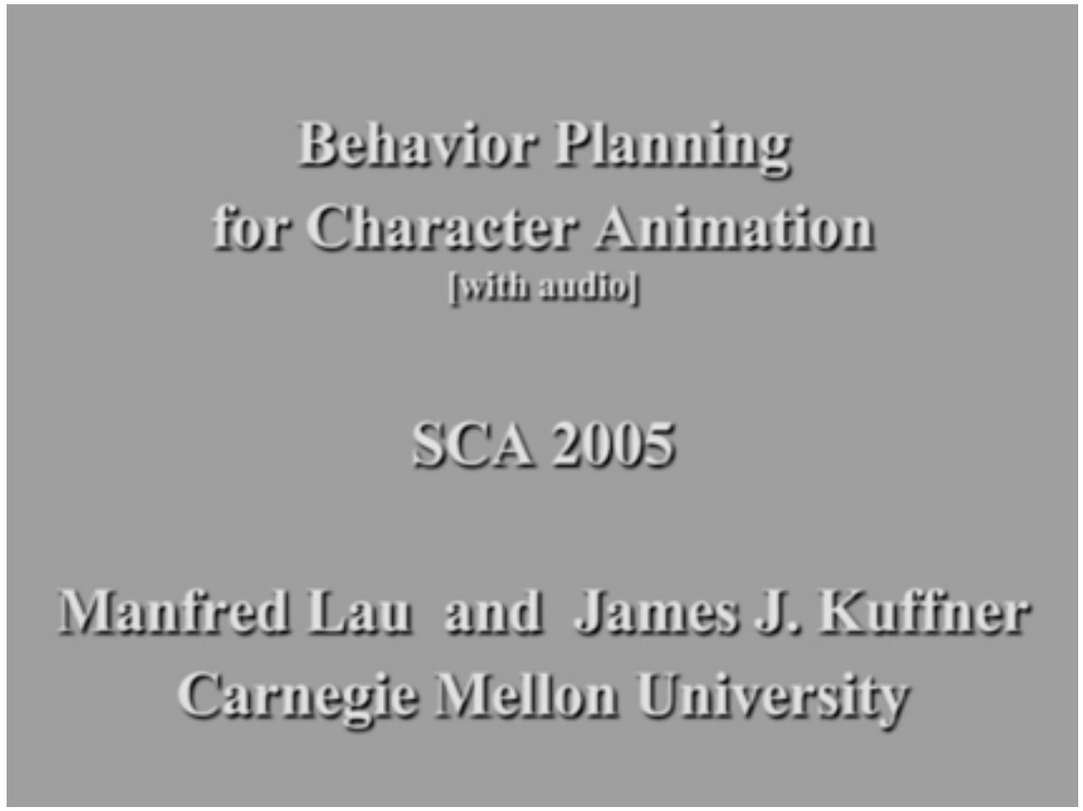




Ejemplo 2: Planificación de movimientos

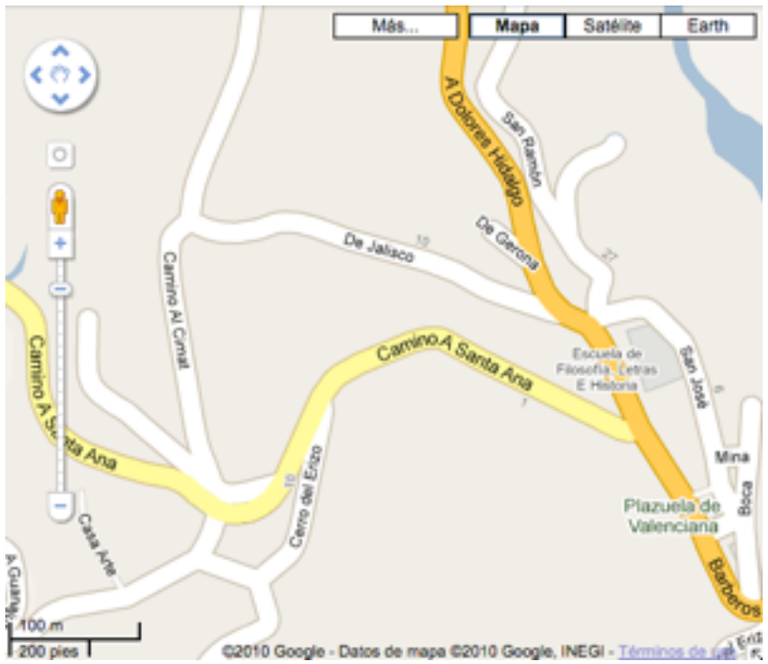
Dada una colección de obstáculos geométricos, encontrar una conexión más corta entre dos puntos, evitando colisiones con los obstáculos.



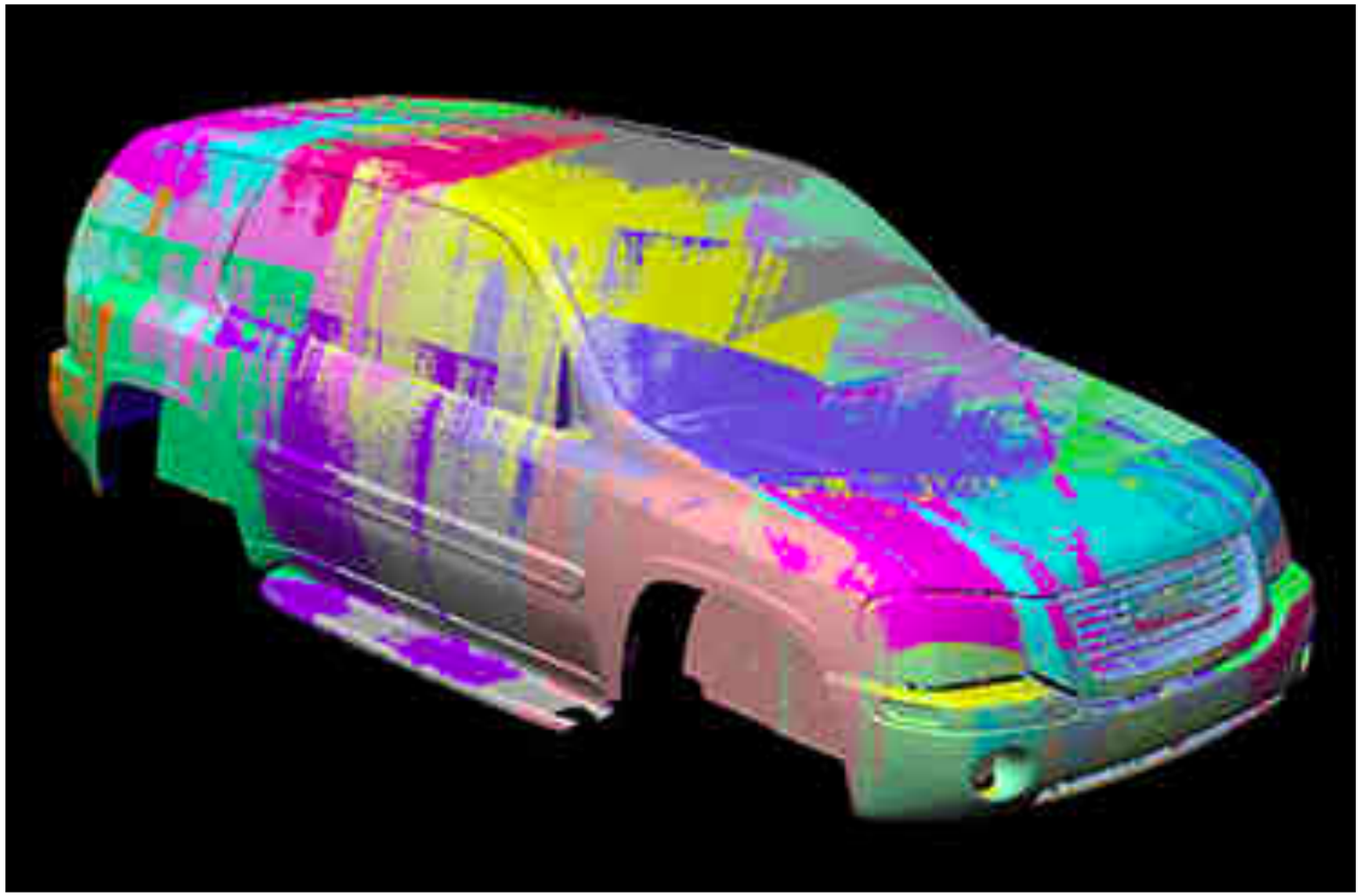


Ejemplo 3: Mapas sobrepuestos

Utilizado en sistemas de información geográfica - encontrar las correspondencias.



Ejemplo 3: Mapas sobrepuestos

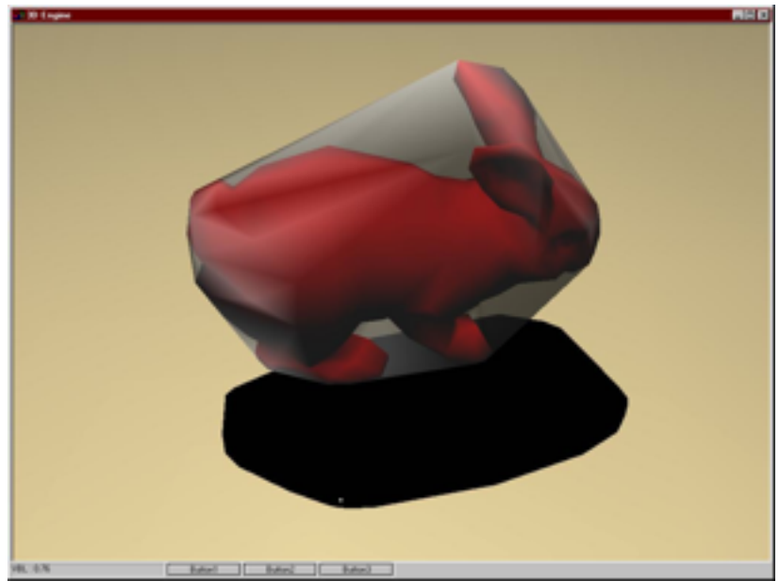


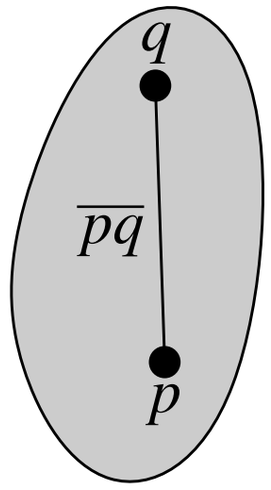
Envolvente Convexo

comp-420

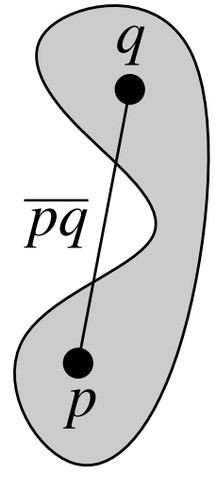
Envolvente convexo

- Uno de los primeros problemas estudiados en la geometría computacional.
- Un conjunto S del plano, se llama convexo si y solo si, para cualquier par de puntos p,q ∈ S, el segmento pq está contenido completamente en S.
- El envolvente convexo CH(S) de un conjunto S es el conjunto convexo más pequeño que pueda contener a S.
- Es la intersección de todos los conjuntos que contienen a S.





convexo



no convexo

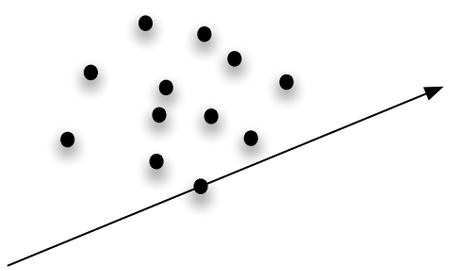
Medios planos (halfplanes)

- Los *medios planos* son importantes para estudiar convexidad.
- Un *medio plano cerrado* H se puede escribir como:

$$H = \{(p_x, p_y) | ap_x + bp_y \geq c\}$$

para números reales *a,b* y *c*. Esto incluye la línea que separa los planos.

- En un *medio plano abierto* esta desigualdad es estricta y ningún punto de la línea está incluido en el plano.
- Una línea l es una *línea de soporte* para un conjunto S si l define un medio plano que contenga S .
- Una línea de soporte l es tangente a S si contiene un punto de S como en la figura.

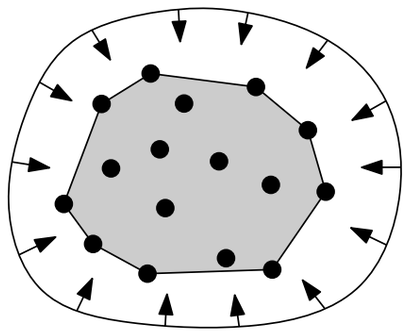


Polígonos

- Un *polígono* es un conjunto en el plano
- cuya frontera se puede describir como una secuencia circular de puntos, llamados vértices $p_0, p_1, \ldots, p_k$, donde $p_0 = p_k$,
- y segmentos de recta llamados aristas que unen vértices consecutivos $(p_0, p_1), (p_1, p_2), \ldots, (p_{k-1}, p_k)$.
- Un *polígono* es *simple* si las únicas intersecciones entre sus aristas ocurren donde aristas adyacentes comparten un mismo vértice.
- Un *polígono* es una región en el plano acotada por un ciclo de segmentos de recta, llamados aristas, unidos en sus extremos formando un ciclo.
- Los puntos donde se unen dos aristas sucesivas se llaman *vértices*.

Envolvente convexo (convex hull)

- Calcular el envolvente convexo de un conjunto finito P de n puntos en el plano.



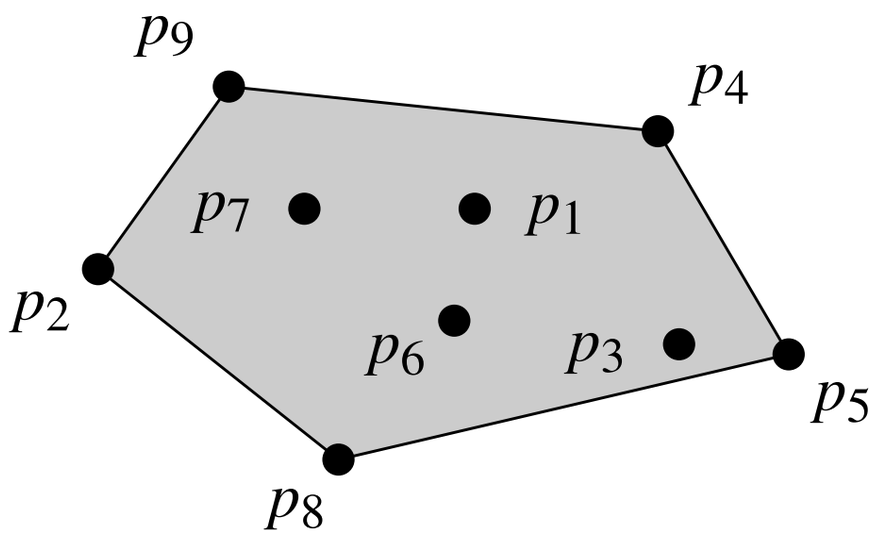
- El área dentro de la liga es el envolvente convexo de P.
- $\mathcal{CH}(P)$ de un conjunto finito P de puntos en el plano es el único polígono convexo cuyos vértices son puntos de P y que contiene todos los puntos de P.
- Una forma natural de representar un polígono es listando sus vértices en el sentido de las manecillas del reloj, empezando de un vértice arbitrario.

Envolvente convexo (convex hull)

- Dado un conjunto de puntos $P = \{p_1, p_2, \ldots, p_n\}$ en el plano, calcular una lista que contenga aquellos puntos de P que sean vértices de $\mathcal{CH}(P)$, en orden de las manecillas del reloj.

- entrada: conjunto de puntos $p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9$.

- salida: representación del envolvente convexo p_4, p_5, p_8, p_2, p_9.



Envolvente convexo (convex hull)

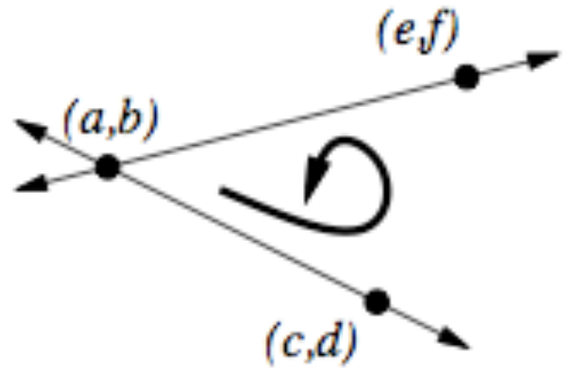
- Calcular el envolvente convexo de un punto es trivial:
- solamente regresamos ese punto.
- Calcular el envolvente convexo de dos puntos también es trivial.
- Para tres puntos, tenemos dos posibilidades:
  - los puntos están listados en un arreglo en sentido horario.
  - los puntos están listados en un arreglo en sentido contrario a las manecillas del reloj.
  - Supongamos nuestros tres puntos *(a,b)*, *(c,d)* y *(e,f)* dados en ese orden.
  - Supongamos también que el primer punto está más lejos a la izquierda, entonces $a < c$ y $a < e$.

tomado del curso de Jeff Erickson de la U. de Illinois

Envolvente convexo (convex hull)

- Los tres puntos están en sentido contrario a las manecillas del reloj si y solo si:
- la línea $\overleftrightarrow{(a,b)(c,d)}$ tiene una pendiente menor que la línea $\overleftrightarrow{(a,b)(e,f)}$:

$$\text{counterclockwise} \iff \frac{d-b}{c-a} < \frac{f-b}{e-a}$$



- que se puede escribir también como:

$$\boxed{\text{counterclockwise} \iff (f-b)(c-a) > (d-b)(e-a)}$$

- Si los puntos están en sentido horario se invierte la desigualdad.
- Si los puntos son colineales (supondremos que este caso no sucederá), las dos expresiones son iguales.

tomado del curso de Jeff Erickson de la U. de Illinois

Envolvente convexo (convex hull)

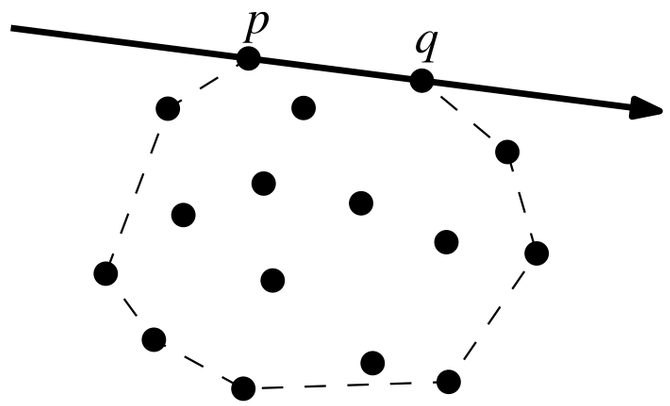
- Otra forma de pensar la prueba de sentido de los puntos es calcular el producto cruz de dos vectores: *(c,d)-(a,b)* y *(e,f)-(a,b)*, definido por un determinante de 2x2:

$$\text{counterclockwise} \iff \begin{vmatrix} c-a & d-b \\ e-a & f-b \end{vmatrix} > 0$$

- Mismo papel que comparaciones en algoritmos de ordenamiento.
- Convex hull de 3 puntos, análogo a ordenamiento de dos números.

Envolvente convexo (convex hull)

- El algoritmo para calcular $\mathcal{CH}(P)$ se basa en su estructura de aristas:
- Los puntos extremos p y q de cada arista son puntos de P.
- Si esta arista está dirigida de tal forma que $\mathcal{CH}(P)$ esté a la derecha, entonces todos los puntos de P deben estar a la derecha de esta linea.
- El argumento inverso es también cierto, si todos los puntos de $P\backslash\{p, q\}$ está a la derecha de la línea dirigida entre p y q, entonces $\overline{pq}$ es una arista de $\mathcal{CH}(P)$.



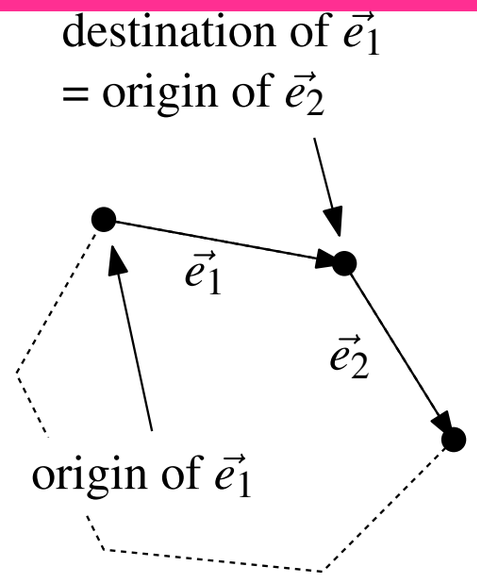
Algoritmo

Algorithm SLOWCONVEXHULL(P)
Input. A set P of points in the plane.
Output. A list $\mathcal{L}$ containing the vertices of $\mathcal{CH}(P)$ in clockwise order.
1. $E \leftarrow \emptyset$.
2. **for** all ordered pairs $(p,q) \in P \times P$ with p not equal to q
3. **do** *valid* $\leftarrow$ **true**
4. **for** all points $r \in P$ not equal to p or q
5. **do if** r lies to the left of the directed line from p to q
6. **then** *valid* $\leftarrow$ **false**.
7. **if** *valid* **then** Add the directed edge $\overrightarrow{pq}$ to E.
8. From the set E of edges construct a list $\mathcal{L}$ of vertices of $\mathcal{CH}(P)$, sorted in clockwise order.

Envolvente convexo (convex hull)

- ¿Cómo se puede verificar que los puntos estén a la derecha o a la izquierda de una línea dirigida?
- en los algoritmos de geometría computacional suponemos esta operación como una de tiempo de ejecución constante.
- ¿Cómo se puede construir una lista, a partir de las aristas, ordenada en el orden de las manecillas del reloj?
- Las aristas en *E* son dirigidas, por lo que tenemos un origen y un destino.
- Como las aristas están dirigidas de tal forma que los otros puntos estén a su derecha, el vértice destino viene después del origen cuando están ordenados en sentido de las manecillas del reloj.

Envolvente convexo (convex hull)



- Una implementación simple del procedimiento de la línea 8 toma $O(n^2)$ pero se puede mejorar a $O(n \lg n)$.
- El resto del algoritmo domina el tiempo total de ejecución.
- Verificamos $n^2 - n$ pares de puntos. Para cada par miramos $n - 2$ puntos para encontrar los que están del lado derecho. Esto hace en total $O(n^3)$.
- La operación de la línea 8 toma $O(n^2)$ por lo que la complejidad total del algoritmo es $O(n^3)$.

Algoritmo incremental: casos degenerados

- Primera etapa: Entender la geometría del problema, ignorar los casos degenerados.
- Segunda etapa: Ajustar el algoritmo a los casos degenerados. Tratar de generalizar y no resolver caso por caso.
- Tercera etapa: Implementación, especial atención a redondeos y puntos flotantes.