

Programa. Convex Hull

0.1 Metodología

Se puede definir la envolvente convexa de un conjunto de puntos X como el conjunto C que es convexo y es el más pequeño pero que contiene a X .

Para encontrar la envolvente convexa de un conjunto de puntos $\{(x_i, y_i)\}_{i=1}^n$ existen 2 métodos que son muy populares, estos son: La marcha de Jarvis y Scan Graham.

La idea del algoritmo de la marcha de Jarvis es seleccionar el punto p_{min} que tenga la coordena x menor y a partir de ahí determinar que otros puntos también formarán parte de la envolvente convexa. Para esto, si tenemos un punto p que ya es parte de la envolvente convexa, determinaremos si un nuevo punto q es parte de la envolvente si cualquier punto r está por arriba de la linea formada por los puntos p y q . Esto equivale a hacer el producto cruz entre los vectores pq y pr y después fijarse en el signo del resultado.

Para el caso de Graham Scan, la idea es bastante similar, sólo que se empieza por el punto que tenga la coordenada y mínima. Después se ordenan los puntos restantes de acuerdo al ángulo que forman respecto a la horizontal formada por un punto inicial. Finalmente, se iteran los puntos de menor a mayor ángulo y se decide si pertenece a la envolvente convexa mediante la misma operación que en el algoritmo de Jarvis, esto es, el producto cruz.

0.2 Implementación

Para ambos algoritmos se usa un vector de elementos de una clase llamada punto que contendrán las coordenadas de cada punto.

Se crea una clase Geom que contiene como métodos a ambos algoritmos, la marcha de jarvis con el nombre convexHull() y Graham Scan con el nombre convexHullGraham(). Además de contener funciones para leer una imagen pgm en blanco y negro a la que se le quiera aplicar el algoritmo o un archivo con una. nube de puntos. Finalmente se implementa una función para poder graficar la imagen y su envolvente convexa mediante cairo.

Algorithm 1: Ejemplo envolvente de imagen con Jarvis

```
// Se lee imagen pgm

g.read_pgm("estrella.pgm");

// Se obtiene envolvente convexa mediante jarvis
```

```
g.convexHullGraham();  
  
// Se grafica tanto la imagen como la envolvente convexa mediante cairo  
  
g.plot("Graham_Convex_Hull");
```

0.3 Resultados

Figure 1: Imagen original

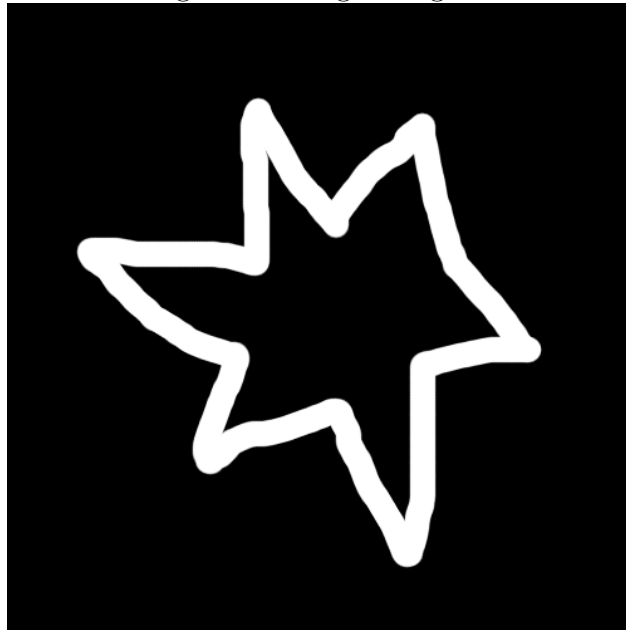


Figure 2: Envolverte convexa Jarvis para imagen

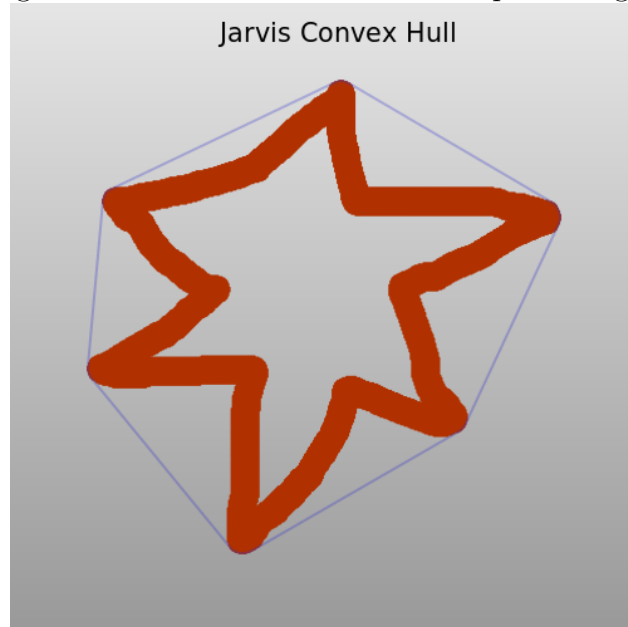


Figure 3: Envolverte convexa Graham scan para nube de puntos

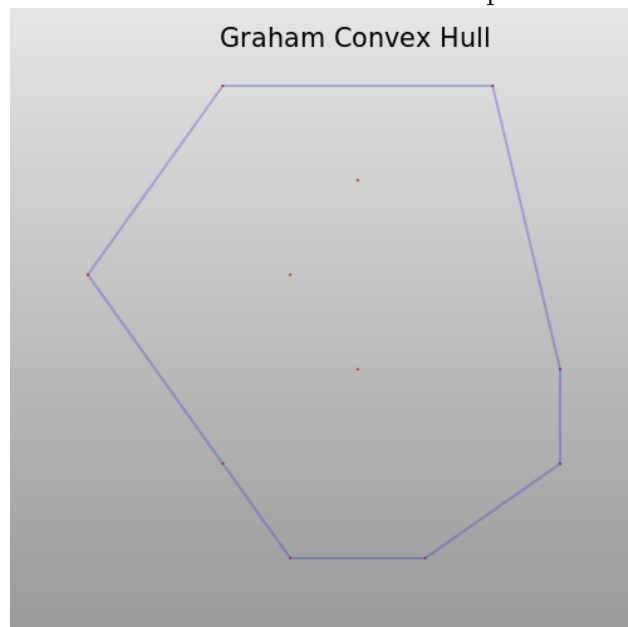
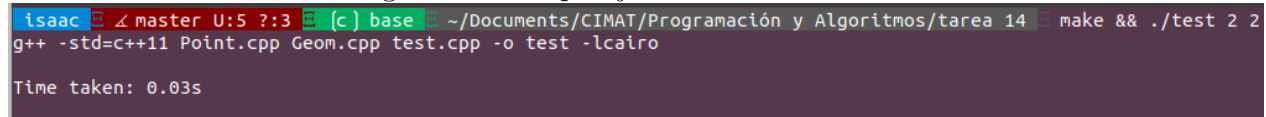


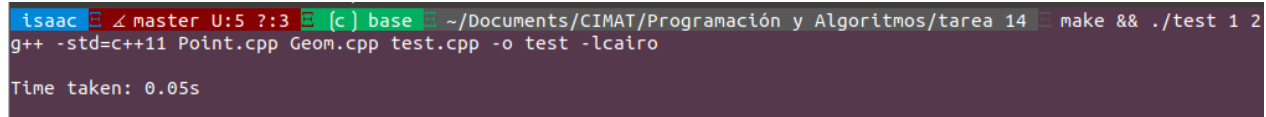
Figure 4: Tiempo ejecución Graham scan



```
isaac @ master U:5 ? :3 [c] base ~/Documents/CIMAT/Programación y Algoritmos/tarea 14 make && ./test 2 2
g++ -std=c++11 Point.cpp Geom.cpp test.cpp -o test -lcairo

Time taken: 0.03s
```

Figure 5: Tiempo ejecución marcha Jarvis



```
isaac @ master U:5 ? :3 [c] base ~/Documents/CIMAT/Programación y Algoritmos/tarea 14 make && ./test 1 2
g++ -std=c++11 Point.cpp Geom.cpp test.cpp -o test -lcairo

Time taken: 0.05s
```

0.4 Conclusión

Se pudo observar que ambos algoritmos dan los mismos resultados, sin embargo, Graham Scan tiene una mejor complejidad ya que es de orden $O(n \log n)$, donde la operación mas compleja es la ordenación de los puntos. Mientras que el algoritmo de jarvis tiene una complejidad de $O(nh)$ donde n es el número de puntos de la imagen y h es el número de elementos de la envolvente convexa, siendo su peor caso $O(n^2)$ cuando todos los puntos forman parte de la envolvente convexa. Esta diferencia en complejidades se pudo observar en los tiempos de ejecución ya que para encontrar la envolvente de la imagen, el algoritmo de Jarvis se tardó 0.05s mientras que Graham Scan se tardó 0.03s.