More Sorting-Based Geometrical Algorithms

Convex Hull Algorithms

Francesco Andreussi

Bauhaus-Universität Weimar

21 June 2019

Bauhaus-Universität Weimar

Fakultät Medien

Convex Hull

What is it?

Convex Hull

What is it?

Definition

A **Convex Hull**, or Convex Envelope or Convex Enclosure, of a set of points $\mathcal{X}$ is the minimum subset of points of $\mathcal{X}$ defining a convex polygon that contains all the other points of the set.

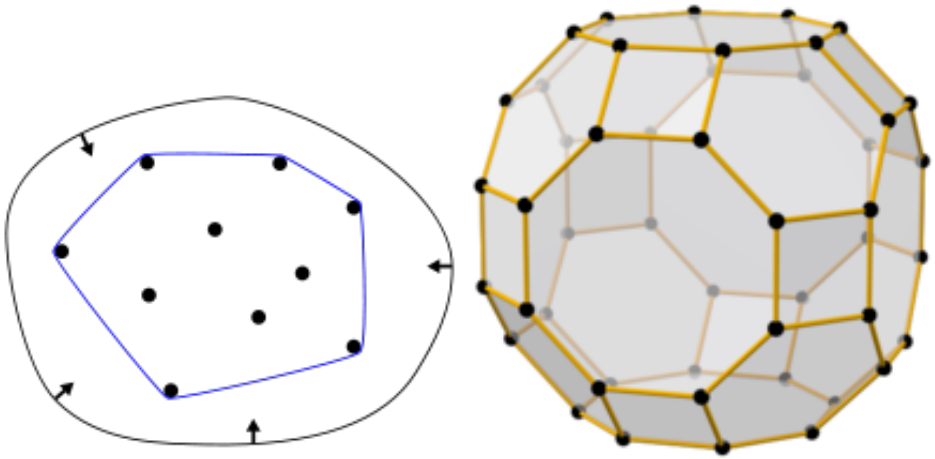

Fig. 1: 2D and 3D Convex Hulls

Three Possibile Paths

- **Incremental Methods:** in this category are included the Sorting-Based Algorithms. The points are oredered horizontally or vertically and every step of the algorithm a new point is added to the Convex Hull.
- **Divide & Conquer:** these methods subdivide recursively the set in two halves (left-right or upper-lower), computes the respective Convex Hull and uses a special method for combining them in one hull.
- **Prune & Search:** similar to the Divide & Conquer approach, but, instead of computing the Convex Hull of the sub sets, it isolates the *(sub)chains* (sequences of consecutive segments that are part of the Convex Hull) removing at every step a constant fraction of points.

Prerequisites

Polar Coordinates vs Cartesian Coordinates

A point in a plane can be represented in several different ways; two of these are **polar** and **cartesian** coordinates.

In the **cartesian** coordinates, the two components (x, y) represent the distance between the point and the origin in the two dimensions of the plane.

In the **polar** coordinates, the two components (d, θ) represent the distance between the point and the origin as a scalar (i.e. $d = \sqrt{x^2 + y^2}$) and the angle between the vector from the origin to the point and the x axis.

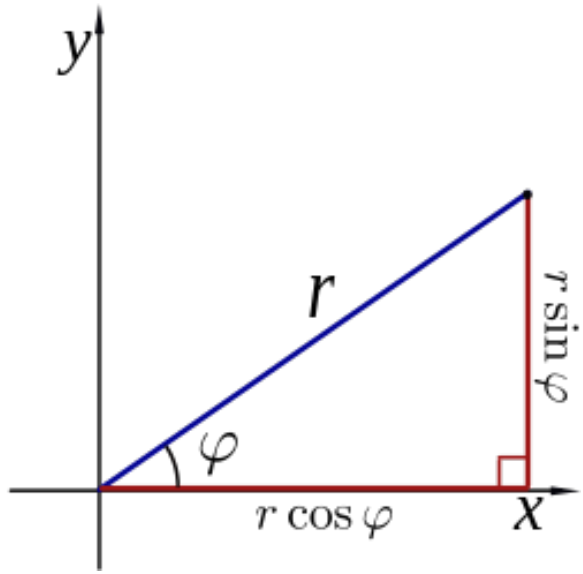

Fig. 2: Form Polar to Cartesian Coordsinates

Graham's Scan

Pseudocode

```
GRAHAM-SCAN(X):
  p_0 = LEFTMOST(LOWEST(X))
  Q = X - p_0
  //sorting in increasing order wrt the polar angle
  Q = SORT(Q)
  S = EMPTY-STACK()
  PUSH(p_0, S)
  PUSH(Q[0], S)
  PUSH(Q[1], S)
  for i = 2 to SIZE-OF(Q):
     while !ANGLE(SECOND(S), FIRST(S), Q[i]).turnsLeft():
       POP(S)
     PUSH(Q[i], S)
  return S
```

Graham's Scan

The Turn Left

A **turn left** happens when the angle defined by three points (where middle one is the main vertex) is **clockwise positive** w.r.t. the vector from the central vertex to the first point of the triple.

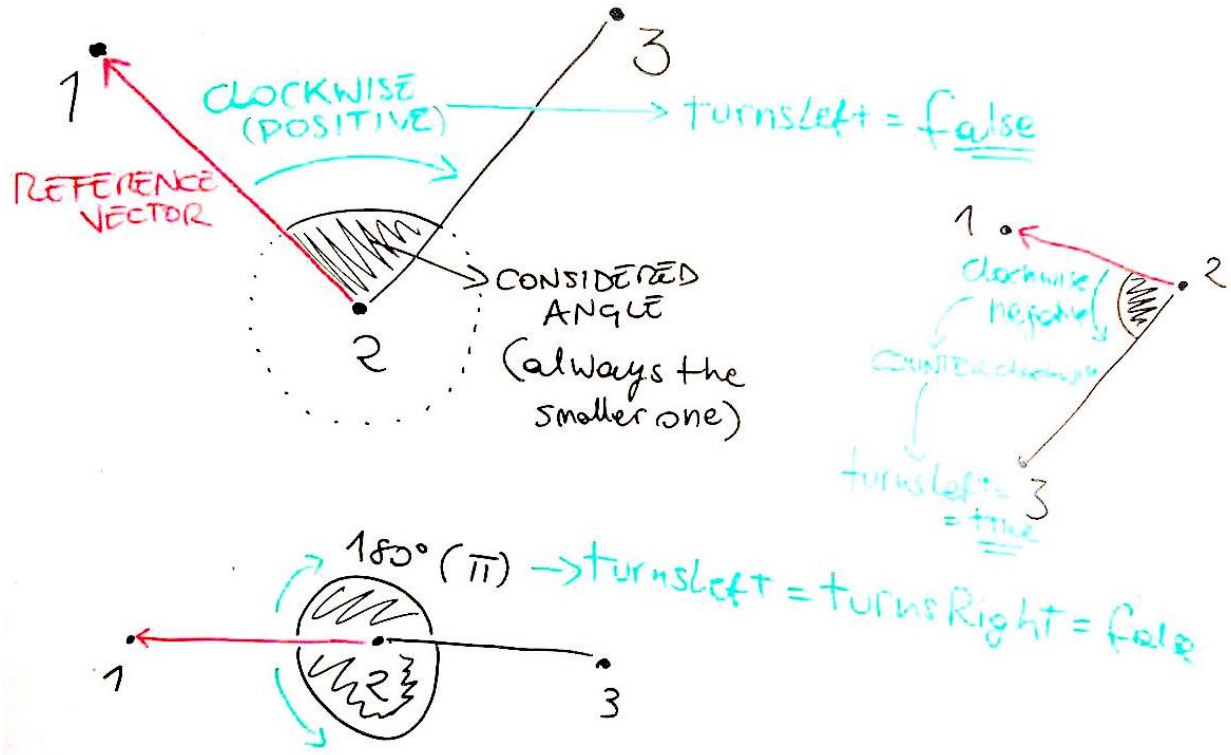

Fig. 3: Sketch of the Left Turn meaning

Graham's Scan

An Example

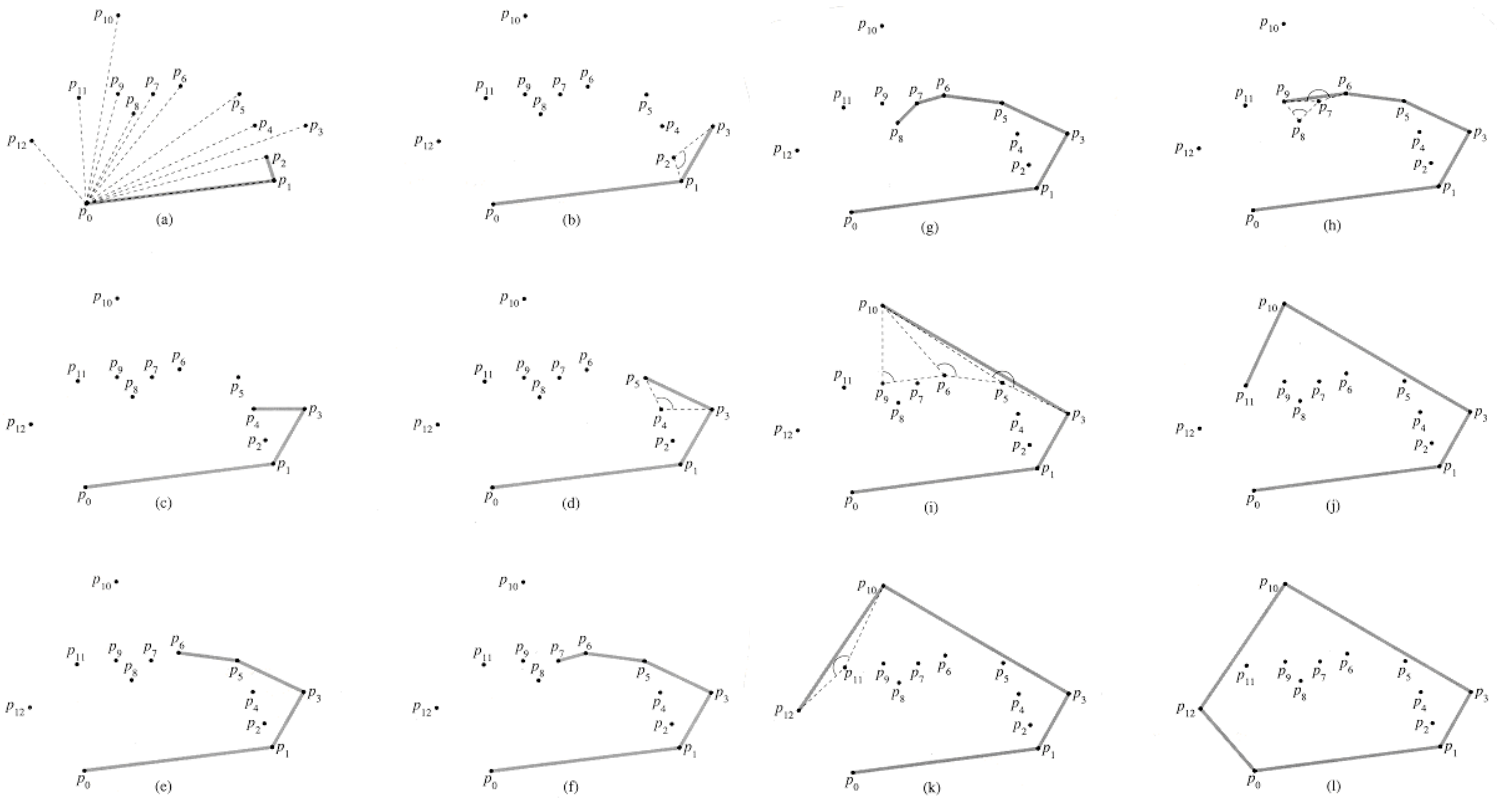

Fig. 4: Example of Execution of Graham's Scan

Image from Cormen, Leiserson, *Introduction to Algorithms*, 3rd Edition, MIT Press

Jarvis' Algorithm (Wrapping)

Pseudocode

```
JARVIS(X):
 p_0 = LEFTMOST(LOWEST(X))
 Q = X
 S = STACK(p_0)
 while min != p_0:
  for p in Q:
 // polar coords of p wrt the last vertex added to S ->
 // -> Rotation of the points around last vertex
     Q = POLAR(p,FIRST(S))
  // min wrt the angle (get it from the operation above)
  min = POPMIN(Q)
  PUSH(min,S)
 return S
```

Jarvis's Algorithm

The Points Rotation

A **rotation** consists in the **recomputation** of the polar coordinates w.r.t. the last point added to the Convex Hull and the following **increment**, to the computed polar angles, of the former polar angle value of the current new center.

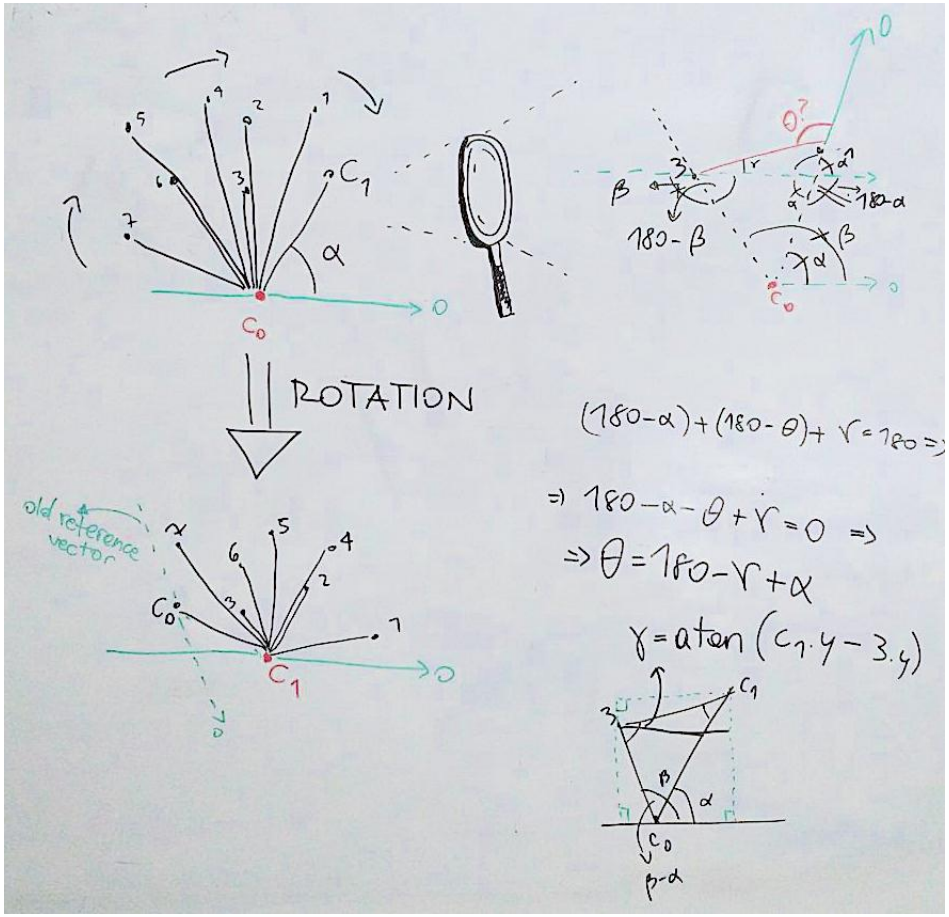

Fig. 5: Sketch of rotation mechanism and new angle calculation

Jarvis's Algorithm

An Example

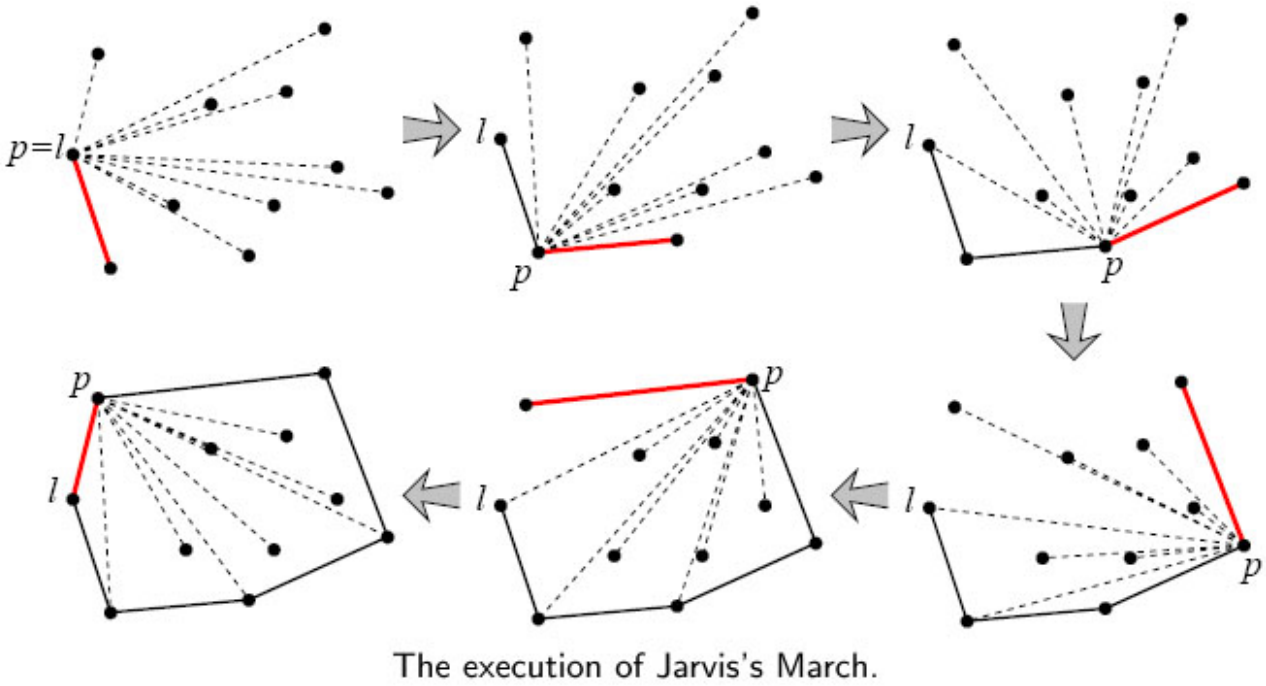

Fig. 6: Example of Execution of Jarvis' Algorithm

Image from Mortensen, *Algorithms for Computing Convex Hulls Using Linear Programming*

Thanks for the Attention!