Advanced Algorithm

Geometric Algorithms

Topics to be covered

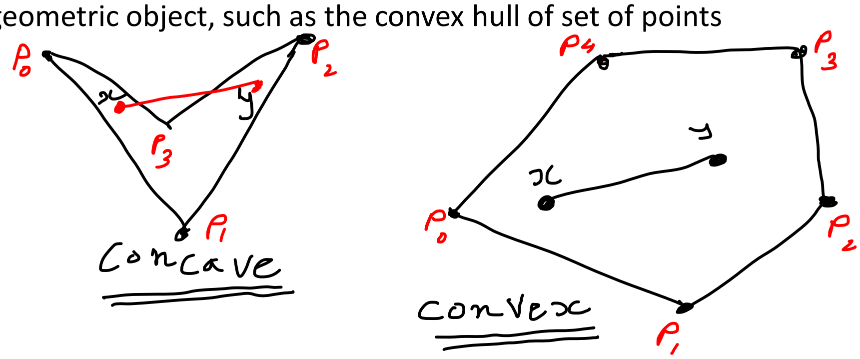
- What is Computation Geometry ?
- Applications of Geometric Algorithm
- Properties of Line Segment
- Closest Pair of points (Brute-Force Approach)
- Intersection of Two Line Segments
- Finding Convex Hull using Graham Scan

What is Geometric Algorithm ?

- Computational geometry is the branch of computer science that studies algorithms for solving geometric problems.
- The input to a computational-geometry problem is typically a description of a set of geometric objects, such as a set of points, a set of line segments, or the vertices of a polygon in **counterclockwise** order.
- Each input object is represented as a set of points $\{p_i\}$, where each $p_i = (x_i, y_i)$ and $x_i, y_i \in \mathbf{R}$.

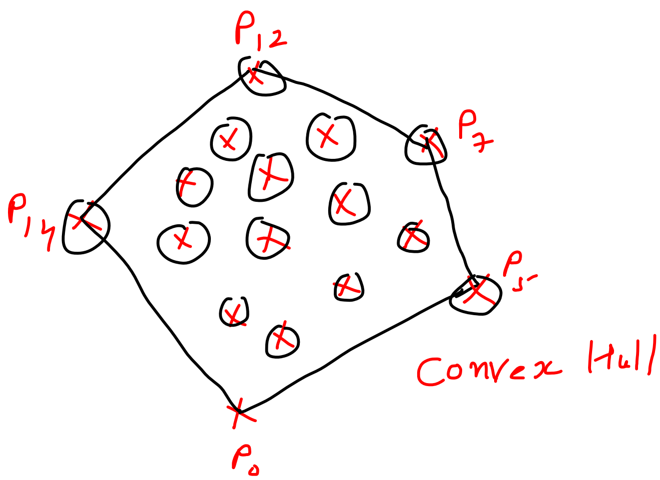
What is Geometric Algorithm ?

- For example, an *n*-vertex polygon *P* is represented by a sequence $p_0, p_1, p_2, \ldots, p_{n-1}$ of its vertices in order of their appearance on the boundary of *P*.
- The output is often a response to a query about the objects, such as whether any of the lines intersect, or perhaps a new geometric object, such as the convex hull of set of points



What is Geometric Algorithm ?

- Computational geometry can also be performed in three dimensions, and even in higher- dimensional spaces, but such problems and their solutions can be very difficult to visualize.



Applications of Geometric Algorithm

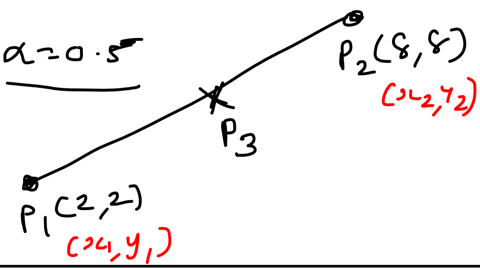
- Data Mining
- Image Processing
- Computer Graphics, Games, Virtual Reality
- Fractal Geometry
- Animation
- VLSI Design
- Computer Aided Design (Civil Drawings)
- Architecture (3D Building Drawings)
- Mechanical Engineering (2D/3D Machine Design)
- Statistics
- Global Positioning System
- Robotics (Finding Paths etc.)
- Airflow around an aircraft wing
- Air traffic Control

Properties of Line Segments

- A ***convex combination*** of two distinct points $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ is any point $p_3 = (x_3, y_3)$ such that for some a in the range $0 <= a <= 1$, we have $x_3 = ax_1 + (1 - a)x_2$ and $y_3 = ay_1 + (1 - a)y_2$. We also write that $p_3 = a\, p_1 + (1 - a)p_2$.

- Intuitively, p_3 is any point that is on the line passing through p_1 and p_2 and is on or between p_1 and p_2 on the line.

$\underline{\alpha = 0.5}$



$$x_3 = \alpha \cdot x_1 + (1-\alpha)x_2$$
$$= 0.5 \times 2 + 0.5 \times 8$$
$$= 0.5(10) = 5$$
$$y_3 = \alpha y_1 + (1-\alpha) y_2$$
$$= 0.5(10) = 5$$

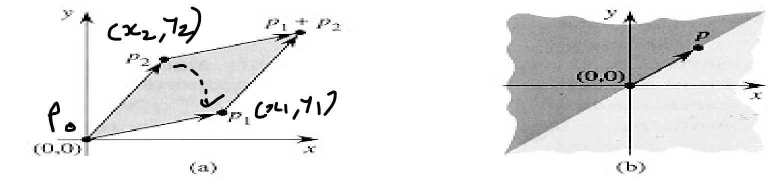
Properties of Line Segments

- Given two distinct points p_1 and $p2$, the ***line segment*** $\overline{p1p2}$ is the set of convex combinations of p_1 and p_2. We call p_1 and p_2 the ***endpoints*** of segment $\overline{p1p2}$

- Sometimes the ordering of p_1 and p_2 matters, and we speak of the ***directed segment*** $\overrightarrow{p1p2}$

- If p_1 is the ***origin*** (0, 0), then we can treat the directed segment $\overrightarrow{p1p2}$ as the ***vector*** p_2.

Computing Cross Product



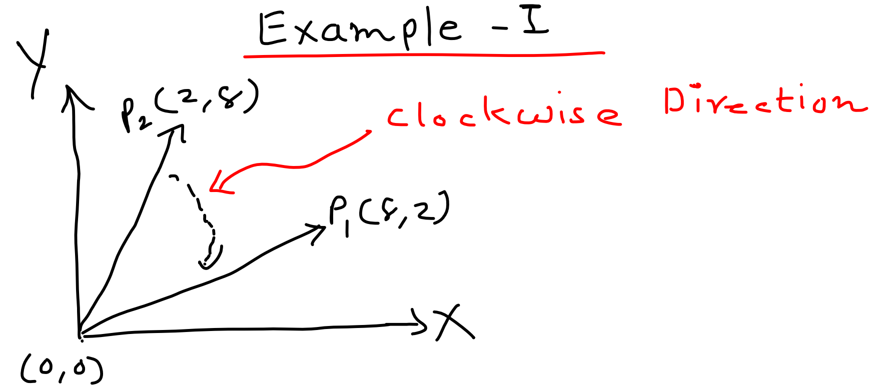
$$p1 \times p2 = \begin{vmatrix} x1 & x2 \\ y1 & y2 \end{vmatrix}$$

p1 x p2 =x1y2-x2y1

If p_1 X p_2 is positive, then p_1 is clockwise from p_2 with respect to the origin (0, 0); if this cross product is negative, then p_1 is counterclockwise from p_2.

A boundary condition arises if the cross product is zero; in this case, the vectors are ***collinear***, pointing in either the same or opposite directions.

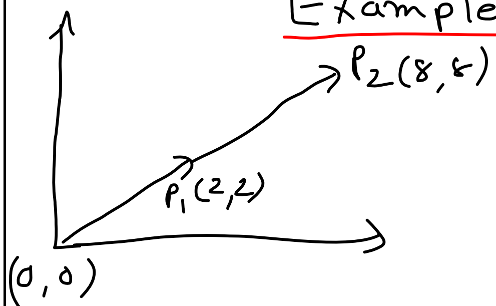
Example - I



$$P_1 \times P_2 = \begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} = \begin{vmatrix} 8 & 2 \\ 2 & 8 \end{vmatrix} = 64 - 4 = 60 > 0$$

$\overrightarrow{P_0 P_1}$ is clockwise from $\overrightarrow{P_0 P_2}$

Example - II



$$p_1 \times p_2 = \begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} = \begin{vmatrix} 2 & 8 \\ 2 & 8 \end{vmatrix} = 2\times 8 - 2\times 8 = 0$$

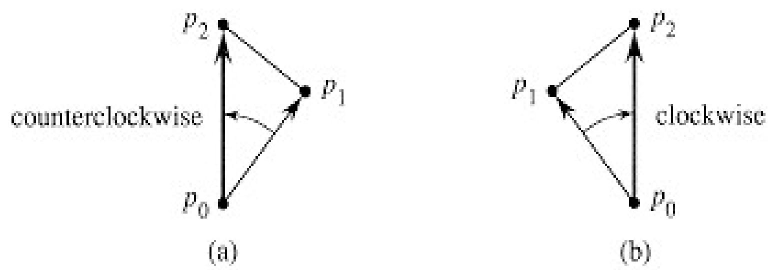
$\overrightarrow{p_0p_1}$ and $\overrightarrow{p_0p_2}$ are Colinear.

Computing Cross Product

- To determine whether directed segment $\overrightarrow{p0p1}$ is clockwise from directed segment $\overrightarrow{p0p2}$ with respect to their common point p0, we translate to use p0 as the origin.
- That is, we let $p_1 - p_0$ denote the vector $p'_1 = (x'_1, y'_1)$, where $x'_1 = x_1 - x_0$ and $y'_1 = y_1 - y_0$, and we define $p_2 - p_0$ similarly.
- We then compute the cross product $(p_1 - p_0) \times (p_2 - p_0) = (x_1 - x_0)(y_2 - y_0) - (x_2 - x_0)(y_1 - y_0)$.

If cross product is positive then, $\overrightarrow{p0p1}$ is clockwise from $\overrightarrow{p0p2}$

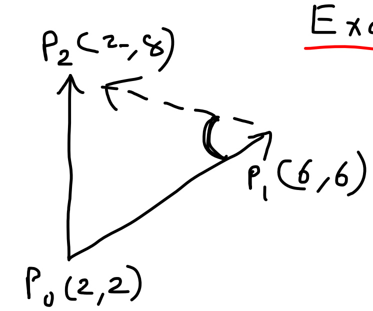
Determining whether consecutive segments turn left or right



We then compute the cross product

$$(p_2 - p_0) \times (p_1 - p_0) = (x_2 - x_0)(y_1 - y_0) - (x_1 - x_0)(y_2 - y_0).$$

Example - III



$P_1(x_1, y_1) = (6,6)$

$P_2(x_2, y_2) = (2,8)$

$P_0(x_0, y_0) = (2,2)$

$$\begin{aligned}(P_2 - P_0) \times (P_1 - P_0) &= (x_2 - x_0)(y_1 - y_0) - (x_1 - x_0)(y_2 - y_0) \\ &= 0 \times 4 - 4 \times 6 \\ &= -24 < 0\end{aligned}$$

$\overrightarrow{P_0P_2}$ is in counterclockwise direction from $\overrightarrow{P_0P_1}$

We are taking a left turn at $\angle P_0P_1P_2$

Closest Pair of Points (Brute-Force Approach)

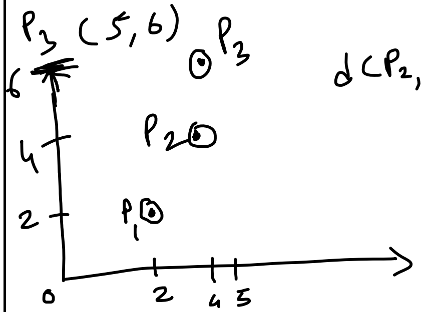
Euclidean distance $d(P_i, P_j) = \sqrt{[(x_i - x_j)^2 + (y_i - y_j)^2]}$

Find the minimal distance between a pairs in a set of points

$P_1 (2,2)$

$P_2 (4,4)$

$P_3 (5,6)$



$d(P_1, P_2) = \sqrt{(2-4)^2 + (2-4)^2}$

$= \sqrt{4+4} = \sqrt{8} = \boxed{2.78}$

$d(P_2, P_3) = \sqrt{(4-5)^2 + (4-6)^2}$

$= \sqrt{1+4} = \sqrt{5} = 2.1$

$d(P_1, P_3) = \sqrt{(2-5)^2 + (2-6)^2}$

$= \sqrt{9+16} = \sqrt{25} = \boxed{5}$

Closest Pair of Points (Brute-Force Approach)

Algorithm BruteForceClosestPoints(P)

// P is list of points

$dmin \leftarrow \infty$

for $i \leftarrow 1$ **to** n-1 **do**

for $j \leftarrow i$+1 **to** n **do**

$d \leftarrow \text{sqrt}((x_i - x_j)^2 + (y_i - y_j)^2)$

if $d < dmin$ **then**

$dmin \leftarrow d$; $index1 \leftarrow i$; $index2 \leftarrow j$

return *index1*, *index2*

i	j	dmin	i1	i2
1	2	2.78	1	2
1	3	2.78	1	2
2	3	2.1	2	3

Closest Pair of Points
(Brute-Force Approach)

$$C(n) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \cdot c = c \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \cdot 1$$

$i=1 \Rightarrow \sum_{j=2}^{n} \cdot 1 = n-1$

$i=2 \Rightarrow \sum_{j=3}^{n} 1 = n-2$

$i=3 \Rightarrow \sum_{j=4}^{n} \cdot 1 = n-3$

$$= c \sum_{i=1}^{n-1} (n-i)$$

$$= c \left[\sum_{i=1}^{n-1} \cdot n - \sum_{i=1}^{n-1} i \right]$$

$$= c \left[n(n-1) - \frac{n(n-1)}{2} \right]$$

$$= c \cdot \frac{n(n-1)}{2} = \Theta(n^2)$$