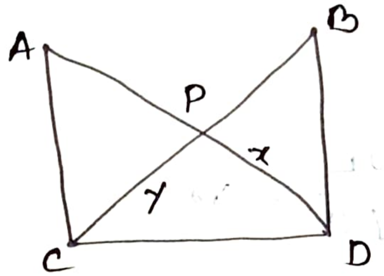
L-8

Given,

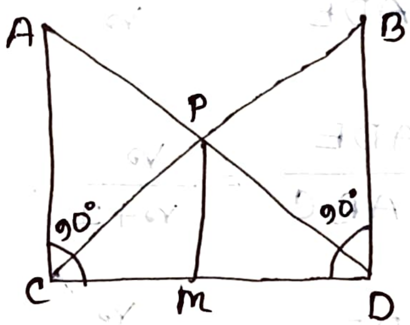
$AC \perp CD$ and $BD \perp CD$

$AD = x$ & $BC = y$



The distance at point p to CD is c find out $CD = ?$

Answer:-



ΔACD & ΔDMP

$$\frac{PM}{AC} = \frac{MD}{CD} = \frac{DP}{AD} \quad \text{--- (i)}$$

ΔBCD & PCm

$\angle C = \angle C$

$\angle D = \angle m$

$\angle P = \angle B$

$$\frac{PM}{BD} = \frac{CM}{CD} = \frac{PC}{BC} \quad \text{--- (ii)}$$

$$AC = \sqrt{AD^2 - CD^2}$$

$$= \sqrt{x^2 - CD^2}$$

$$BD = \sqrt{BC^2 - CD^2} = \sqrt{y^2 - CD^2}$$

① + ⑪

$$\frac{PM}{AC} + \frac{PM}{BD} = \frac{MD}{CD} + \frac{CM}{CD}$$

$$\Rightarrow \frac{PM}{AC} + \frac{PM}{BD} = \frac{MD + CM}{CD}$$

$$\Rightarrow PM\left(\frac{1}{AC} + \frac{1}{BD}\right) = \frac{CD}{CD} = 1.$$

$$\Rightarrow PM\left(\frac{BD + AC}{AC \cdot BD}\right) = 1.$$

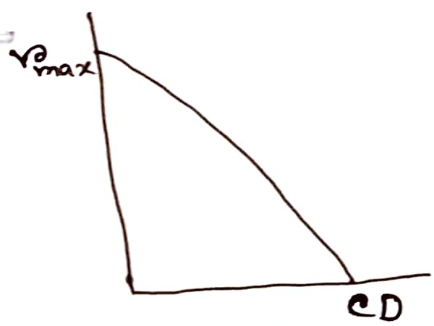
$$\Rightarrow \frac{AC \cdot BD}{BD + AC} = PM$$

$$\Rightarrow \frac{\sqrt{x^2 - CD^2}\,\sqrt{y^2 - CD^2}}{\sqrt{x^2 - CD^2} + \sqrt{y^2 - CD^2}} = PM = c$$

↓
ratio function :

Using numerical method :-

$CD = [0, \min(x, y)]$



strictly decreasing monotonic function

if $x > y$

$CD_{max} = y$

if $x < y$

$CD_{max} = x$.

Here we use binary search because this is a strictly decreasing monotonic function.

get-ratio (x, y, CD)

1. compute $r = \dfrac{sqrt(x*x - CD*CD) * sqrt(y*y - CD*CD)}{sqrt(x*x - CD*CD) + sqrt(y*y - CD*CD)}$

2. return r.

get_CD (x, y, c)

① Set low $= 0.0$

② Set high $= \min(x, y)$

③ While (low <= high) or (high - low $\geq 10^{-6}$)

(i) compute mid = (low + high)/2.0

(ii) set CD = mid

(iii) comput r = get-ratio(x, y, CD)

(iv) if (r > c) low = mid.

else high = mid.

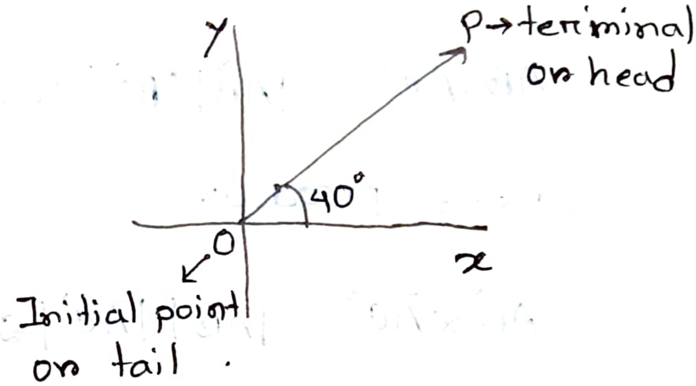
L-9

$OP = (r, \theta)$ [Polar]

magnatute → OP

$|\vec{OP}|$

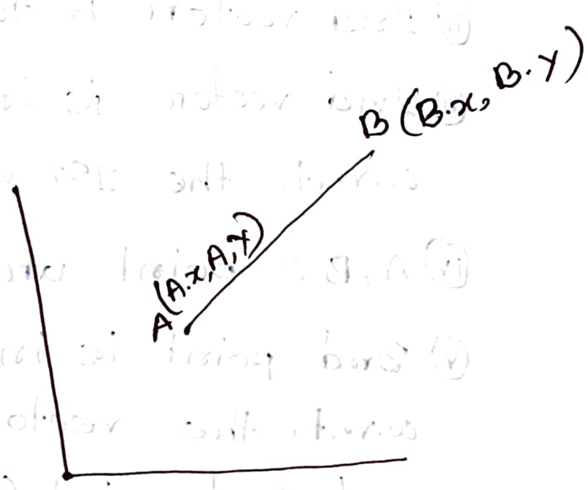
$||\vec{OP}||$



OP_x / $OP.x = r\cos\theta$ & $OP.y = r\sin\theta$.

$$\vec{OP} = (OP.x, OP.y) = (r\cos\theta, r\sin\theta)$$

$$\vec{AB} = (B.x - A.x, B.y - A.y)$$



$$\vec{AB} = \begin{bmatrix} AB.x \\ AB.y \end{bmatrix}$$

$$= \begin{bmatrix} AB.x & AB.y \end{bmatrix}$$

Two vector are parallel, when their angles are same

- ☐ Antiparallel vector
- ☐ Equal vector
- ☐ Zero Vector.

Dot Product

$\vec{AB} \cdot \vec{AC} = |\vec{AB}||\vec{AC}| \cos\theta$.

cross product

$\vec{AB} \times \vec{AC} = |\vec{AB}||\vec{AC}| \sin\theta \cdot \hat{n}$ → unit vector.

$|\vec{AB} \times \vec{AB}| > 0$ হবে যখন

(i) cross product is in counter clockwise direction.

(ii) 2nd vector is to the left of the 1st vector

(iii) 2nd vector is in counter clockwise direction w.r.t. the 1st vector.

(iv) A, B, C point are in counterclockwise direction

(v) 3rd point is in counter clockwise orientation w.r.t. the vector produced by 1st poin (A) and 2nd point (B).

Algorithm for counter-clock-wise (ccw) function

CCW (A, B, C)

(i) $\vec{AB} = (B.x - A.x \,,\; B.y - A.y)$

(ii) $\vec{AC} = (C.x - A.x \,,\; C.y - A.y)$

(iii) $|\vec{AB} \times \vec{AC}|$ = get_cross_product $(\vec{AB}, \vec{AC})$

(iv) If $|\vec{AB} \times \vec{AC}| > 0$

printf ("c is in the left of AB line)

(v) If $|\vec{AB} \times \vec{AC}| < 0$

printf (" c is in the right of AB line)

(vi) else printf (" A, B,C are co-linear points) .

get_cross_product $(\vec{AB}, \vec{AC})$

return (AB.x * AC.y - AB.y * AC.x)

$$\begin{vmatrix} AB.x & AC.x \\ AB.y & AC.y \end{vmatrix}$$

Lecture-10

Given, a line AB & A point P. Write a algorithm ~~whethe~~ whether the point p is on the line or NOT

Algorithm

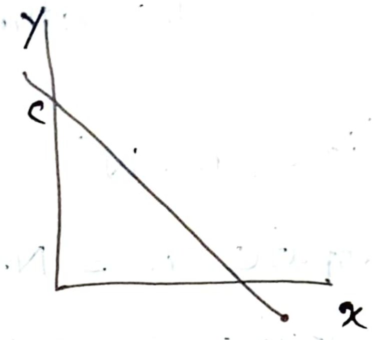
Point_on_line (A, B, P)

① If (ccw (A,B,P)==0 && min (Ax, Bx) ≤ Px ≤ max (Ax, Bx)
&& min (Ay, By) ≤ Py ≤ max (A.y, B.y))

return true;

② else return false.

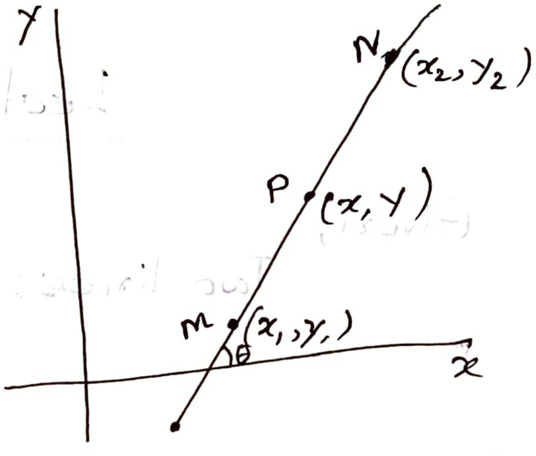
Line equation (slope intersect form)



$y = mx + c$

m ↓ slope, c ↳ y intersect.

*Calculate the standard form of line.



$m = \tan\theta$.

Given, $L_1 = MN$

$L_2 = MP$

$m_{L_1} = m_{L_2}$

$$\Rightarrow \frac{y_2 - y_1}{x_2 - x_1} = \frac{y - y_1}{x - x_1}$$

$$\Rightarrow \frac{x - x_1}{x_2 - x_1} = \frac{y - y_1}{y_2 - y_1}$$

$$\Rightarrow x(y_2 - y_1) - x_1(y_2 - y_1) = y(x_2 - x_1) - y_1(x_2 - x_1)$$

$$\Rightarrow x(y_2 - y_1) - x_1y_2 + x_1y_1 = -y(x_1 - x_2) - x_2y_1 + x_1y_1$$

$$\Rightarrow x\underbrace{(y_2 - y_1)}_{A} + y\underbrace{(x_1 - x_2)}_{B} = \underbrace{x_1y_2 - x_2y_1}_{C}$$

$\therefore Ax + By = C$ (Standard form of line) get_line Algorithm

get_line(M, N)

(i) Compute $A = N.y - M.y$

(ii) Compute $B = M.x - N.x$

(iii) Compute $C = ((M.x * N.y) - (N.x * M.y))$

(iv) return A, B, C.

Lecture-11

Given,

Two lines: $L_1 \; (P_1, q_1)$

$L_2 \; (P_2, q_2)$

[Q] Find out there intersect point

Answer:

$L_1 = A_1x + B_1y = C_1$

$L_2 : A_2x + B_2y = C_2$

If $\frac{A_1}{A_2} = \frac{B_1}{B_2}$ then $L_1 \parallel L_2$

or, $A_1B_2 = A_2B_1$

or, $A_1B_2 - A_2B_1 = 0$

$$\Delta = \begin{vmatrix} A_1 & B_1 \\ A_2 & B_2 \end{vmatrix} = A_1 B_2 - A_2 B_1$$

constant Matrix, $M = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix}$

Chremar's law,

$$x = \frac{\begin{vmatrix} C_1 & B_1 \\ C_2 & B_2 \end{vmatrix}}{\Delta} = \frac{B_2 C_1 - B_1 C_2}{A_1 B_2 - A_2 B_1}$$

$$y = \frac{\begin{vmatrix} A_1 & C_1 \\ A_2 & C_2 \end{vmatrix}}{\Delta} = \frac{A_1 C_2 - A_2 C_1}{A_1 B_2 - A_2 B_1}$$

Algorithm

get_intersection_point (P_1, q_1, P_2, q_2)

(i) ~~get line (P_1, q_1)~~

(i) A_1, B_1, C_1 = get_line(P_1, q_1)

(ii) $A_2\ B_2\ C_2$ = get_line(P_2, q_2)

(iii) Compute $\Delta = A_1 B_2 - A_2 B_1$

(iv) If $\Delta == 0$, return $L_1 \parallel L_2$.

(v) else, $x = \dfrac{B_2 C_1 - B_1 C_2}{\Delta}$, $y = \dfrac{A_1 C_2 - A_2 C_1}{\Delta}$

(vi) return x, y.

Algorithm for calculating distance between two points

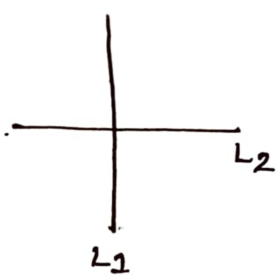
get_dist (p, q)

① return $sqrt((p.x - q.x)*(p.x - q.x) + (p.y - q.y)*(p.y - q.y))$

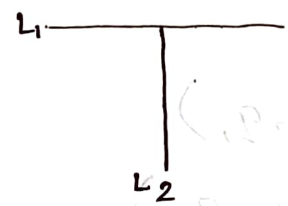
Given two line segment L_1, L_2. Write a algorithm whether the line segments intersect each other or not.

Let us consider some case.

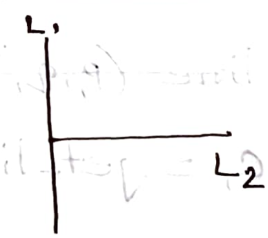
Case: 1



Case: 2

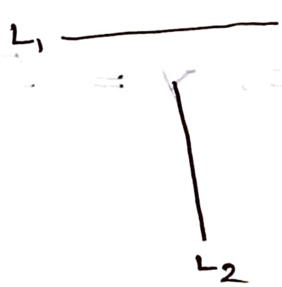


Case: 3

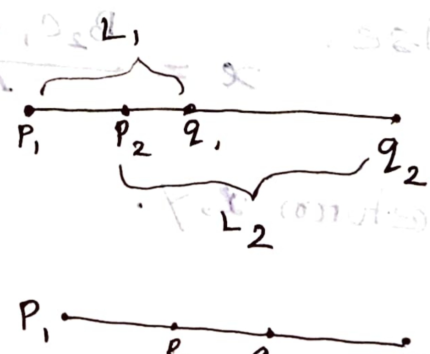


Case - 4

Case: 5



Case : 6



Algorithm

do_Is_Intersect (p_1, q_1, p_2, q_2)

(i) $O_1 = ccw(p_1, q_1, p_2)$

(ii) $O_2 = ccw(p_1, q_1, p_2)$

(iii) $O_3 = ccw(p_2, q_2, p_1)$

(iv) $O_4 = ccw(p_2, q_2, q_1)$

(v) If $O_1 != O_2$ & $O_3 != O_4$

return true

(vi) If $O_1 = 0$ & point_on_line$(p_1, q_1, p_2) ==$ true

return true.

(vii) If $O_2 = 0$ & point_on_line$(p_1, q_1, q_2) ==$ true

return true

(viii) If $O_3 = 0$ & point_on_line$(p_2, q_2, p_1) ==$ true

return true.

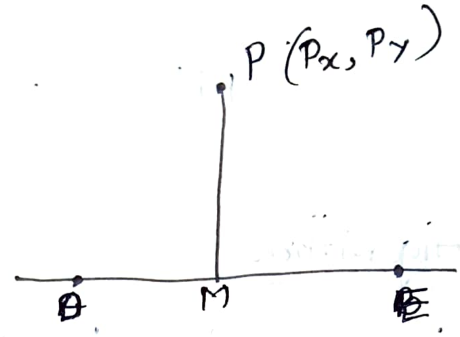
(ix) If $O_4 = 0$ & point_on_line$(p_2, q_2, q_1) ==$ tru

return true

(x) return false.

Lecture - 12

Given a point P &
a line 'L' by points
~~A&B~~ D & E .
Finds the distance (P, L)

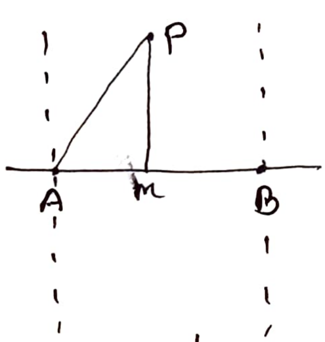


$Ax + By = C$

$$PM = \left| \frac{A.P_x + B.P_y - C}{\sqrt{A^2 + B^2}} \right|$$

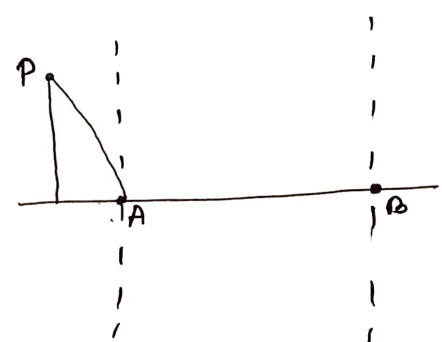
We can solve this problem by using vector algebra .

case 1



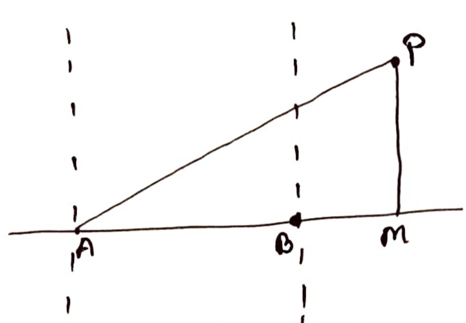
in Boundary

case - 2



Lelf of point A .

case : 3



right of point B

$$AB \times PM = \|\vec{AB} \times \vec{AP}\|$$

$$PM = \frac{\|\vec{AB} \times \vec{AP}\|}{AB}$$

Algorithm

distance_btw_line&point (P, A, B)

(i) AB = get_dist (A, B)

(ii) $\vec{AB} = B - A = (B.x - A.x, B.y - A.y)$

(iii) $\vec{AP} = P - A = (P.x - A.x, B.y - A.y)$

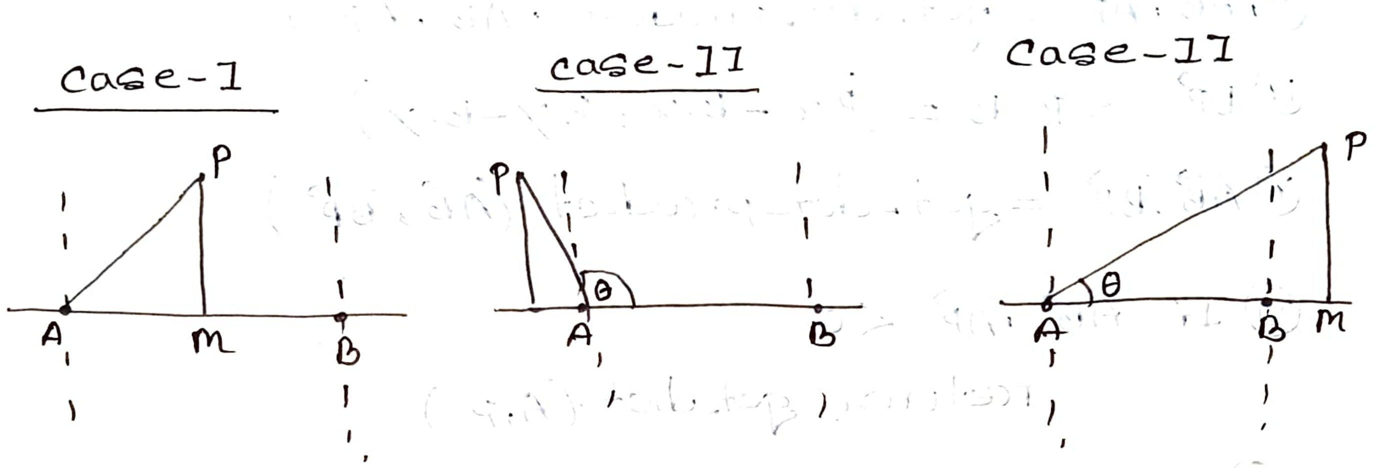
(iv) $\|\vec{AB} \times \vec{AP}\|$ = |get_cross_product $(\vec{AB}, \vec{AP})$|

(v) $PM = \frac{\|\vec{AB} \times \vec{AP}\|}{AB}$

(vi) return PM.

4. Given a point p and a line segment Ls. specified by point A & B. Find distance between P & Ls.

Let us consider,



In case II : θ is always θ>90°

$\overrightarrow{AB} \cdot \overrightarrow{AP} = AB \cdot AP \cdot \cos\theta < 0$ if $\theta > 90°$

$\overrightarrow{AB} \cdot \overrightarrow{BP} = AB \cdot BP \cos\theta > 0$ if $\theta < 90°$

Algorithm

Dist_btw_point&LS (P, A, B)

(I) $\vec{AB} = B - A = (B.x - A.x,\ B.y - A.y)$

(II) $\vec{AP} = P - A = (P.x - A.x,\ P.y - A.y)$

(III) $\vec{AB} \cdot \vec{AP}$ = get_dot_product $(\vec{AB}, \vec{AP})$

(IV) $\vec{BP} = P - B = (P.x - B.x,\ P.y - B.y)$

(V) $\vec{AB} \cdot \vec{BP}$ = get_dot_product $(\vec{AB}, \vec{BP})$

(VI) If $\vec{AB} \cdot \vec{AP} < 0$

return get_dist (A, P)

(VII) else if $\vec{AB} \cdot \vec{BP} > 0$

return get_dist (B, P)

(VIII) else

return distance_btw_line&point (P, A, B)

get_do_product $(\vec{A}, \vec{B})$

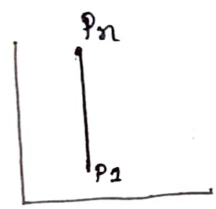
(I) return $((A.x * B.x) + (A.y * B.y))$.

Lecture - 13

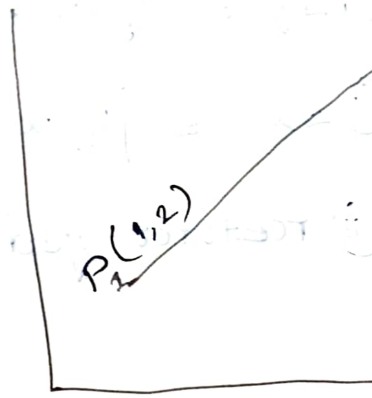
Find latice points in a line .

If

$x_1 = x_n$ $\longrightarrow$



latice points $= |(y_n - y_1)| + 1$



If,

$y_1 = y_n$

latice points $= |x_n - x_1| + 1$ $\rightarrow$

for slanted line,

$$M_{P_1P_n} = \frac{y - y_1}{x_n - x_1} = \frac{5-2}{7-1} = \frac{3}{6} = \frac{1}{2} = \frac{\Delta y}{\Delta x'}$$

$$\Delta x' \times n = \Delta x$$

$$\Delta y' \times n = \Delta y$$

$$n = \gcd(|\Delta y|, |\Delta x|)$$

Algorithm

get_n_latice_points (P_1, P_n)

(i) $\Delta y = |P_n.y - P_1.y|$

(ii) $\Delta x = |P_n.x - P_1.x|$

(iii) return gcd $(\Delta y, \Delta x)$ → If either of ~~the~~ one end point is exclusive

or,

return gcd $(\Delta y, \Delta x)+1$ → if both P_1 & P_n include

or

return gcd $(\Delta y, \Delta x)-1$ → if ,, P_1 & P_n exclusive

$\boxed{Q}$ $A(0,0)$, $B(2,0)$ & $P(4,0)$

$\boxed{Q}$ $A(1,0)$, $B(2,0)$ & $P(1,1)$

Calculate the distance ~~b/t~~ between the line segment AB & the point P.

Algorithm

get_n_latice_points (P_1, P_n)

(i) $\Delta y = |P_n \cdot y - P_1 \cdot y|$

(ii) $\Delta x = |P_n \cdot x - P_1 \cdot x|$

(iii) return gcd $(\Delta y, \Delta x)$ → If either of the one end point is exclusive

or,

return gcd $(\Delta y, \Delta x) + 1$ → if both P_1 & P_n include

or return gcd $(\Delta y, \Delta x) - 1$ → if ,, P_1 & P_n exclusive

Q A (0,0), B(2,0) & P(4,0)

Q A (1,0), B(2,0) & P(1,1)

Calculate the distance ~~bt~~ between the line segment AB & the point P.