Step1:- Get the 2 end pts. of a line (x_1, y_1) & (x_2, y_2)

Step2:- Identify Region code for both end pts.

1) If Both end points Region code is 0000 then line is completely inside the window. (LA)

2) else if both end pt. have 1 in same bit position in region code the line is outside window.

or

Perform AND operation for both region codes if the result is not 0000 then line is also completely outside, so rejected.

3) else move to step 3.

if line crosses $X_{w}min/x_{w}max$ then find

$$y = y_1 + m(x - x_1)$$

where $x = x_{w}min/x_{w}max$

else (line crosses $y_{w}min/y_{w}max$)

$$x = x_1 + m\,\frac{(y - y_1)}{m}$$

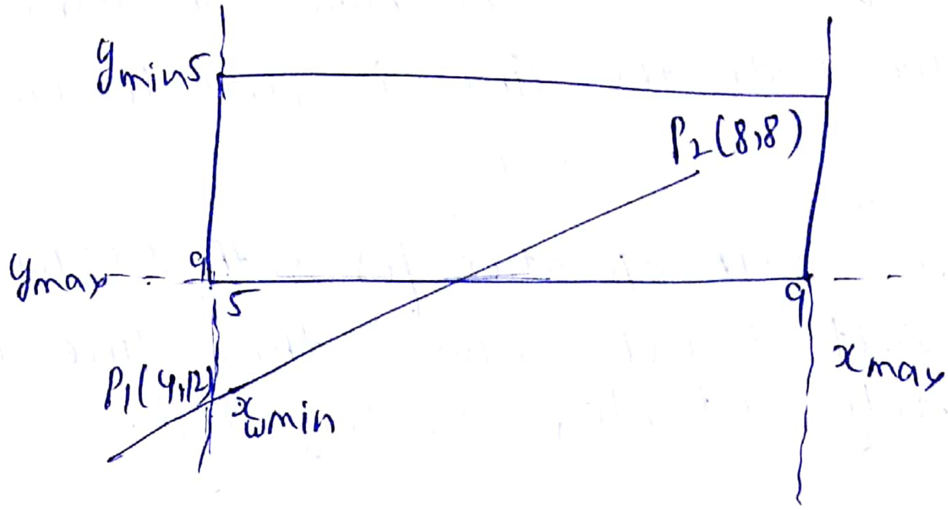
Step4:- If cal. (x, y) doesn't satisfy foll. condition then repeat step 3

$$x_{w}min \leq x \leq x_{w}max$$
$$y_{w}min \leq y \leq y_{w}max$$

A = 1001
B = 1010
1000

0100
0010
0000

Q. consider window size from 5 to 9, clip foll line (4,12) (8,8)



P_1 = 1001 (outside)

P_2 =

Xmin
Xmax
Ymin
Ymax

(0000) inside

P_1 is outside so calc new P_1

$x_1 = 4$, $y_1 = 12$

$x_2 = 8$, $y_2 = 8$

$$m = \frac{y_2 - y_1}{x_2 - x_1} = -1$$

line intersect with x_{wmin}

Find y →

$y = y_1 + m(x - x_1)$

$x = 5 = x_{wmin}$

$y = 11$

∴ (5,11)

$x_{wmin} \leq x \leq x_{wmax}$ — 5

$y_{wmin} \leq y \leq y_{wmax}$ — 11

condition fails Apply algo:-

$(5,11)$ — $(8,8)$

$x_1 = 5, \; y_1 = 11$

$x_2 = 8, \; y_2 = 8$

$$m = \frac{x_2 - x_1}{y_2 - y_1} = -1$$

line intersect at y_{wmax}

Find x $\quad x = x_1 + \frac{(y - y_1)}{m}$

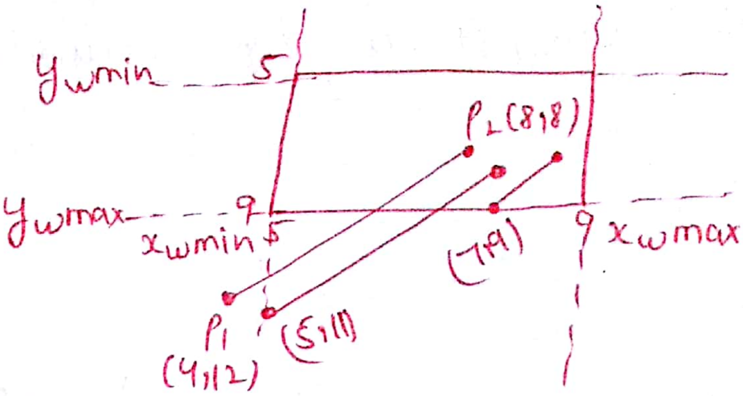
$= 7$

$(7,9)$

$x_{wmin} \leq 7 \leq x_{wmax}$

$y_{wmin} \leq 9 \leq y_{wmax}$.

consider window of size 5 to 9. clip line (4,12) (8,8)



$P_1 = 1001$ (outside)
$P_2 = 0000$ (inside)
0000

P_1 calculation

$x_1 = 4 \quad x_2 = 8$
$y_1 = 12 \quad y_2 = 8$

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{8-12}{8-4} = -1$$

line intersect with x_{wmin}

Find y.

$$y = y_1 + m(\underbrace{x}_{x_{wmax} / x_{wmin}} - x_1)$$
$$= 12 + (-1)(5-4)$$
$$= 11$$

(5,11) new pt.

$x_{wmin} \leq x \leq x_{wmax}$ ✓
$5 \leq 5 \leq 9$

$y_{wmin} \leq y \leq y_{wmax}$ (X)
$\times\ 5 \leq 11 \leq 9$

$(m \neq 0)$

(5,11) — (8,8)
$x_1\ y_1 \qquad x_2\ y_2$

$m = -1$

line intersect with y_{wmax} $= y = 9$

$$x = x_1 + \frac{(\underbrace{y}_{y_{wmax} / y_{wmin}} - y_1)}{m}$$
$$= 5 + \frac{(9-11)}{-1}$$
$$= 7$$

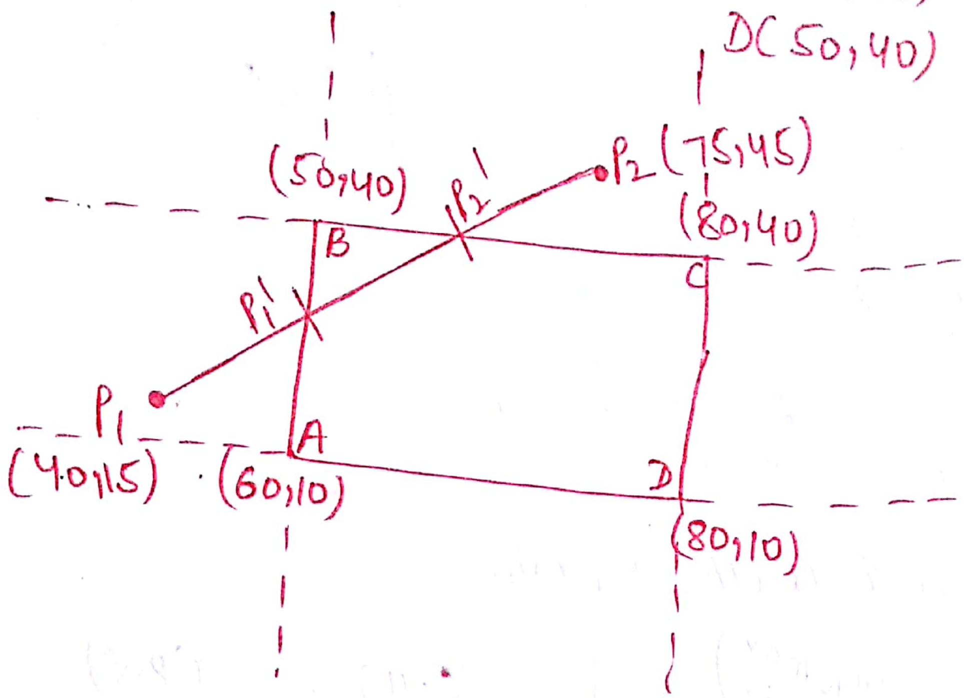
$5 \leq 7 \leq 9$

(7,9)

$(m \neq 0)$

Scanned by CamScanner

Q1. Use CS algo to clip line $P_1(40,15)$ $P_2(75,45)$ against a given window A(60,10) B(80,10) C(80,40) D(50,40)



P_1 = 0001 non zero
P_2 = 1000 non zero

0000

$$\text{Slope} = \frac{y_2 - y_1}{x_2 - x_1} = \frac{45-15}{75-40} = \frac{30}{35} = \frac{4+2}{7} = \frac{6}{7}$$

$x_{AB} = 50$

$$(y-15) = \frac{6}{7}(x-50)$$

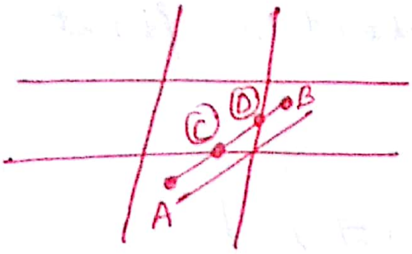
$$= \frac{6}{7}(15-50)$$

A = 0000
B = 0000 } Both 0000
0000

Trivially accepted

C = 1010
B = 1001
1000 ≠ 0000

Trivially rejected

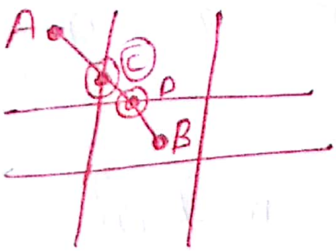


A = 0100
B = 0010
0000 = 0000

some part of line, inside viewing window.

C = 0000
B = 0010 > AND
0000 — Again check

D = 0000
C = 0000
0000 — accept.



A = 1001 non zero
B = 0000 zero
0000

Intersection point C

C = 1000
B = 0000
0000

D = 0000
B = 0000
0000 accept.

Line intersection & clipping

Intersection points are determined when line is partially visible i.e when both outcodes are different but AND operation is 0000.

∴ Intersection points will be :-

1) For left window edge, intersection point will be (x_L, y)

$$\boxed{y = m(x_L - x_1) + y_1 \quad , \quad m \neq \infty}$$

2) For Right window edge, intersection pt. will be (x_R, y)

$$\boxed{y = m(x_R - x_1) + y_1 \quad , \quad m \neq \infty}$$

3) For top window edge, " " " "
(x, y_T)

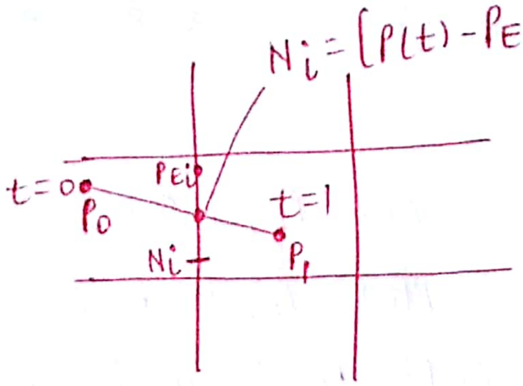
$$\boxed{x = x_1 + \frac{1}{m}(y_T - y_1) \quad , \quad m \neq 0}$$

4) For bottom window edge, (x, y_B)

$$\boxed{x = x_1 + \frac{1}{m}(y_B - y_1) \quad , \quad m \neq 0}$$

where, $\begin{bmatrix} x_L \le x \le x_R \\ y_B \le y \le y_T \end{bmatrix}$ y-value of bottom edge of clipping window.

Cyrus Beck (Parametric line clipping)



where $P(0) = P_0$; $P(1) = P_1$

$$\boxed{P_0P_1 = P(t) = P_0 + t(P_1 - P_0)}$$

$$\left[\begin{array}{l} x = x_0 + (x_1 - x_0)t \\ y = y_0 + (y_1 - y_0)t \end{array}\right]$$

Parametric eq.

Intersection [P(t)]

Dot product $\boxed{N_i \cdot [P(t) - P_{E_i}] = 0}$

$N_i \cdot [P_0 + t(P_1 - P_0) - P_{E_i}] = 0$

$0 = (N_i \cdot [P_0 - P_{E_i}] / -(N_i \cdot (P_1 - P_0))$

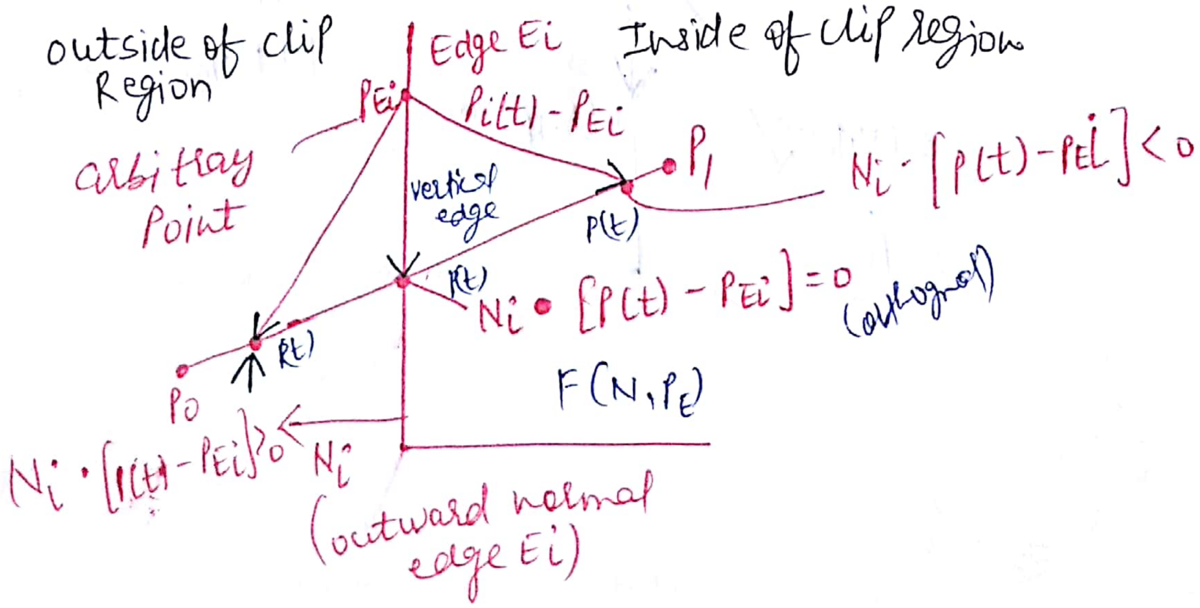
$\therefore \boxed{t = (N_i \cdot [P_0 - P_{E_i}]) / (-N_i \cdot D)}$ (1)

$\boxed{\text{where } D = P_1 - P_0}$

→ Compute (t) for each clip rectangle edge

Discard values of (t) outside [0,1]

→ they aren't on line segment (t_L, t_B, t_R)



→ Dot product determines whether $P(t)$ is

"inside the clip edge"

"outside " " " "

" on " " " "

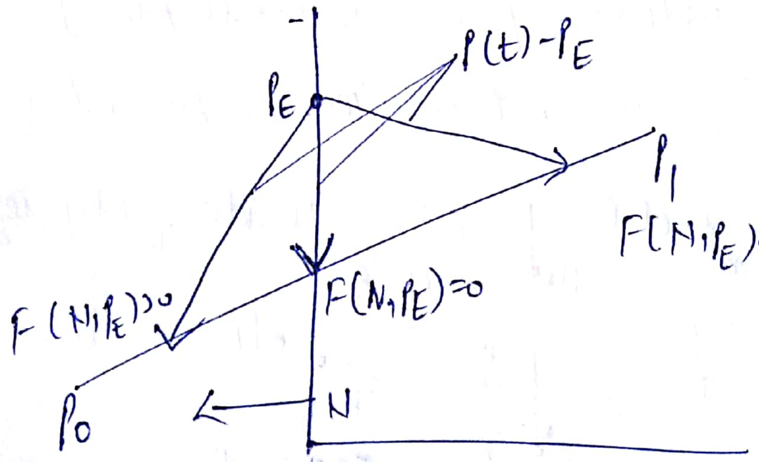
→ $P(t)$ is inside if $N_i \cdot [P(t) - P_{E_i}] < 0$

→ " " outside " $N_i \cdot [P(t) - P_{E_i}] > 0$

→ " " on " $N_i [P(t) - P_{E_i}] = 0$ (intersection point)

→ eq. ① is valid for foll. conditions :-

- $N_i \neq 0$
- $D \neq 0 \; (P_1 \neq P_0)$
- $N_i D \neq 0$ ($P_0 P_1$ not parallel to E_i)



Mid Point Subdivision

→ Recursive process in which we used Binary searching

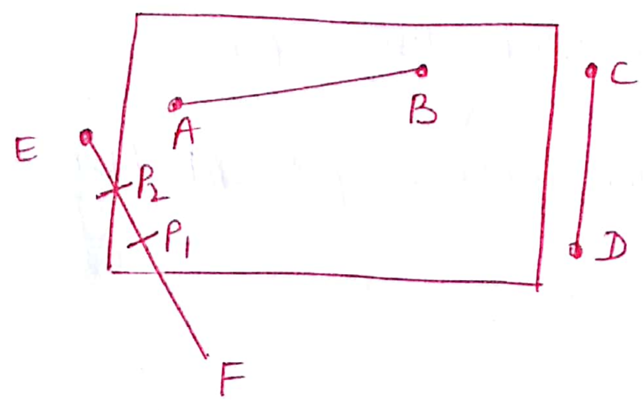
→ we compute outcodes for each of line segment that has been divided at its mid-point.

→ mid-pt coordinate $P_m(x_m, y_m)$ of a line segment joining $P_1(x_1, y_1)$ & $P_2(x_2, y_2)$ are:-

$$x_m = \frac{x_1 + x_2}{2}, \quad y_m = \frac{y_1 + y_2}{2}$$

→ For each of this sub-line segment outcodes are determined & again logical AND is taken.

Ex:-



a) line segment AB :- R.code for A = 0000

B = 0000

0000 — AB is completely visible.

b) CD :- C = 0010

D = 0010

0010 ≠ zero — totally invisible

c) EF :- E = 0001

F = 0100

0000 = zero

EF is not fully visible.

→ So, divide EF into 2 equal parts, say P_1 is its mid point.

→ 2 parts are :- EP_1 & P_1F

outcode for P_1 = 0000
" " E = 0001
AND = 0000 line is not fully invisible.

So, we again divide line EP_1 in 2 parts :- EP_2, P_2P_1 where P_2 is mid-point of EP_1 line.

outcode for P_2 = 0001
E = 0001
0001 ≠ zero

∴ discard line EP_2.

Ilrly we can proceed for P_1F line segment also.

Ans 3

= In mid pt. subdivision algo. the clipping line is divided into sub-line segments from mid-pt of line segments, unlike at every clipping boundaries in C-S algorithm.

→ In this algo, after giving outcodes to line segment given, Line segment is divided into 2 equal parts & now again outcodes are given to subline segments including their midpts.

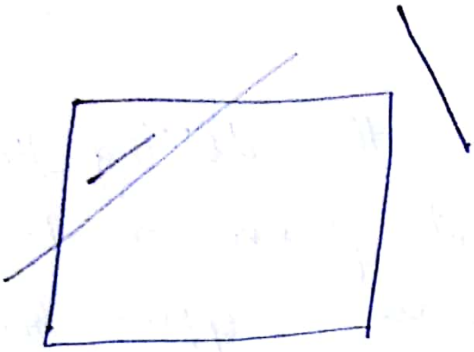
→ Each subdivided line segment is checked for possible rejection or acceptance.

→ Like in C-S algo, if logical AND operation performed on outcode & it gives non-zero result, then subdivision will be rejected. otherwise it will be accepted.

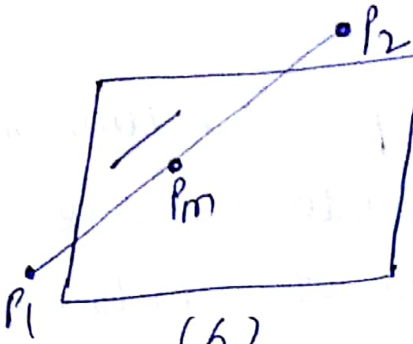
→ $P_1(x_1, y_1)$ & $P_2(x_2, y_2)$ are points of line P_1 & P_2 then mid pt value $P_m = ((x_1+x_2)/2, (y_1+y_2)/2)$ so new line is P_1P_m & P_mP_2 → now check if endpts are in visible area of clipping window.

If not further divide P_1P_m into $P_1P_{m_1}$ & $P_{m_1}P_m$.

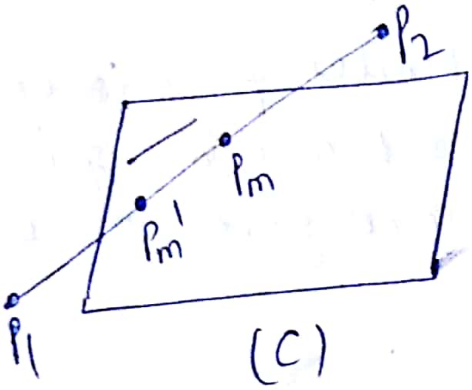
either visible or invisible totally.



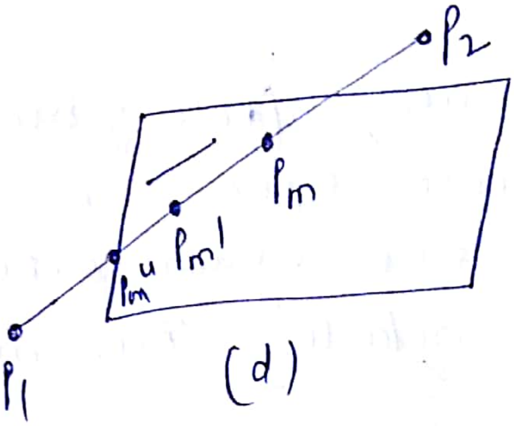
(a)



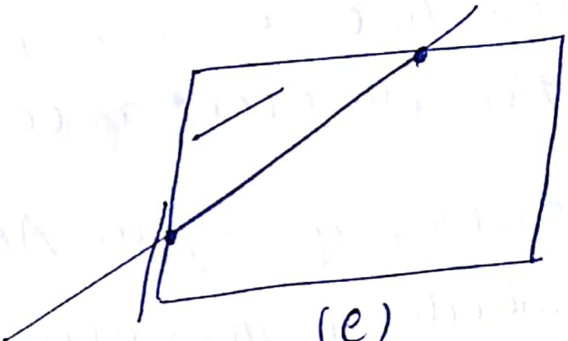
(b)



(C)



(d)



(e)

Liang Barsky

step1:- Get the end points $(x_1, y_1) - (x_2, y_2)$

step2:- Find out $\Delta x, \Delta y, P_1, P_2, P_3, P_4, q_1, q_2, q_3, q_4$

step3:- Assign $\left.\begin{array}{l} t1 = 0 \\ t2 = 1 \end{array}\right\}$ line intersection parameters.

(i) if $P_K = 0$ $(K = 1, 2, 3, 4)$ then line is parallel to win.

(ii) if $q_K < 0$ (") " " " outside window.

(iii) For non zero values of P_K

if $\boxed{P_K < 0}$ then find t_1 = segment potentially enters the clipping window

$t_1 = \max(0, q_K/P_K)$

else $\boxed{P_K > 0}$ then find t_2 = " " leaves " " "

$t_2 = \min(1, q_K/P_K)$

If $t_1 > t_2$ then line is completely outside - reject

or

else find new set of (x, y) if t_1, t_2 is changed

$$\left.\begin{array}{l} x = x_1 + t\Delta x \\ y = y_1 + t\Delta y \end{array}\right\} \text{parametric equation}$$

where $\left.\begin{array}{l} \Delta x = x_2 - x_1 \\ \Delta y = y_2 - y_1 \\ 0 \leq t \leq 1 \end{array}\right]$

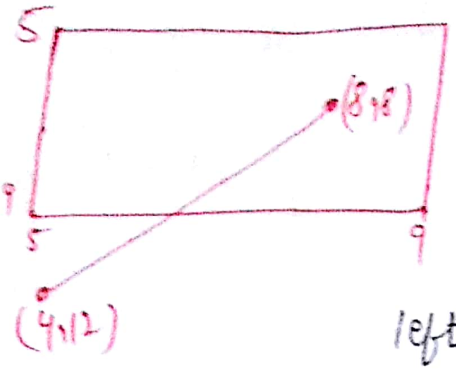
For a pt. (x, y) inside the clipping window, we have

$$\left[\begin{array}{l} xw_{min} \leq x_1 + \Delta x \cdot t \leq xw_{max} \\ yw_{min} \leq y_1 + \Delta x \cdot t \leq yw_{max} \end{array}\right.$$

$\boxed{P_K t \leq q_K}$ $K = 1, 2, 3, 4.$

Scanned by CamScanner

9.



$$\begin{bmatrix} P_1 = -\Delta x \\ P_2 = \Delta x \\ P_3 = -\Delta y \\ P_4 = \Delta y \end{bmatrix}$$

left — $q_1 = x_1 - x_{wmin}$
Right — $q_2 = x_{wmax} - x_1$
Bottom — $q_3 = y_1 - y_{wmin}$
Top — $q_4 = y_{wmax} - y_1$

$P_1 = (\overset{x_1}{4}, \overset{y_1}{12})$ $\quad P_2 (\overset{x_2}{8}, \overset{y_2}{8})$

$x_{wmin} = 5$ $\quad x_{wmax} = 9$
$y_{wmin} = 5$ $\quad y_{wmax} = 9$

$$\begin{bmatrix} \Delta x = x_2 - x_1 = 4 \\ \Delta y = y_2 - y_1 = -4 \end{bmatrix}$$

$\boxed{P_1 = -4}$, $P_2 = 4$, $P_3 = 4$, $\boxed{P_4 = -4}$

$q_1 = 4 - 5 = -1$
$q_2 = 9 - 4 = 5$
$q_3 = 12 - 5 = 7$
$q_4 = 9 - 12 = -3$

Initial value $t_1 = 0$
$t_2 = 1$

$\boxed{P_1, P_4 < 0}$

$t_1 = \max(0, q_1/P_1, q_4/P_4)$
$= \max(0, 1/4, 3/4)$
$t_1 = 3/4$

$\boxed{P_2, P_3 > 0}$

$t_2 = \min(1, q_2/P_2, q_3/P_3)$
$= \min(1, 5/4, 7/4)$
$t_2 = 1$ $\quad \therefore$ $t_1 < t_2$

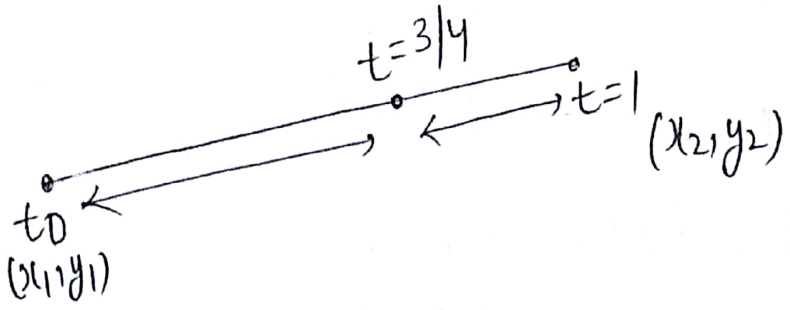
$\rightarrow$ $\boxed{\begin{aligned} x_1 &= x_1 + t_1 \Delta x \\ &= 4 + 3/4 \cdot 4 = 7 \end{aligned}}$

$\rightarrow$ $\boxed{\begin{aligned} y_1 &= y_1 + t_1 \Delta y \\ &= 12 + 3/4(-4) \\ &= 9 \end{aligned}}$

$x_2 = x_2 + t_2 \Delta x$
$= 8 + 4 = 8 + 4 = 12$

$y_2 = y_2 + t_2 \Delta y$
$= 8 + (-4) = +4$

Scanned by CamScanner



at start of line (x_1, y_1) $t=0$

" end " " (x_2, y_2) $t=1$

at $3/4^{th}$ of line path

$\boxed{t = 3/4}$

The location is

$$\boxed{\begin{aligned} x &= 1/4\, x_1 + 3/4\, x_2 \\ y &= 1/4\, y_1 + 3/4\, y_2 \end{aligned}}$$

<u>At time t</u>

$$x = (1-t)\cdot x_1 + t \cdot x_2$$

$$y = (1-t)\cdot y_1 + t \cdot y_2$$

$$x = x_1 - x_1 t + t \cdot x_2$$
$$= x_1 + t(x_2 - x_1)$$
$$= x_1 + t\Delta x$$

lly $y = y_1 - y_1 t + t \cdot y_2$
$$= y_1 + t(y_2 - y_1)$$
$$= y_1 + t\Delta y$$

we know,

$$x_{wmin} \leq x \leq x_{wmax}$$
$$y_{wmin} \leq y \leq y_{wmax}$$

<u>Substitute x, y value</u>

$$x_{wmin} \leq x_1 + t\Delta x \leq x_{wmax}$$
$$y_{wmin} \leq y_1 + t\Delta y \leq y_{wmax}$$

where,

$$x_1 + t\Delta x \geq x_{wmin}$$
$$x_1 + t\Delta x \leq x_{wmax}$$
$$y_1 + t\Delta y \geq y_{wmin}$$
$$y_1 + t\Delta y \leq y_{wmax}$$

$$\therefore\ t\Delta x \geq x_{wmin} - x_1$$
$$t\Delta x \leq x_{wmax} - x_1$$
$$t\Delta y \geq y_{wmin} - y_1$$
$$t\Delta y \leq y_{wmax} - y_1$$

multiply by $\ominus$

$$-t\Delta x \leq x_1 - x_{wmin}$$
$$t\Delta x \leq x_{wmax} - x_1$$
$$-t\Delta y \leq y_1 - y_{wmin}$$

$tp_K \leq q_K \ (K = 1, 2, 3, 4)$

$t = 0 \qquad t_2 = 1$

But here $t_1 = 3/2$

$t_2 = 1$

$\therefore \quad x = x_1 + t_1 \Delta x = 7$

$y = y_1 + t_1 \Delta y = 9$

But t_2 remains same.

Nicholl- Lee -Nicholl (NLN) Line Clipping

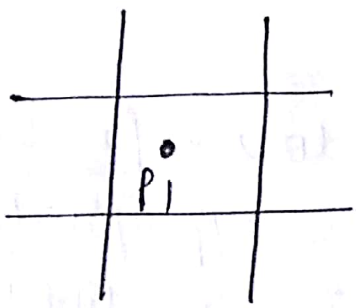
→ Symmetry algorithm that categories the endpoints into one of the 3 regions:- | Clip window
edge region
corner region.

→ Then according to the endpt. category it checks several clip cases.

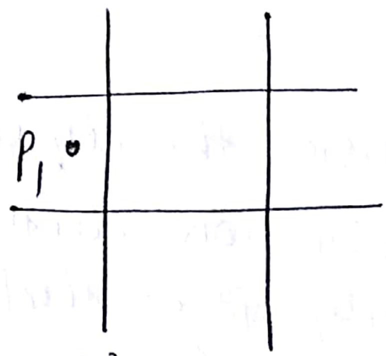
→ Before calculating intersection pts, more region testing is done in this. So, no extra calculations are involved here.

→ Suppose, we have a line AB with endpoints as P_1 & P_2.

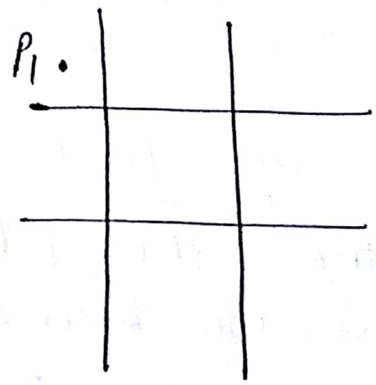
↳ First, we find out the position of point P_1 for 9 possible regions relative to clipping rectangle.



(a) P_1 in clip window



b) P_1 in edge region



c) P_1 in corner region.

→ Here, we will be considering these 3 regions & if P_1 lies in any other (six) region, then we can move it to one of the 3 regions by using a symmetry.

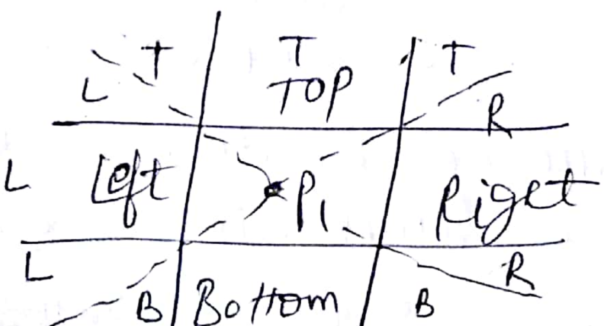
→ suppose if we have a region directly above the clip window can be transformed to the region left of clip window using a reflection about the line $y = -x$ & or we could use a 90° anticlockwise rotation.

→ Secondly, we find out the position of P_2 relative to P_1.

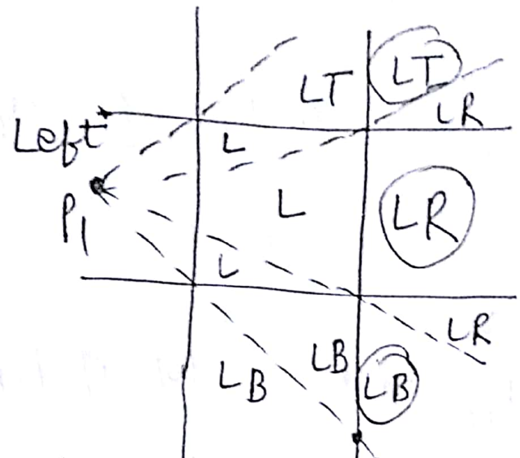
↳ So, we create some new regions in the plane, depending upon the location of P_1 & pass through the window corners.

Algorithm steps

① If P_1 is inside the clip window & P_2 is outside then the intersection with the appropriate window boundary is carried out depending upon one of the 4 regions (L, T, R, B) contains P_2



② If P_1 is to left of window then we set 4 regions say, L, LT, LR & LB i.e

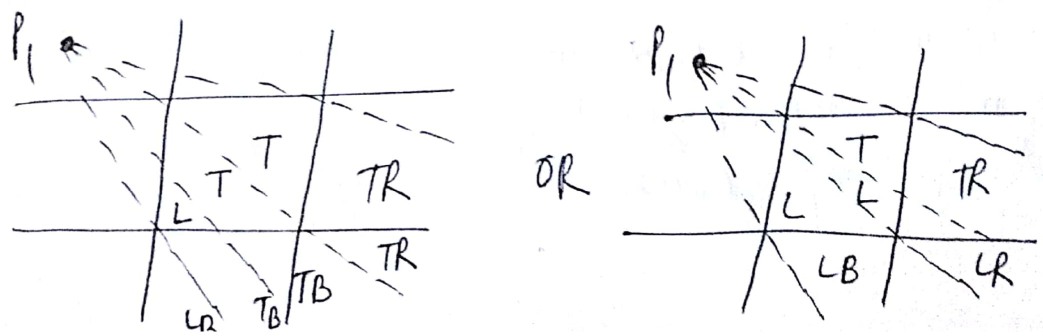


→ These 4 regions determine a unique boundary for line segment.

→ If P_2 is in region L, then we clip line at left boundary & save line segment from this intersection pt. to P_2.

But if P_2 is in region LT, we save line segment from the left window boundary to top boundary.

③ If P_1 is to left & above the Clip window the 2 case arises:-



where P_2 is located?

To find P_2, we compare the slope of line to slopes of boundaries of clip region.

For eg:- If P_1 is to left of clipping rectangle then P_2 is in region LT if :-

$$\boxed{\text{slope of } P_1P_{TR} \text{ line} < \text{slope of } P_1P_2 < \text{slope of } P_1P_{TL} \text{ line}}$$

OR
$$\frac{y_T - y_1}{x_R - x_1} < \frac{y_2 - y_1}{x_2 - x_1} < \frac{y_T - y_1}{x_L - x_1}$$

Also, we clip entire line if :-

$$(y_T - y_1)(x_2 - x_1) < (x_2 - x_1)(y_2 - y_1)$$

T = ray intersects top boundary
L = " " left "
R = " " Right "
B = " " Bottom "
LT = " " Left & TOP "
LR = " " " & Right "
LB = " " " & Bottom "
TR = " " TOP & Right "
TB = " " TOP & Bottom "