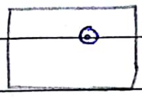
5/1/18

Computer Graphics

Schaum Series
Hern & Baker.

~~Screen (2-D)~~

Resolution 

higher resolution, more cost

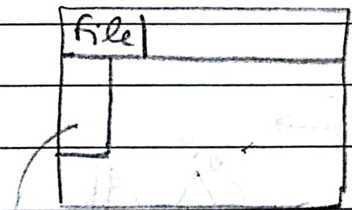
In resolution, smallest unit → dot, pixel

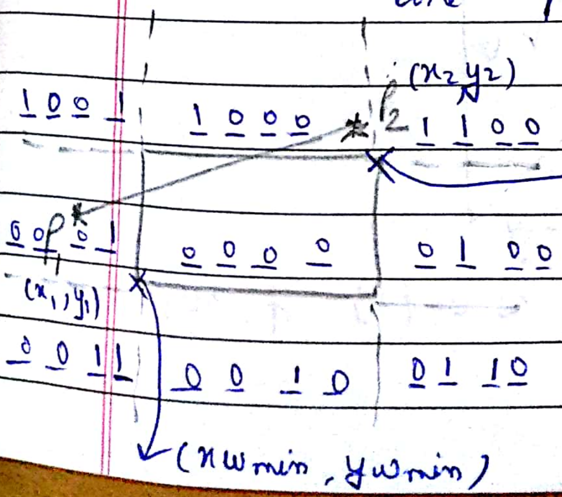
pix el → picture element

pixel is used for → 1) drawing
2) manipulation
3) coloring

Drawing — primitive — mathematical eq^n known
— not known
→ Basic ones – line
circle
ellipse
parabola
hyperbola

Manipulation — clipping — line, polygon


dropdown menu when cursor over file



(xw_{max}, yw_{max})

Address needs to be 4 bit coz we have 9 parts of the grid.

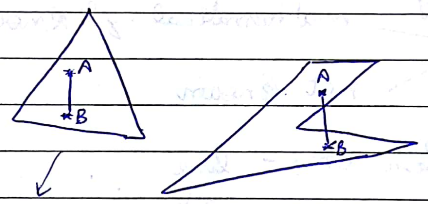
LEFT 0001
BOTTOM 0010
RIGHT 0100
TOP 1000

Make a function
i/p → x_1, y_1, x_2, y_2, xw

PS: a point on the window is considered inside the window.

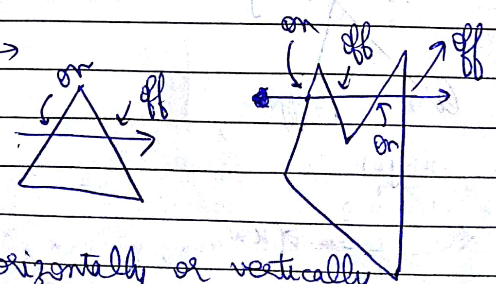
conditions → $x_{w\,min} \le x \le x_{w\,max}$

$y_{w\,min} \le y \le y_{w\,max}$



a virtual line is moved inside the polygon such that both points are inside, we check if entire line is also inside the polygon

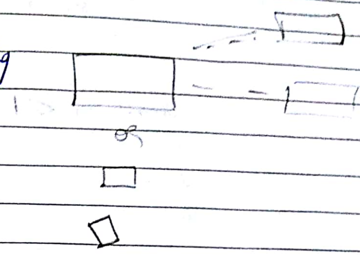
colouring →



Scanning horizontally or vertically
Illumination start when parity is on(1) & stops when parity is off (0).

2D objects — Transformation

Scaling, rotation or shifting



Camera → DLL (driver files)

→ rotate
hidden surfaces (dotted lines)

→ simulate, automate

font development → used curves.
→ different 2-D curves
→ designing characters using curves
→ angles, maths is involved.
→ blending at each posⁿ of curve with iteration

Adobe → main work → font dev.

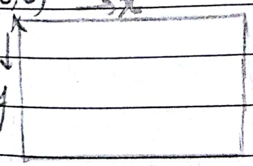
Primitive Drawing

Line

eqⁿ → $y = mx + c$

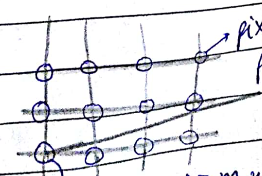
(0,0) → x
y ↓

origin by default



Screen

2D Matrix



pixel position

Grid (x_i, y_i) $y_i = mx_i + c$

Slope → $= \infty$, $= 0$, < 1, > 1, $= 1$ — 5 cases for line → vector

or → 2 end pts.

slope is < 1

$y = mx + c$

l value

r value

approach

I $\underline{m < 1}$ ① DDA (digital Differential Analyser)

Line

x	10	11
y	10	

keep incrementing x.

★ Major movement in x

For ★ Major movement in y

→ increment y.

$$\begin{cases} x_{i+1} = x_i + 1 \\ y_{i+1} = y_i + m \end{cases}$$

inputs → x_1, y_1, x_2, y_2

$x = x_1$

$y = y_1$

$dx = x_2 - x_1$

$dy = y_2 - y_1$

if abs(dy) < abs(dx)

{

Putpixel (x, y, WHITE);

y+ = (float)

If we apart

II $\underline{m > 1}$

classmate
Date
Page

$y += m$ → ★ Floating point computation will take more time always.

$x++;$

putpixel (x, y, WHITE);

y+ = (float) dy/dx

round (y)

}

}

If we increment y here, dots will be far apart . . . . . .

II $\underline{m>1}$ → $y_{i+1} = y_i + 1$

$x_{i+1} = x_i + 1/m$

$$y_{i+1} = m x_{i+1} + c$$

$$\frac{y_i + 1 - c}{m} = x_{i+1}$$

$$\frac{y_i - c}{m} + \frac{1}{m} = x_{i+1}$$

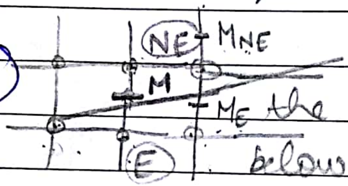
```
if abs(dy) > abs(dx)
{
    putpixel (x, y, WHITE);
    while (y <= y2)
    {
        y++;
        x += 1/m;
        putpixel (x, y, WHITE);
    }
}
```

```
int round (float p)
{
  if (p - (int) p < 0.5)
     return p;
  else
     return p+1;
}
```

approach II

linear Inequality — mid point approach

I



the line is closer to above pt or below pt

$f(x, y) = ax + by + c \longrightarrow = 0$

< 0

> 0

(decision part) $d = f(M) = f(x_i + 1, y_i + 1/2)$

$= a(x_i + 1) + b(y_i + 1/2) + c$

initialisation $= \underbrace{ax_i + by_i + c}_{\text{initial pt so } 0} + a + b/2 = a + b/2$

if $d < 0$

Selection on E

$d_{new} = f(M_E) = f(x_i + 2, y_i + 1/2)$

$= a(x_i + 2) + b(y_i + 1/2) + c$

Noting of change → differential

Whenever for a primitive drawing, we need to note a change, we use differential.

~~first derivative $\rightarrow f(x+h) - f(x)$~~

$\Delta E = d_{new} - d$

$= a(x_i+2) + b(y_i+1/2) + c - [a(x_i+1) + b(y_i+1/2) + c]$

$= \underset{\text{constant}}{a}$ $= dy$

$y = mx + c$ $\frac{dy}{dx}$

$dx \cdot y = dy \cdot x + c \cdot dx$

$dy \cdot x - dx \cdot y + c \cdot dx = 0$ —— ①

$ax + by + c = 0$ —— ②

Since initialization is a+b/2, it is a floating pt, we have a property of line by which it does not change its characteristics when multiplied by a scalar. Hence, decision will be same.

$$\Delta E = 2dy$$

$$\Delta NE = 2(dy - dx)$$

$$d_i = 2dy - dx$$

```
x = x1   y = y1
dx = x2 - x1 , dy = y2 - y1
        d = 2 * dy - dx;
putpixel (x, y, WHITE);
while (x <= x2)
{
    if (d<0)   /* selection is (E) */
        { d += 2*dy ; }
    else
        { d += 2 * (dy - dx)
          y++;
        }
    x++;
putpixel (x, y, WHITE);
```

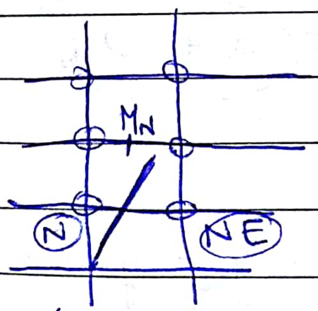
II

x_1	y_1	x_2	y_2
10	10	20	18

$dx = 10 \quad dy = 8.$

x_i	y_i	d
10	10	6
11	11	2
12	12	-2
13	12	14
14	13	10
15	14	6.
16	15	2
17	16	-2
18	16	14
19	17	10
20	18	6.

II Do for $m > 1$.



(x_i, y_i)

$x = x_1, \; y = y_1$

$dx = x_2 - x_1$

$dy = y_2 - y_1$

~~Wheather~~ whether start N or NE

Major movement y.

$$y_{i+1} = y_i + 1$$

$d = F(M) = F(x_i + 1/2 \, , y_i + 1)$

if selection N

$d_{new} = F(M_N)$

$= F(x_i + 1/2 \, , y_i + 2)$

$\Delta N = b$ (coz doubled) $2b$

a replaced by b $= 2dx$

when selection NE,

$d_{new} = F(M_{NE})$

$= F(x_i + 3/2, \; y_i + 2)$

$= 2(a+b)$

$= 2(dy + dx)$

$d_i = a/2 + b = \boxed{a + 2b}$

$d = -dy + 2dx$

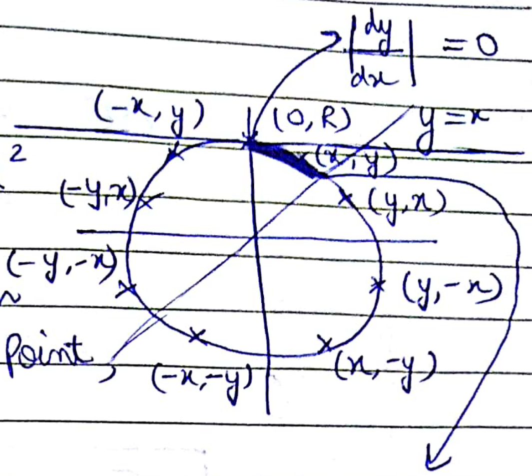
```
x=x1, y=y1, dx=x2-x1, dy=y2-y1
d = 2dx-dy , putpixel(x,y,WHITE)
while(y<=y2) {
   if(d<0) {
        d+=2dx; }
   else {
        d+=2(dx-dy);
   }    x++;
   y++;
} putpixel(x,y,WHITE);
```

8/1/18

Scan Conversion

Primitive

Circle → $x^2 + y^2 = R^2$



Drawing in clockwise fashion starting from (0,R), point by point, pixel by pixel

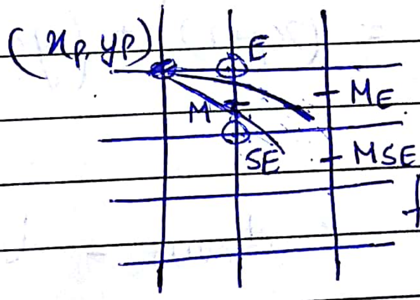
$\left|\frac{dy}{dx}\right| = 1.$

8-symmetry system.

In this octant, $\left|\frac{dy}{dx}\right|$ is b/w 0 and 1. $m<1$

→ major movement in x.

```
void circle_symmetry (x,y)
{
        putpixel (x, y, WHITE);
        putpixel (y, x, WHITE);
        |
        |
        |
}
```



At (x_p, y_p)

$f(x_p, y_p) = x_p^2 + y_p^2 - R^2$ —— ①

For decision consideration,

$$f(M) = f(x_p + 1, y_p - 1/2)$$

$$= (x_p + 1)^2 + (y_p - \tfrac{1}{2})^2 - R^2 \text{ —— } ②$$

if $d < 0$, selection is E

$d_{new} = f(M_E) = f(x_p + 2, y_p - 1/2)$

$= x_p^2 + 4x_p + 4 - (x_p^2 + 2x_p + 1)$

$$\boxed{\Delta E = 2x_p + 3}$$

else

selection is SE

$d_{new} = f(M_{SE})$

$= f(x_p + 2, y_p - 3/2)$

$= (x_p+2)^2 + (y_p - 3/2)^2 - R^2$ ——— (4)

Initialization

$$d(0,R) = 5/4 - R \approx 1 - R.$$

$$\begin{cases} (0+1)^2 + (R-\frac{1}{2})^2 - R^2 \\ 1 + \cancel{R^2} - R + \frac{1}{4} - \cancel{R^2} \\ \frac{5}{4} - R \end{cases}$$

```
x=0;
y=R;
d=1-R;
void circle Symmetry (x, y)
{ while (x<=y)
  { if d<0
      d+=2*x+3;
    else
    { d+=2(x-y)+5;
      y--;
```

circle Symmetry (x, y);

}
}

DRY RUN

R=8 case

x	y	d
0	8	-7
1	8	-4
2	8	1
3	7	-6
4	7	3
5	6	2
6	5	5
7	4	12

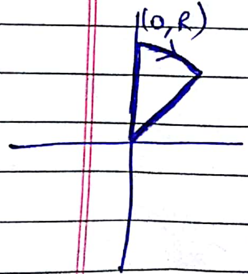
×

Qu. Can we 2nd differential here?

Yes, as circle eq^n is degree 2.

But why is 2nd differential needed?

$d = 2x + 3$, → first derivative



Integer addition is faster than multiplication

hence, we find 2nd derivative which would be a const.

2nd derivative ke liye, we need 1st derivative first.

At (x_p, y_p)

$$\Delta E = 2x_p + 3$$

$$\Delta SE = 2(x_p - y_p) + 5$$

if $d < 0$

/* selection E */

At E $[x_p + 1, y_p]$

$$\left.\begin{aligned} \Delta E_{new} &= 2(x_p) + 5 \\ \Delta SE_{new} &= 2(x_p - y_p) + 7 \end{aligned}\right\} \quad \text{--- (2)}$$

eq (2) − (1)

$$\Delta^2 E^{(E)} = 2$$

$$\Delta^2 SE^{(E)} = 2$$

else,

/* selection is SE */

At SE $[x_p + 1, y_p - 1]$

$$\left.\begin{aligned} \Delta E_{new} &= 2(x_p) + 5 \\ \Delta SE_{new} &= 2(x_p - y_p) + 9 \end{aligned}\right\} \quad \text{--- (3)}$$

$$\Delta^2 E^{(SE)} = 2, \quad \Delta^2 SE^{(SE)} = 4$$

initialisation, at 0, R

$d = 1 - R$

$\Delta E = 3$, $\Delta SE = 5 - 2R$

```
x = 0;
y = R;
d = 1-R;
delta E = 3
delta SE = 5-2R

void circlesymmetry (x, y) {
while (x<=y)
```

2/1/18

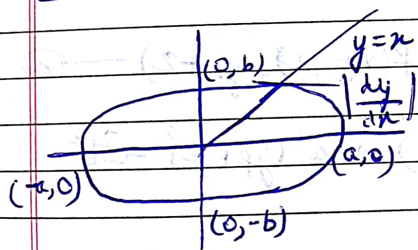
(k=8)

x	y	d	delta E	delta SE
0	8	-7	3	-11
1	8	-4	5	-9
2	8	1	7	-7
3	7	-6	9	-3
4	7	3	11	-1
5	6	2	13	3

R | 1/18

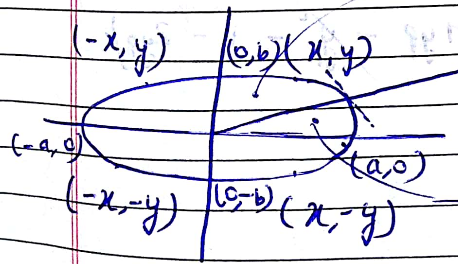
Mid-Point approach

Ellipse



$\left|\frac{dy}{dx}\right| \neq 1$ 4-way Symmetry

→ Region II abs(slope of tangent) < 1.



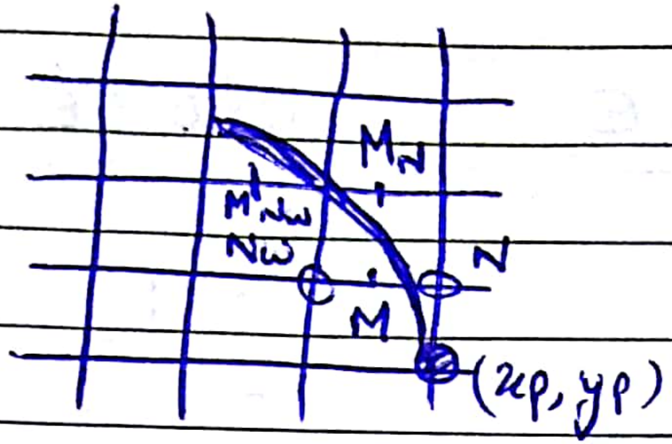
$\left|\frac{dy}{dx}\right| = 1$ (slope of tangent is 1 here)

→ Region I abs(slope of tangent > 1)

$$f(x,y) = b^2x^2 + a^2y^2 - a^2b^2$$

$$b^2 2x + a^2 2y \frac{dy}{dx} = 0 \therefore dy = \frac{-b^2x}{a^2y}$$

Region I



Major movement in Y

At (x_p, y_p)

$$f(x_p, y_p) = b^2 x_p^2 + a^2 y_p^2 - a^2 b^2$$

$$d^{I} = f(M) = f(x_p - 1/2, y_p + 1)$$

$$= b^2 (x_p - 1/2)^2 + a^2 (y_p + 1)^2 - a^2 b^2 \quad (1)$$

If $d^{I} < 0$,

Selection is N

Date ____
Page ____

$$= b^2(x_p - 3/2)^2 + a^2(y_p + 2)^2 - a^2b^2$$

eq (3) − eq (1)

$$\Delta NW = a^2(2y_p + 3) + 2b^2(1 - x_p)$$

$$d_i^{I}(a,0) = f(a - 1/2,\ 1)$$

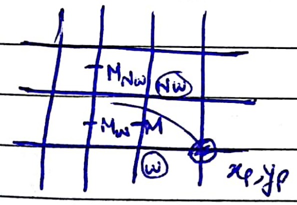
$$= b^2(a - 1/2)^2 + a^2(1) - a^2b^2$$

$$= b^2\left[a^2 + \frac{1}{4} - a\right] + a^2 - a^2b^2$$

$$= \frac{b^2}{4} - ab^2 + a^2$$

$$= a^2 + b^2\left(\frac{1}{4} - a\right)$$

Region II



$$d^{II} = f(x_p - 1,\ y_p + 1/2)$$

$$= b^2(x_p - 1)^2 + a^2(y_p + 1/2)^2 - a^2b^2 \quad \text{(4)}$$

if $d^{II} < 0$

selection is NW

$$d_{new}^{II} = f(x_p - 2,\ y_p + 3/2)$$

$$= b^2(x_p - 2)^2 + a^2(y_p + 3/2)^2 - a^2b^2 \quad \text{(5)}$$

$\Delta NW = b^2(3 - 2x_p) + 2a^2(1 + y_p)$

else

selection is N

$d^{II}_{new} = f(x_p - 2, y_p + 1/2)$

$= b^2(x_p - 2)^2 + a^2(y_p + 1/2)^2 - a^2b^2$ ——— 6

/* Re

eq 6 − eq 4

$\Delta W = b^2(\cancel{x_p^2} + 4 - 4x_p - \cancel{x_p^2} - 1 + 2x_p)$

$= b^2(-2x_p + 3)$

Code –

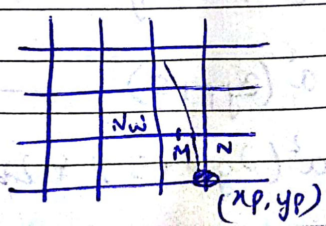
```
x=a;
y=0;
d = a*a + 0.25 *(b*b) - a*b*b;
while (b²|x| > a²|y|)
{
    if (d<0)
        d+ = a*a*[2*yp + 3]
```

gion I*/

```
else
    { d+ = (a*a*(2*y+3)) + (2*b*b*(1-x));
      x--;
    }
    y++;
}
d = b^2*(x-1)^2 + a^2*(y+1/2)^2 - a^2b^2;
while (x>0)
{
/* Region II */
        if (d<0)
        { d+ = (b*b*(3-2*x)) + (2*a*a*(1+y));
          y++;
        }
        else
        { d+ = b*b*(3-2*x);

        }
        x--;
}
```

Ellipse (2nd Differential)

⇒ Region I



$$\begin{cases} \Delta N = a^2(2y_p+3) \\ \Delta NW = a^2(2y_p+3) + 2b^2(1-x_p) \end{cases}$$

Date
Page

initialization, at $(a, 0)$

$\Delta N = 3a^2$

$\Delta NW = 3a^2 + 2b^2(1-a)$

$d^{I} = f(x_p - 1/2, y_p + 1)$

if $d^{I} < 0$

Selection (N)

$(x_p, y_p + 1)$

$\Delta N_{new} = a^2(2y_p + 5)$ —— (3)

$\Delta NW_{new} = a^2(2y_p + 5) + 2b^2(1 - x_p)$ —— (4)

eq (3) - eq (1) & eq (4) - (2)

$\Delta^2 N_{(N)}^{\boxed{I}} = 2a^2$

$\Delta^2 NW_{(N)}^{\boxed{I}} = 2a^2$

else if $d^{I} > 0$

selection (NW)

$(x_p - 1, y_p + 1)$

$\Delta N_{new} = a^2(2y_p + 5)$ —— (5)

$\Delta NW_{new} = a^2(2y_p + 5) + 2b^2(2 - x_p)$ ——

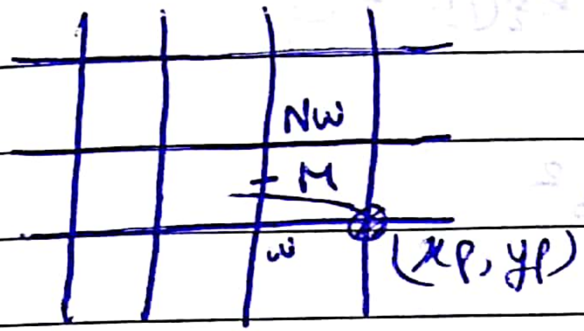
⟹ Reg

$\Delta^{\dots}_{NW}$

$$\Delta^2 NW^{\boxed{II}}_{NW} = 2a^2 + 2b^2$$

$$= 2(a^2 + b^2)$$

⇒ Region II



$$\Delta W = b^2(3 - 2x_p) \quad \text{—— (7)}$$

$$\Delta NW = b^2(3 - 2x_p) + 2a^2(1 + y_p) \quad \text{—— (8)}$$

$$d^{II} = f(x_p - 1, y_p + 1/2)$$

if ($d^{II} < 0$)
Selection is NW.

else if

$d^{I} > 0$

Selection is W

@ ~~(x_p, y_p+1)~~ (x_p-1, y_p)

$\Delta W_{new} = b^2(3 - 2x_p)$ —— (11)

$\Delta NW_{new} = b^2(3 - 2x_p) + 2a^2(1 + y_p)$ —— (12)

eq (11) − (7) & eq (12) − (8)

$\Delta W^{2\,\boxed{II}}_{(W)} = 2b^2$

$\Delta NW^{\boxed{II}}_{(W)} = 2b^2$

Difficulties / issues

1) Floating pt computation.
2) Mid point approach.
    ↳ overcomes the problems in floating pt approach.
3) Bresenham approach

But,

$f(M) < 0$ / > 0 | deciding this is an issue.

↓

overcomes the problems associated with Mid pt approach.

(3) Propose — (1) Measurable Quantity
   ↳ difference in Measurable quantity

(2) Use the measurable quantity in finding the differentials.

(I) $m < 1$

$\boxed{\text{line}}$

$y = mx + c$

$$= 2\left[\frac{\Delta y}{\Delta x}(x_p+1)+c\right] - y_p - (y_p+1)$$

$$D^{\boxed{A}} = D\cdot\Delta x = 2\left[\Delta y\,(x_p+1) + c\cdot\Delta x\right] - y_p\Delta x - (y_p+1)\cdot\Delta x$$

Actual decision.

$$= 2\Delta y\, x_p - 2\Delta x\, y_p + C$$

(II)

If $D<0$,

selection is (E)

$D_{initialisation} = $ Using (A),

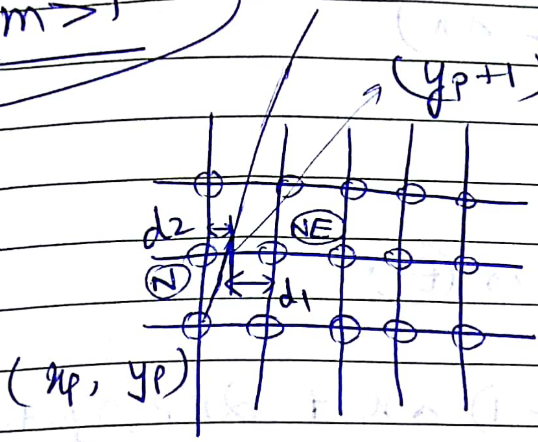
$D \cdot \Delta x = 2\Delta x \, [\overbrace{m x_p + c - y_p}^{0}]$

$+ 2\Delta y - \Delta x.$

$\rightarrow 2\Delta y - \Delta x$

(I) $m > 1$

$(y_p+1) = m x_p + c$



$d_1 = -(x - (x_p+1))$

$d_2 = -(x_p - x)$

$x = \frac{1}{m}\left[[y_p+1] - c \right]$

$D = d_1 - d_2$

$= -(2x - (x_p+1) - x_p)$

$D = -2\left[\frac{(1+y_p) - c}{m}\right] + (x_p+1) + x_p.$

$= -2\left[\frac{(y_p+1) - c}{\Delta y}\right]\Delta x + (x_p+1) + x_p$

$D^{(A)} = D \cdot \Delta y = -2[(y_p+1) - c]\,\Delta x + x_p \Delta y + (x_p+1)\Delta y$

$= -2y_p \Delta x + 2x_p \Delta y + c$

Scanned by CamScanner

selection is E

$D_{new} = -2(y_p + 1)\Delta x + 2x_p \Delta y + c$

$\Delta E = -2\Delta x$

$d_D = |(x_i+1)^2 + (y_i - 1)^2 - R^2|$

$d_V = |x_i^2 + (y_i - 1)^2 - R^2|$

Case ①

$OD^2 - R^2 < 0$

$OD^2 < R^2$

Case ③

$OD^2 - R^2 > 0$

$OD^2 > R^2$

Selection b/w H & D

If one wants to select b/w H & D

$$= (x_i+1)^2 + (y_i-1)^2 - R^2 + y_i^2 + (x_i+1)^2 - R^2$$

$$= 2[(x_i+1)^2 + (y_i-1)^2 - R^2] + 2y_i - 1$$

$\left[\because OD^2 < R^2 \quad OH^2 > R^2\right]$

$$\left[\underbrace{+y_i^2 - (y_i-1)^2}\right]$$

$$\boxed{\delta_{HD} = 2\Delta D_i + 2y_i - 1}$$

<u>Selection between V & D</u>

$$\delta_{VD} = d_V - d_D$$

$$= |OV^2 - R^2| - |OD^2 - R^2|$$

$\left\{\because OD^2 > R^2 \quad OV^2 < R^2\right\}$

$$= (R^2 - OV^2) - (OD^2 - R^2)$$

$$\delta_{VD} = 2R^2 - (x_i^2 + (y_i-1)^2) - (x_i+1)^2 - (y_i-1)^2$$

$$= 2[R^2 - (x_i+1)^2 - (y_i-1)^2] - x_i^2 + (x_i+1)^2$$

$$= 2[R^2 - (x_i+1)^2 - (y_i-1)^2] + 2x_i + 1$$

19/11

Part I

Illuminating pixel H.

$$\Delta D_i = (x_i+1)^2 + (y_i-1)^2 - R^2$$

@ $H(x_{i+1}, y_i)$ $\quad \Delta D_{i+1} = (x_i+2)^2 + (y_i-1)^2 - R^2$

$$\Delta H = \Delta D_{i+1} - \Delta D_i$$
$$= 2x_i + 3$$

Part II

Illuminating pixel V

$$\Delta D_i = (x_i+1)^2 + (y_i-1)^2 - R^2$$

@ $V(x_i, y_i-1)$ $\quad \Delta D_{i+1} = (x_i+1)^2 + (y_i-2)^2 - R^2$

$$\Delta V = \Delta D_{i+1} - \Delta D_i$$
$$= -2y_i + 3$$

Part III

Illuminating pixel D

$$\Delta D_i = (x_i+1)^2 + (y_i-1)^2 - R^2$$

@D (x_{i+1}, y_i-1)

$$\Delta D_{i+1} = (x_i+2)^2 + (y_i-2)^2 - R^2$$

$$= 2x_i + 6 - 2y_i$$

$$= 2(x_i - y_i) + 6.$$

Initialisation

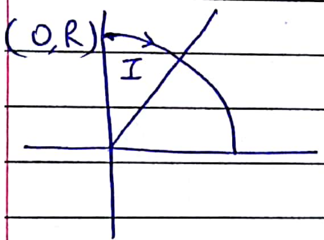
$$\Delta D_{i\,(0,R)} = 1 + (R-1)^2 - R^2$$

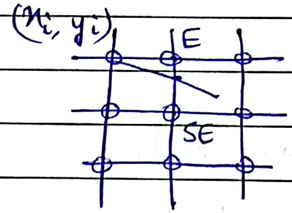
$$= 1 + \cancel{R^2} - 2R + 1 - \cancel{R^2}$$

$$= 2(1-R)$$

Measurable quantity is always looked wrt to degree the equation of the primitive in this approach.

Eg: here, degree is 2, measurable quantity is in degree 2 :





with reference to E

$d_1 = y_i^2 - y^2$

$d_2 = y^2 - (y_i - 1)^2$ → SE.

$$\boxed{y^2 = R^2 - (x_i+1)^2}$$

$D_i = d_1 - d_2$

$= y_i^2 + (y_i - 1)^2 - 2y^2$

$$D_i = y_i^2 + (y_i - 1)^2 - 2[R^2 - (x_i+1)^2] \quad \text{—— (1)}$$

If $D_i < 0$
selection is E

eq ② − ①

$$\Delta E = D_{i+1} - D_i$$
$$= 2(2x_i + 3)$$

Else

Selection is SE $x_{i+1}, y_i - 1$

$$D_{i+1} = (y_i - 1)^2 + (y_i - 2)^2 - 2[R^2 - (x_i + 2)^2] \quad ③$$

Initialisation,

$$D_i(0, R) = 3 - 2R$$

eq ③ − ①

$$\Delta SE = 2(2x_i + 3) + 4(1 - y_p)$$

// Code

```
x=0;
y=R;
d=3-2R;
while (x<=y)
{
  if d<0
     d += 2*(2x+3);
  else
```

x	y	d
0	8	-13
1	8	-7
2	8	3
3	7	-11
4	7	7
5	6	5
6	5	

$3 + (14) + -22$

Second differential

$d = 3 - 2R$

$\Delta E = 2 * (2x_i + 3)$

$\Delta E_{new} = 2(2x_i + 5)$

$$\boxed{\Delta^2 E^{SE} = 4}$$

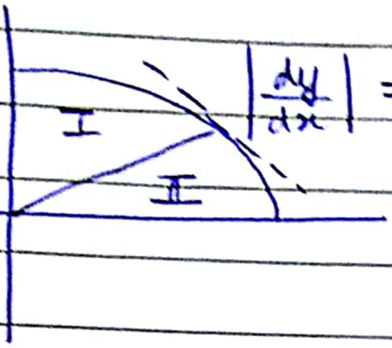
$\Delta SE_{new} = 2(2x_i + 5) + 4(2 - y_i)$

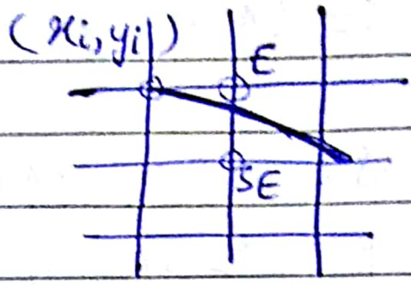
$$\boxed{\Delta^2 SE^{SE} = 8}$$

Code

```
x = 0;  y = R;
delta E = 6;
delta SE = 2(5 - 2R);
  while (x <= y)
{
```

$$b^2x^2 + a^2y^2 = a^2b^2$$





ion I

$$d_E = a^2y_i^2 - a^2y^2$$

$$d_{SE} = a^2y^2 - a^2(y_i - 1)^2$$

$$D^{I} = d_E - d_{SE}$$

$$= a^2\left[y_i^2 + (y_i - 1)^2 - 2y_1^2\right]$$

$$\left[a^2y^2 = a^2b^2 - b^2(x_i + 1)^2\right]$$

If $D < 0$,

selection is E:

x_i+1, y_i

$$D_{i+1} = a^2[y_i^2 + (y_i-1)^2 - 2y_2^2]$$

$$D_{i+1} - D_i = \cancel{a^2[-2}$$

$$-2a^2y_2^2 + 2a^2y_1^2$$

$$= 2[\cancel{a^2b^2} - b^2(x_i+1)^2 - \cancel{a^2b^2} + b^2(x_i+2)^2]$$

$$\Delta E = 2b^2[2x_i+3]$$

else

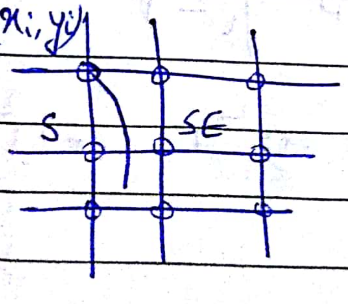
selection is SE

x_i+1, y_i-1

$$D_{i+1} = a^2[(y_i-1)^2 + (y_i-2)^2 - 2y^2]$$

$$\Delta SE = D_{i+1} - D_i = 2b^2(2x_i+3) + 4a^2(1-y_i)$$

Region II (x_i, y_i)



$$d_{S} = b^2x_i^2 - b^2x^2$$

$$d_{SE} = b^2x^2 - b^2(x_i+1)^2$$

$\boxed{D} = d_S - d_{SE}$

$= b^2[x_i^2 + (x_i+1)^2 - 2x_1^2]$

$\{b^2x^2 = a^2b^2 - a^2(y_i-1)^2\}$

If $D<0$,

selection is S,

$x_i, y_i - 1$

$D_{i+1} = b^2[x_i^2 + (x_i+1)^2 - 2x_2^2]$

$\Delta S = D_{i+1} - D_i$

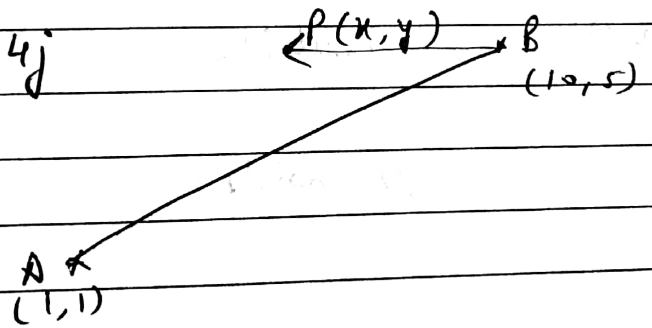
$= 2[b^2x_1^2 - b^2x_2^2]$

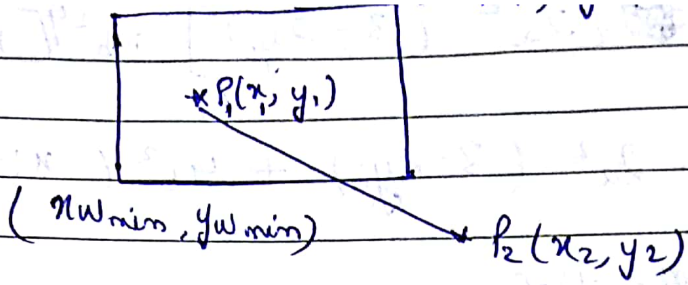
$$= [2a^2 (3 - 2yi)] + b^{24}[1 + xi]$$

$$= 2a^2 (3 - 2yi) + 4b^2 \cdot (xi + 1).$$

Second Differential

$\vec{AB} = 9i + 4j$





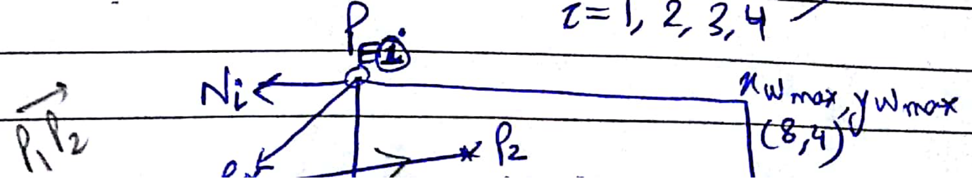
$$\left.\begin{array}{l} xw_{min} \le x \le xw_{max} \\ yw_{min} \le y \le yw_{max} \end{array}\right\} \text{cond}^n \text{ for pt inside}$$

```
int inside (Point P1)
{
        return 0;        // outside

        return 1;        // inside
}
```

P_1 & P_2 in 'inside' func return →
- 0 0 → totally outside (no change on screen)
- 0 1, 1 0 } clipping is to be performed
- 1 1 → totally inside (retain)

Cyrus Beck

$i = 1, 2, 3, 4$ → for all 4 edges



Scanned by CamScanner

$$P(t) = P_1 + (P_2 - P_1)t$$

$$0 \le t \le 1.$$

$$\vec{v} = P(t) - P_{Ei}$$

$$\vec{v} \cdot N_i \to \begin{cases} < 0 \\ = 0 \\ > 0 \end{cases}$$

<u>Intersection</u>

$$\vec{v} \cdot N_i = 0$$

$$(P(t) - P_{Ei}) \cdot N_i = 0.$$

$$[P_1 + (P_2 - P_1)t - P_{Ei}] \cdot N_i = 0$$

$$(P_1 - P_{Ei}) \cdot N_i + N_i \cdot (P_2 - P_1)t = 0$$

$$t = \frac{N \cdot (P_1 - P_{Ei})}{-N_i \cdot (P_2 - P_1)}$$

Left edge

Denominator

$N_i \cdot (P_2 - P_1) \neq 0$

$> 0 \longrightarrow$ potentially exiting

$< 0 \longrightarrow$ potentially entering

Interaction pt is now calculated using t,

$\left(0, \frac{7}{4}\right)$.

$$P(t) = P_1 + (P_2 - P_1)\,t$$
$$= -3i + j + (5i + 3j + 3i - j)\,\frac{3}{8}$$
$$= -3i + 3i + j + \frac{3}{4}j = 0i + \frac{7}{4}j$$

	Normal	$P_1 - P_{Ei}$	$N_i \cdot (P_1 - P_{Ei})$	$N_i \cdot (P_2 - P_1)$	t
Left	$N_i = -i$	$i + 3j$	[illegible]	-12	$3/12$
Bottom	$N_i = -j$	$i + 3j$	[illegible]	-9	$5/9$
Right	$N_i = +i$	$8i + 8j$	[illegible]	12	$10/12$
Top	$N_i = +j$	$8i + 8j$	[illegible]	9	$10/9$


(8,8)

			P_{Ei}		
$\vec{AC}$	$3i+9j$	$-9i+3j$	$i+j$		$\vec{t_{AC}} = \frac{12}{31}$
$\vec{CB}$	$3i-5j$	$5i+3j$	$4i+10j$		$\vec{t_{CB}} = 48/61$
$\vec{BA}$	$-6i-4j$	$4i-6j$	$7i+5j$		$\vec{t_{BA}} = 15/17$

✓ P_{Ent}

✓ P_{Ent}

✓ P_{exit}

X

line		normal	
+	+	−	+
+	−	+	+
−	+	−	−
−	−	+	−

(anticlockwise normals drawn)

$$(-9i+3j)\cdot(-2i+4j-i-j) = \frac{27+9}{}$$

classmate
Date
Page

$$\vec{t_{CB}} = \frac{(5i+3j) \cdot (-2i+4j - 4i - 10j)}{(-5i-3j) \cdot (11i+2j)}$$

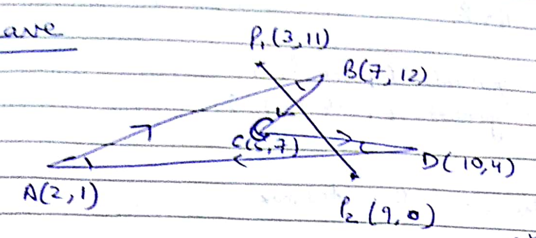
$$= \frac{-30 - 18}{-55 - 6} = \frac{48}{61}$$

$$\vec{t_{BA}} = \frac{(4i-6j) \cdot (-2i+4j - 7i - 5j)}{(-4i+6j) \cdot (11i+2j)}$$

$$= \frac{-36 + 6}{-44 + 12} = \frac{30}{32} = \frac{15}{16}$$

29/1/18

Concave



$3i+11j$ $\qquad$ $6i-11j$

t	Edge		Normal (N)	P_{Ei}	P_1-P_{Ei}	$N\cdot(P_1-P_{Ei})$	$N\cdot(P_2-P_1)$
39/121	$\vec{AB}$	$5i+11j$	$-11i+5j$	$2i+j$	$i+10j$	$-11+50=39$	$+121$
18/52	$\vec{BC}$	$-2j-5j$	$+5i-2j$	$7i+12j$	$-4i-j$	$-20+2=-18$	-52
14/37	$\vec{CD}$	$5i-3j$	$+3i+5j$	$5i+7j$	$-2i+4j$	$-6+20=14$	37
77/106	$\vec{DA}$	$-8i-3j$	$+3i-8j$	$10i+4j$	$-7i+7j$	$-21-56=-77$	-106.

0 $\quad P_{Ent}$ $\quad P_{exit}$ $\quad P_{ent}$ $\quad P_{exit}$ $\quad$ 1

$S: \left\{0, \frac{39}{121}, \frac{18}{52}, \frac{14}{37}, \frac{77}{106}, 1\right\}$

$S: \{0, P_{Ent1}, P_{Ent2}, P_{Exit1}, P_{Ex2}, 1\}$

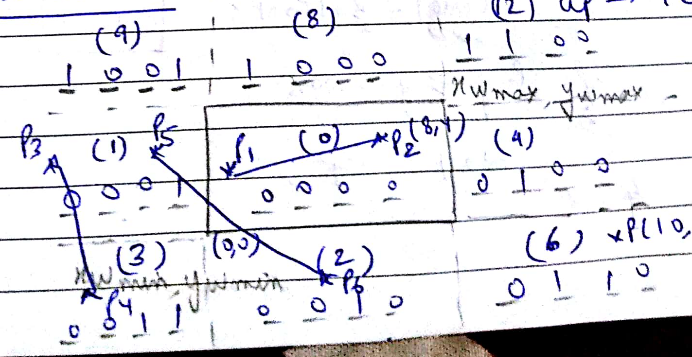
Take a pair of entering & exiting & draw a line

Wherever there is more than 1 entering pt, we check the signature.

bottom → 0010
left → 0001
right → 0100
up → 1000

Cohen - Sutherland

9 addresses for these regions



```
#define LEFT    0x1
#define BOTTOM 0X2
#define RIGHT 0x3
#define TOP  0X4

Code = 0x0

gencode (point P)
{
        if P.x ≤ xwmin
              code | = LEFT;
        if P.x ≥ xwmax
              code | = RIGHT;
        if P.y ≤ ywmin
              code | = BOTTOM;
        if P.y ≥ yw max
              code | = Top;
}
```

for P(10, -2)

0000

0100

$\boxed{0110}$

do { Method: code1 = gencode (...); code2 = gencode (...);

Case 1 (P_1 & P_2)

if (code 1 & code 2 = = 0) & & (code 1 | code 2 = = 0)

/* line displayed */

done = 1;

Case 2 (P_3 & P_4)

if (code1 & code 2 ! = 0)

/* line is not visible */

done = 1;

Case 3 (P_5 & P_6)

if (code 1 & code 2 = = 0) && (code 1 | code 2 ! = 0)

/* line clip ( ) */

} while (done ! = 1)

gencode (point P)

{

//left

code [3]

// if sign bit of $x - xw_{min}$ (difference is -ve) is 1

code [3] = 1 else 0.

//right

code [2]

// bottom

code [1]

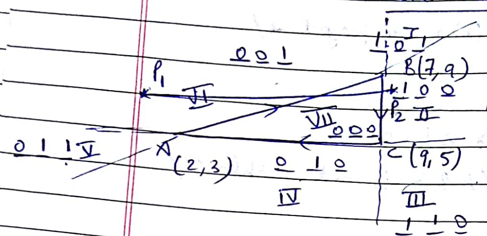
// Top

code [0] =

}

2/2/18

Line Clipping (Cohen Sutherland)



Non - rectangular window

7 regions ⟹ 3 bits for addressing

I, VI and V → 1 at ~~last~~ bit

V, IV and III → 1 at 2^{nd} last bit

Also, Note - and so on.

ORing of VI & II → I

step ① Allocating the region code for address.

code 1 = gencode (P_1)

code 2 = gencode (P_2) $\rightarrow y - y_1 = \left(\dfrac{y_2 - y_1}{x_2 - x_1}\right)(x - x_1)$

Edge AB

(2,3) (7,9)

$f(x,y) = 5y - 6x - 3 = 0$

C(9,5) < 0

Edge AC

(2,3) (9,5)

$f(x,y) = -7y + 2x + 17$

B(7,9) < 0

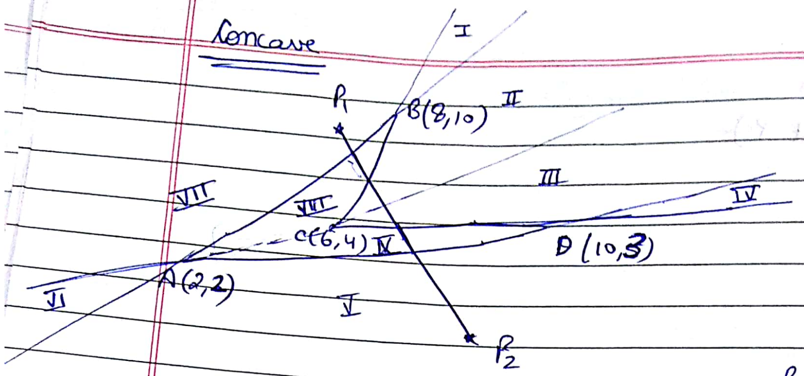
Edge BC

(7,9) (9,5)

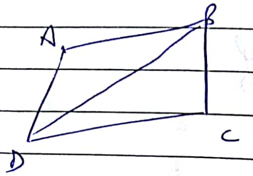
$f(x,y) = y + 2x - 23$ A(2,3) < 0

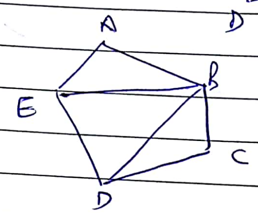
```
code = 0 * 0
gencode(point P)
{
        if

        if

        if

}.
            done = 0;
while {
```

Concave



A
B
C
D





Convex → take 3 points
contⁿ → join

Concave → find pts of concavity.
Starting from there take 3pts, join

Nicholl – Lee – Nicholl

9 regions rectangular window

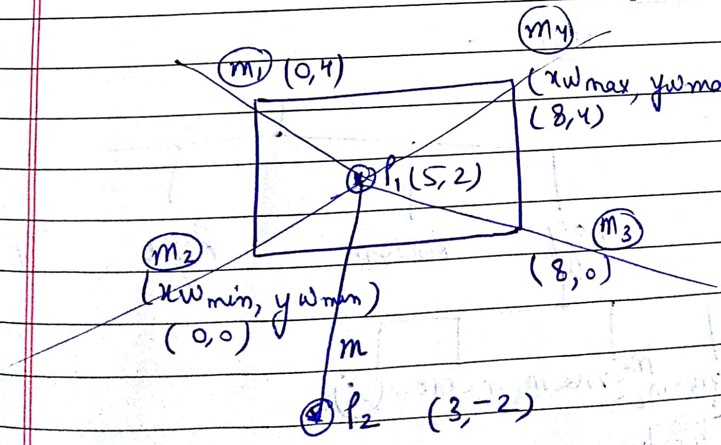
Cluster – I one of the end points of a line is at centre (or inside a rectangular window)

Cluster – II one of the end points is at the edge of rectangular window (Left, Bottom, Right, Top)

Cluster – III one of the end point of a line lies at a corner.

III	II	III
II	I	II
III	II	III

Cluster – I "Using slope as a concept for clipping"



$m = \frac{-4}{-2} = 2$

$m_1 = \frac{2}{-5}$

$m_2 = \frac{2}{5}$

$m_3 = \frac{2}{-3}$

$m_4 = \frac{2}{3}$

$m_1 \le m < m_2$ $m_2 \le m < m_3$ $m_3 \le m < m_4$ $m_4 \le m < m_1$

$P_2 . y_2 < yw_{min}$ Else

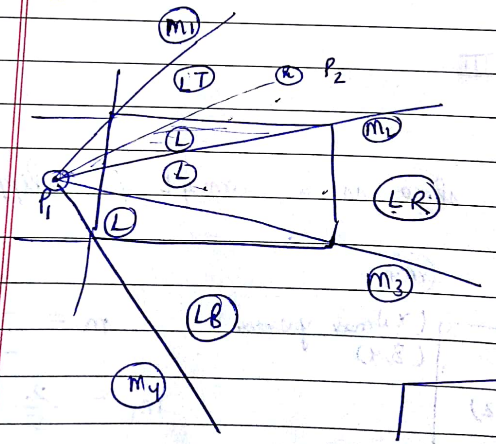
clip   no clipping

if $((m < m_3)$ && $(m > m_2)$ && $(P_1 . y_1 > P_2 . y_2))$

if $(P_2 . y_2 < yw_{min})$

→ clipping

Cluster - II

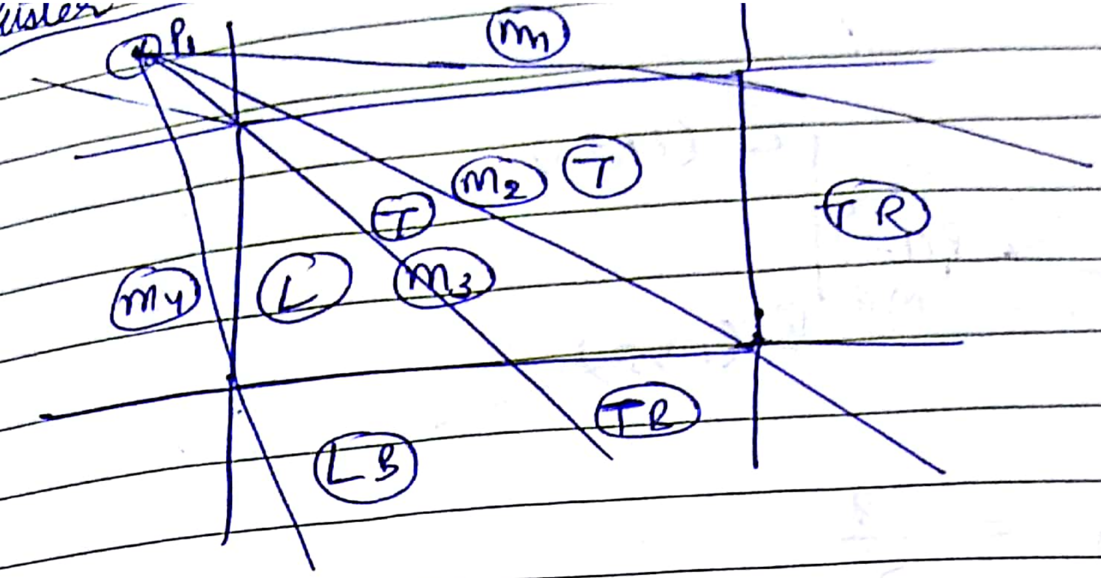


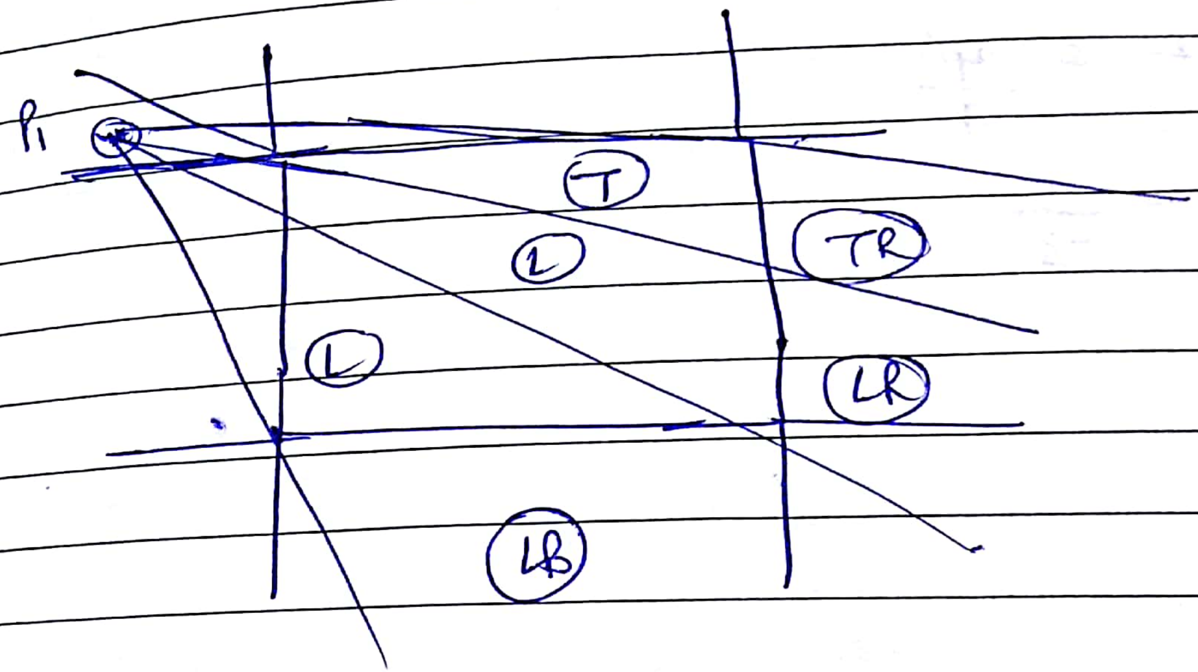
Left   Bottom   Right   Top

$m_1 \le m < m_2$ $m_2 \le m < m_3$ $m_3 \le m < m_4$ X

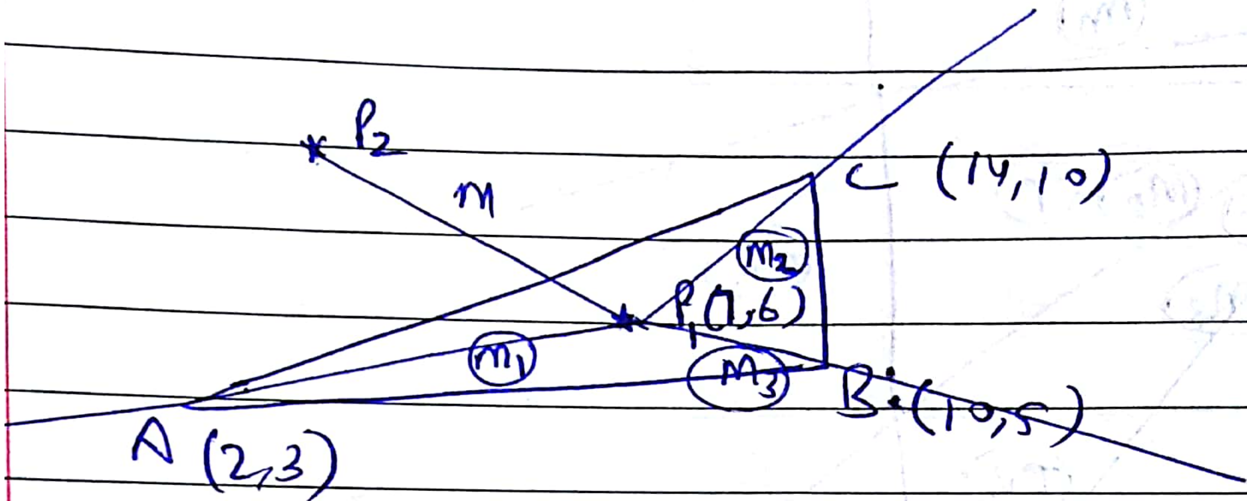
L   LT X

Cluster





Non Rectangular Window



$$m_1 = \frac{3}{5}$$

$$m_2 = \frac{4}{7}$$

$$m_3 = \frac{1}{-3}$$

5/1/18

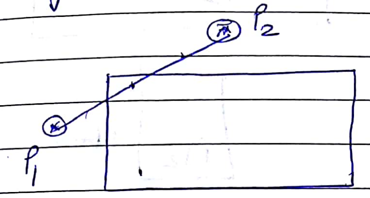
Recursion

```
fact (int n):
{
    if (n==0) || (n==1)
        return (1);
    else
        return (n * fact(n-1));
}
```

Rectangular window

~~(Multiple so~~

(Midpoint subdivision line clipping algorithm)



Condition

(a) Code 1 & Code 2 both = 0

→ line visible

(b) Code 1 & Code 2 != 0

→ line invisible

(c) Code 1 & code2 == 0 && code 1 | code 2 != 0

recursive function

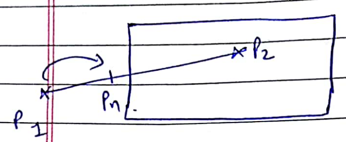
Breaking the line

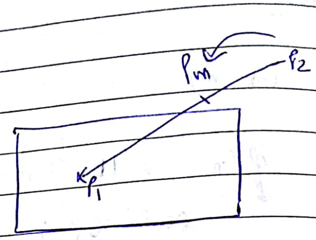
$$P_m = (P_1 + P_2)/2$$

→ Consider $P_1 P_m$

→ Consider $P_m P_2$

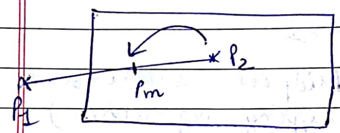
Scenario I

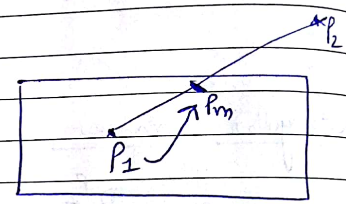


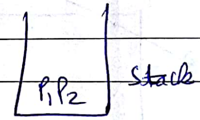


Scenario II

$P_m = = 0$







Algorithm

input : P_1, P_m, xw_{min}, yw_{min}, xw_{max}, yw_{max}

Method :

step ① : Code 1 = gencode (P_1);
Code 2 = gencode (P_2);
if code 1, code 2 both = 0
draw $P_1 P_2$;

Step ② : if code 1 & code 2 <> 0
/* ignore line */

Step ③ : $P_m = (P_1 + P_2) / 2$;
Code m = gencode (P_m);

Step ④: if code m <> 0
then
    if code 1 & code m <> 0
        then
            $P_1 \leftarrow P_m$; goto step ①
    else if
        code 2 & code m <> 0
            then
                $P_2 \leftarrow P_m$, goto step ①

Step ⑤: if code m = = 0
    if code 1 , code m both = 0
        then Consider $P_m P_2$
    else if code 2, code m both = 0
        then Consider $P_1 P_m$

Step ⑥: Consider $P_1 P_m$

do { $P_{m1} = (P_1 + P_m)/2$;
    Code m1 = gencode (P_{m1});
    if Code m1 <> 0 then $P_1 \leftarrow P_{m1}$
    else $P_m \leftarrow P_{m1}$
}

while ($P_1.x$ <> x_{wmin} && $P_1.x$ <> x_{wmax}
    && $P_1.y$ <> y_{min} && $P_1.y$ <> y_{wmax});

$P_1 \leftarrow P_m$

Step ⑦: Consider $P_m P_2$

do {

$P_{m_2} = (P_2 + P_m)/2$; code m_2 = gencode (P_{m_2})

if code m2 <> 0 then $P_2 \leftarrow P_{m_2}$

else $P_m \leftarrow P_{m_2}$

}

while ($P_2.x$ <> xwmin && $P_2.x$ <> xwmax && $P_2.y$ <> ywmin && $P_2.y$ <> ywmax)

$P_2 \leftarrow P_m$

Example

$P_1(120, 5)$

P_2 (~~###~~ 180, 30)

xwmin = 100 xwmax = 160

ywmin = 10 ; ywmax = 40

P_1	code 1	P_2	Code 2	P_m	Code m	Remarks
(120,5)	0010	(180,30)	0100	(150,18)	0000	Consider $P_1 P_m$ Consider $P_m P_2$
Consider $P_1 P_m$						
(120,5)	0010	(150,18)	0000	(135,12)	0000	draw (150,18) & (135,12)
(120,5)	0010	(135,12)	0000	(128,9)	0010	nothing to do
(128,9)	0010	(135,12)	0000	(132,11)	0000	draw (135,12) to (132,11)
(128,9)	0010	(132,11)	0000	(130,10)	0000	$P_1 \leftarrow$ 3?0
		(130,10)	draw (132,11) to (130,10)			

$\boxed{\text{consider } P_mP_2}$

(150,18) 0000 (180,30) 0100 (165,24) 0100

(150,18) 0000 (165,24) 0100 (158,21) 0000

(158,21) 0000 (165,24) 0100 (162,23) 0100

(158,21) 0000 (162,23) 0100 (160,22) 0000

Lian Baski line clipping → Do it yourself.

Liang Barsky

$x = x_1 + t(x_2 - x_1)$
$y = y_1 + t(y_2 - y_1)$

$xw_{min} \le x_1 + t\Delta x \le xw_{max}$
$yw_{min} \le y_1 + t\Delta y \le yw_{max}$

$xw_{min} \le x_1 + t\Delta x \Rightarrow -t\Delta x \ge xw_{min} - x_1$ (p_k, q_k)

$-t\Delta x \le x_1 - xw_{min}$
$t\Delta x \le xw_{max} - x_1$

$p_1 = -\Delta x$ $q_1 = x_1 - xw_{min}$
$p_2 = \Delta x$ $q_2 =$
$p_3 = -\Delta y$ q_3
$p_4 = \Delta y$ q_4

Algo

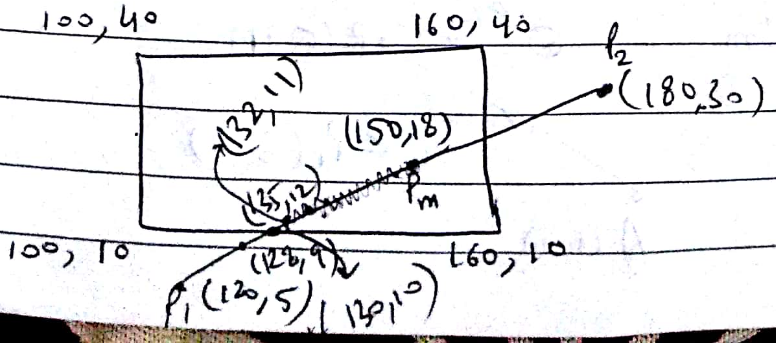
$p_k = 0$ $q_k < 0$ reject.

$p_k < 0$ $q_1/p_1 = t$

$p_k > 0$

t_1, t_2

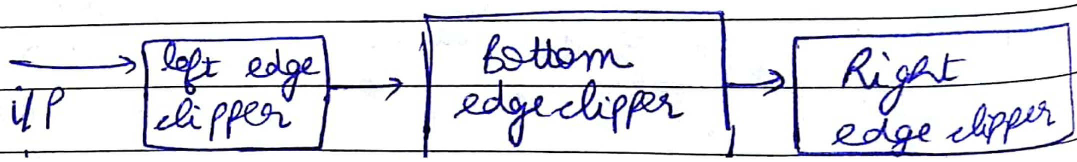
$\{ 1, t, t \}$



Output

Clipped polygon (list of vertices)

(6 vertices)



Vector concept

$\vec{AB} = 4i + 10j$

$N_{\vec{AB}} = -10i + 4j$

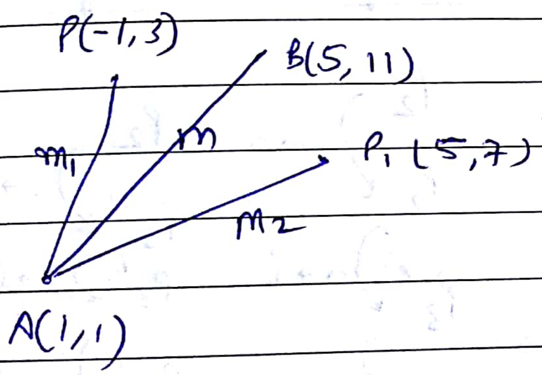
$[P - P_E(A)] = -2i + 2j$

Dot product $\vec{N_{AB}} \cdot \overrightarrow{[P - P_E A]} = 20 + 8$
$= 28 > 0$

for P_1,

$[P_1 - P_E(A)] = 4i + 6j$

Dot product $= -40 + 24$
< 0.



for (clipping window)

for (subject polygon)

<u>i/p vertices list</u> <u>Left Edge</u>

i/p vertices list	Left Edge		
$P_1 \to P_2$	$0 \to i$	intersection!	P_1', P_2
$P_2 \to P_3$	$i \to i$		P_3
$P_3 \to P_4$	$i \to i$		P_4
$P_4 \to P_1$	$i \to 0$	intersection!	P_4', P_1'

$$x_I = x_{wmin}$$

$$y_I = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x_1 - x_{wmin})$$

Rule	$P_i \to P_j$
$i \to o$	Intersection (I)
$o \to i$	① Intersection point (I) ② second Vertex (P_j)
$i \to i$	Second vertex (P_j)
$o \to o$	— Nil —

i/p

$P_1 \xrightarrow{o \to i} P_2 \xrightarrow{i \to i} P_3 \xrightarrow{i \to i} P_4 \xrightarrow{i \to o} P_1$

left-edge

$P_1' \xrightarrow{i \to i} P_2 \xrightarrow{i \to i} P_3 \xrightarrow{i \to o} P_4 \xrightarrow{o \to i} P_4' \xrightarrow{i \to i} P_1'$

bottom edge

$P_2 \xrightarrow{i \to o} P_3$

$P_3' \xrightarrow{o \to i} P_4'' \xrightarrow{i \to i} P_4' \xrightarrow{i \to i} P_1' \xrightarrow{i \to i} P_2$

left edge

$P_2' \xrightarrow{i \to i} P_3''$

$P_3' \xrightarrow{i \to i} P_4'' \xrightarrow{i \to i} P_4' \xrightarrow{i \to i} P_1' \xrightarrow{i \to i} P_2 \xrightarrow{i \to o} P_2' \xrightarrow{o \to i}$

top edge

P_3''

P_3'

P_4''

P_4'

P_1'

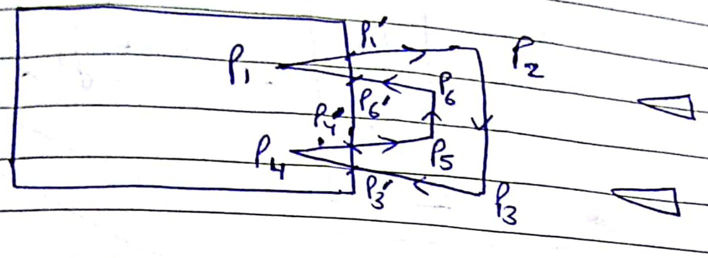
P_1''

P_2''

P_2'

P_3''

Case ②



	i/p	Right edge
i→0	$P_1 \to P_2$	P_1'
0→0	$P_2 \to P_3$	—
0→i	$P_3 \to P_4$	P_3', P_4
i→0	$P_4 \to P_5$	P_4'
0→0	$P_5 \to P_6$	—
0→i	$P_6 \to P_1$	P_6', P_1
		P_1'

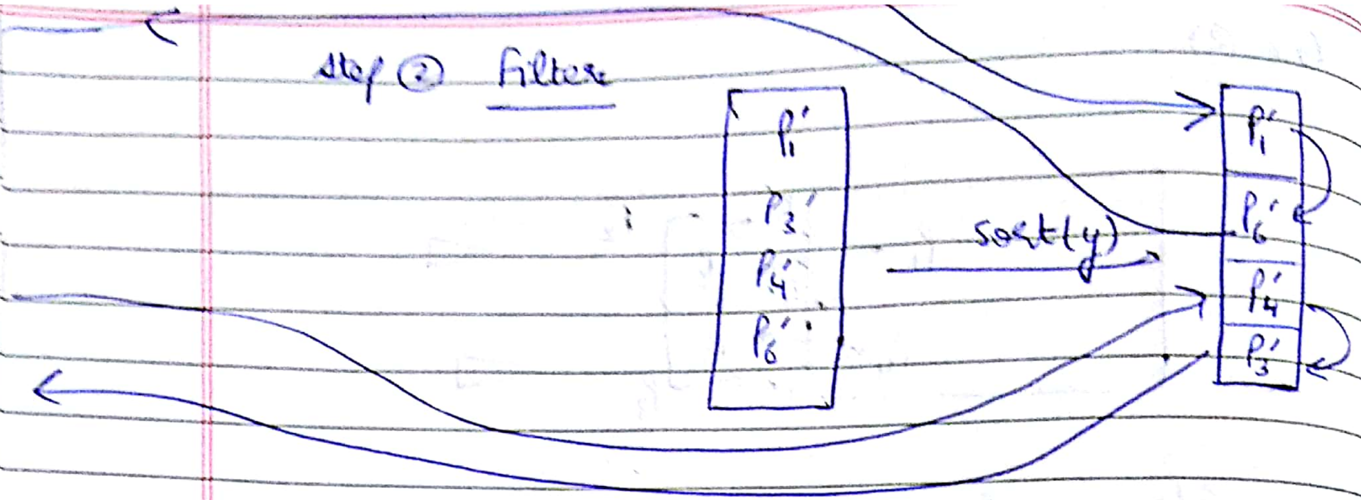
Step ①

bit = 1 when $i \to 0$

	Bit	Traversal
P_1'	1	~~0~~ 1
P_3'	0	1 ~~0~~
P_4	0	~~0~~ 1
P_4'	1	~~0~~ 1
P_6'	0	~~0~~ 1
P_1	0	1 ~~0~~
P_1'	0	~~0~~ 1

1 signifies → lying on the right edge!

Step ② Filter

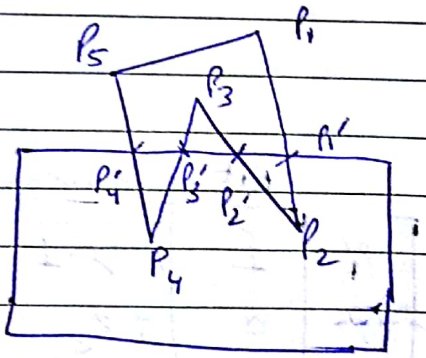


line (P_1', P_6')
line (P_6', P_1)
line (P_1, P_1')
line (P_3', P_4)
line (P_4, P_4')
line (P_4', P_3')

whenever bit is 1, go to sorted (y) table.

Stop when traversal bits are all 1.

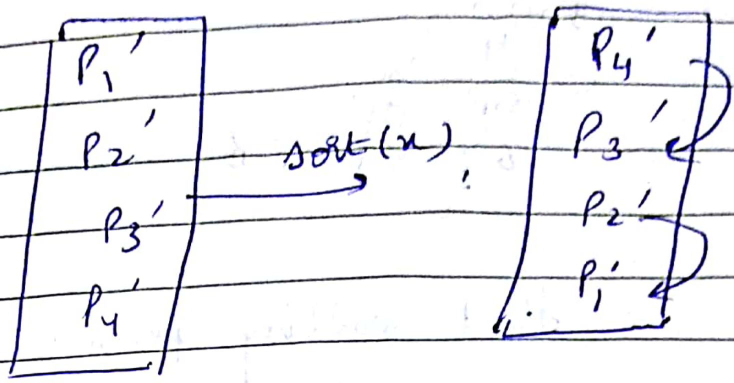
Case ③



Step①

	i/P	Top Edge		Traversal
0 → i	P_1	P_1'	0	~~0~~ 1
	P_2	P_2	0	~~0~~ 1
i → 0		P_2'	1	~~0~~ 1
0 → i	P_3	P_3'	0	~~0~~ 1
	P_4	P_4	0	~~0~~ 1
i → 0	P_5	P_4'	1	~~0~~ 1
0 → 0	P_1	P_1'	0	~~0~~ 1

Step 2 filter



line ($P_1' \to P_2$)

line ($P_2 \to P_2'$)

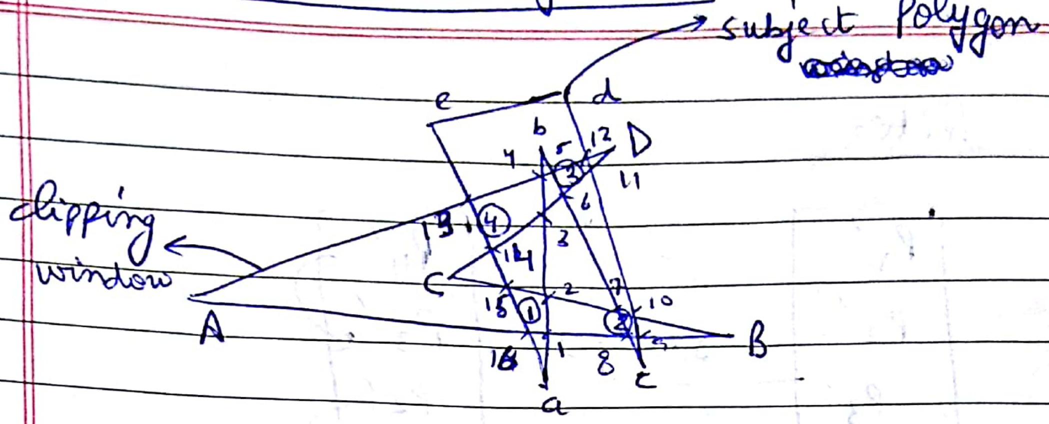
line ($P_2' \to P_1'$)

line ($P_3' \to P_4'$)

line ($P_4 \to P_4'$)

line ($P_4' \to P_3'$)

Weiler Atherton Algorithm



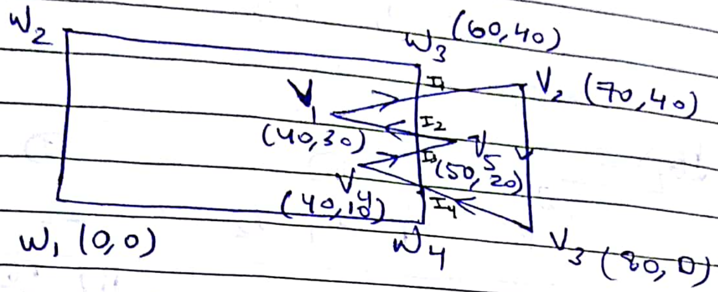
Start walking from a, when we reach intersection pt ① [o → i], we illuminate till ① → ②. When ② is reached, we take a turn & swap roles. [i → o]
The subject polygon now becomes the clipping window & vice-versa.

SP	a	b	c	d	e
CW	A	B	C	D	

12/2/18

Polygon clipping

Weiler Atherton



ⓐ Exiting (Switching the traversal from subject polygon to clipping window)
ⓑ Entry

Edge → $W_3 W_4$ (Clipping window)
Cyrus Beck

Entry/Exit	Subject Polygons	N	Numerator	Denominator	t
0 I_1	$V_1 V_2$ $(30i + 10j)$	$-10i + 30j$	$-200+300$		$1/12$
—	$V_2 V_3$ $(10i - 40j)$	$40i + 10j$			-1
1 I_4	$V_3 V_4$ $(-40i + 10j)$	$-10i - 40j$			$7/8$
0 I_3	$V_4 V_5$ $(10i + 10j)$	$-10i + 10j$			$1/4$
1 I_2	$V_5 V_1$ $(-10i + 10j)$	$-10i - 10j$			$3/4$

$$t = \frac{N \cdot (W_3 - V_{Ei})}{-N \cdot (W_4 - W_3)}$$

$t : (1/12, 7/8, 1/4, 3/4)$

↓ Sort

$(1/12, 1/4, 3/4, 7/8)$

Step-1

SP: V_1 V_2 V_3 V_4 V_5

CW: ω_1 ω_2 ω_3 ω_4

Step-2

SP: V_1 $\overset{\boxed{0}}{I_1}$ V_2 V_3 $\overset{\boxed{1}}{I_4}$ V_4 $\overset{\boxed{10}}{I_3}$ V_5 I_2

CW: ω_1 ω_2 ω_3 $\overset{\boxed{11}}{I_1}$ $\overset{\boxed{10}}{I_2}$ $\overset{\boxed{11}}{I_3}$ $\overset{\boxed{10}}{I_4}$ ω_4

Step-3

Entry list: $\{I_4, I_2\}$

Step-4

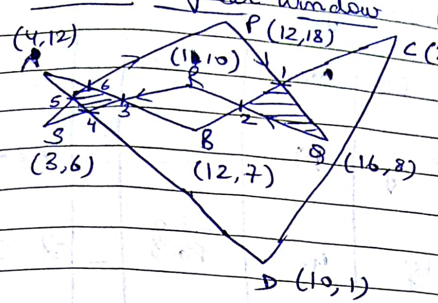
Tagging intersection ~~pts~~ with 0 & 1 from entry list.

line (I_4, V_4)
line (V_4, I_3)
line $(I_3\ I_4)$

line (I_2, V_1)
line (V_1, I_1)
line (I_1, I_2)

16/2/18

Non Rectangular Window (Weiler Atherton)



ABCD + (Non rect. window)

PQRS – subject polygon

Step ①

CW → A B C D

SP – P Q R S

Method {

Output 1 Q 2

3 4 5 6

Step ②

Intersection point is evolved

PQ – $4i - 10j$

Edge	Normal	$P - E_i$	Num	Den		t	
AB $8i - 5j$	$5i + 8j$	$8i + 6j$	88	60	–	1.46	×
BC $11j + 8j$	$-8i + 11j$	$11j$	121	142		0.85	✓ ×
CD $-13i - 14j$	$14i - 13j$	$-11i + 3j$	-193	-186	–	1.037	
DA $-6i + 11j$	$-11i - 6j$	$2i + 17j$	-124	-16	–	7.75	×

DA $-6i+11j$	$-11i-6j$	$i+9j$	-65	-112

Step 2 SP

	P	1	Q	2	R	3	4	S	5	6
Tag	-1	0	-1	1	-1	0	1	-1	0	1
Traversal Bit	0	~~0~~ 1	~~0~~ 1	~~0~~ 1	0	~~0~~ 1	~~0~~ 1	0	~~0~~ 1	~~0~~ 1

$SP \rightarrow 9i + 12j$

Edge	Normal	S − Gi	Num	Den		t
AB $8i-5j$	$5i+8j$	$-i-6j$ ~~$7i-2j$~~	-53	-141	1	$\frac{53}{141}$
BC $11i+8j$	$-8i+11j$	$-9i-j$ ~~$-1i-3j$~~	61	-60	−	$\frac{-61}{60}$ ✗
CD $-13i-14j$	$14i-13j$	$-20i-9j$ ~~$-12i-5j$~~	163	-30	−	$\frac{163}{-30}$ ✗
DA $-6i+11j$	$-11i-6j$	$-7i+5j$ ~~$i+9j$~~	47	171	0	$\frac{47}{171}$

for $P \to Q \to R \to S$

for $A \to B \to C \to D \to A$

updated sp list

OR

for $A \to B \to C \to D$

for $P \to Q \to R \to S \to P$

CW

	A	6	3	B	2	1	C	D	4	5
	-1	0	1	-1	0	1	-1	-1	0	1
Traversal bit	0	~~0~~ 1	~~0~~ 1	0	1 ~~0~~	1 ~~0~~	0	0	~~0~~ 1	~~0~~ 1

line (1,0)
line (0,2)
line (2,1)

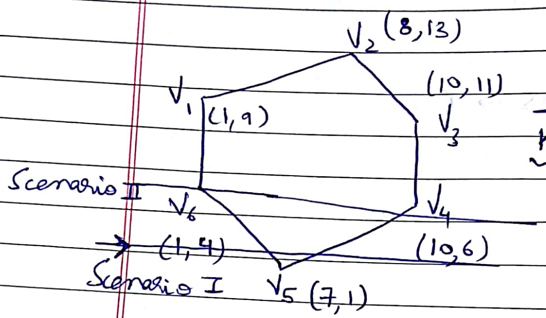
line (3,4)
line (4,5)
line (5,6)
line (6,3)

* [Do ~~for~~ for QR to find pt 2 ± value]

Polygon Filling

Approach - I

Scanline → Horizontal
→ Vertical
→ or in any angle

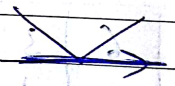


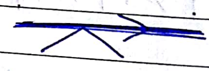
Cases:

non-Horizontal Edges

Scenario II → Vertices

$$\begin{cases} y_{min}\, E_1 = y_{min}\, E_2 \\ \text{or} \\ y_{max}\, E_1 = y_{max}\, E_2 \end{cases}$$





↓

Vertices should be taken as 2 edges here as we need to start as well as stop the illumination to get a single point

$y = mx + c$

$y_i + 1 = m(x_{i+1}) + c$

$$\begin{cases} y_{i+1} \longrightarrow y_i + 1 \\ x_{i+1} \longrightarrow x_i \pm \frac{1}{m} \end{cases}$$

$m x_i + \cancel{c} + 1 = m(x_{i+1}) + \cancel{c}$

$x_{i+1} = x_i + 1/m$

Global Edge Table

[Horizontal scanning from left to right, bottom to top]

Edge

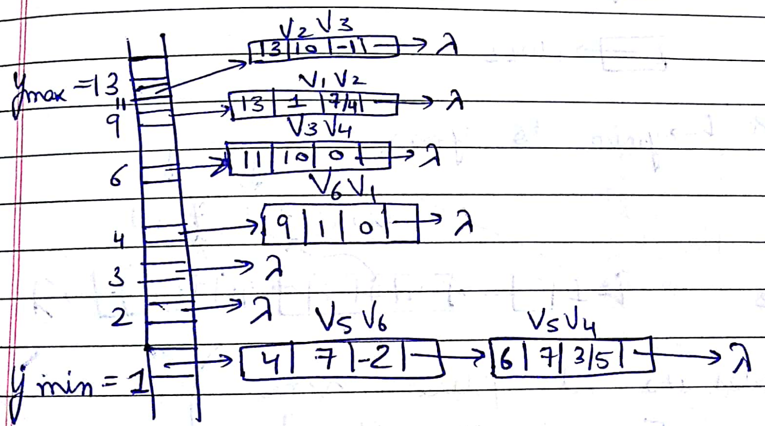
y_{max}	$x_{y_{min}}$	$1/m$	→

Example

$V_5 V_6$

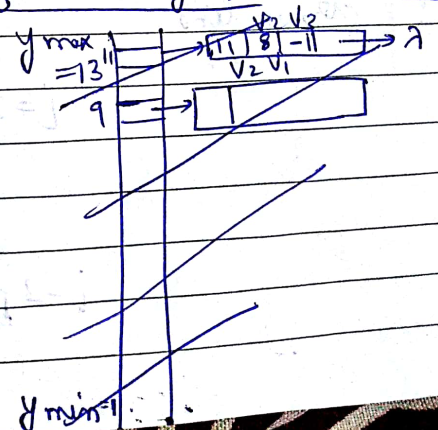
4	7	-2	→

Table → array of linked lists



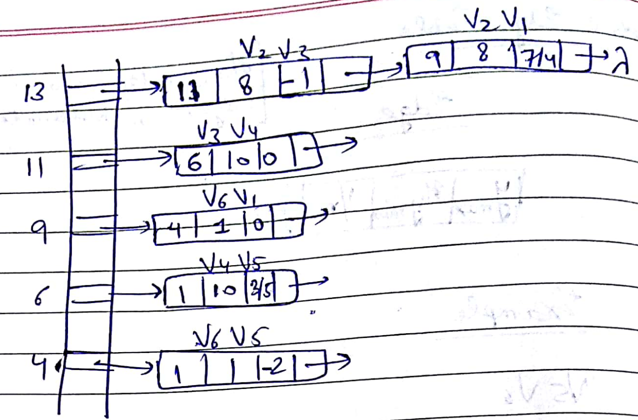
for Horizontal scanning left to right, top to bottom

Global Edge Table ←



y_{min}	$x_{y_{max}}$	$1/m$	→

$$\left.\begin{array}{l} y_{(i+1)} \to y_i - 1 \\ x_{i+1} \to x_i \pm 1/m \end{array}\right]$$



Active Edge Table

(AET)

[For filling the polygon]

☐ → NULL

for i → ymin to ymax

line(7, 1 , 7, 1)

Rules

1. Merge   i=1 ☐ → | 4 | 7 | -2 | → | 6 | 7 | 3/5 | → λ

2. If nodes are old update them

$[x_{i+1} = x_i \pm 1/m]$

3. draw line.

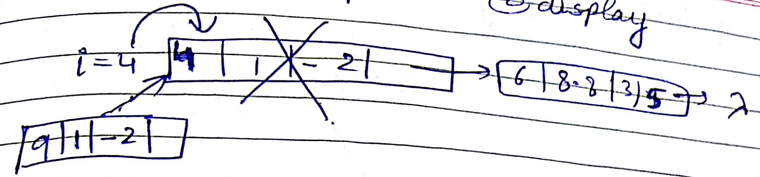
i=2 ☐ → | 4 | 5 | -2 | → | 6 | 7.6 | 3/5 |

line (5, i, 7.6, i)

i=3 ☐ → | 4 | 3 | -2 | → | 6 | 8.2 | 3/5 |

line (3, i, 8.2, i)

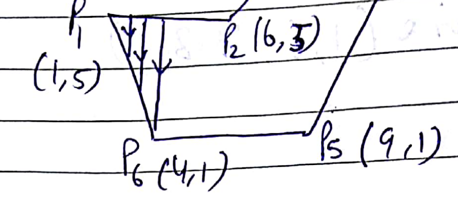
ⓐ merge ⓑ update ⓒ display



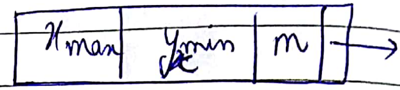
line (1, i, 8.8, i)

Vertical Scanning

(Scan line, Polygon Filling)



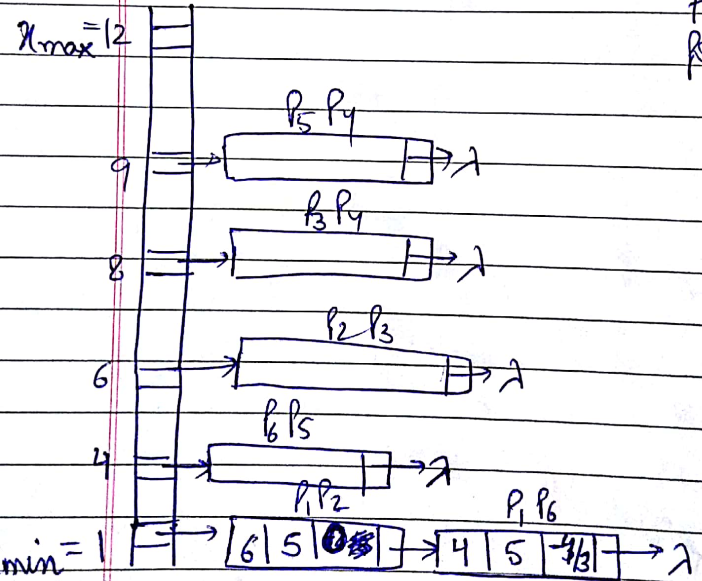
A Edge Structure $\Longrightarrow$



B Global Edge Table

Scanning top - bottom

Vertical Scanning (start from the left side of polygon)



Scanned by CamScanner

$\boxed{c}$ Active Edge

1) Merge
2) update (if nodes are the previous one)

3) Delete (if node (first member element) $= i$)

4) Draw line.

for $i = x_{min}$ to x_{max} {

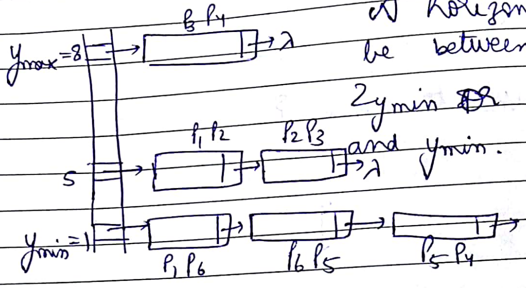
}

$i = 1$

Horizontal scanning

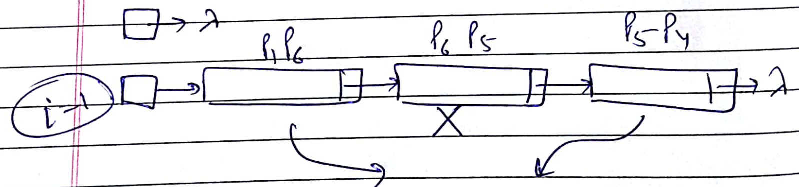
(We have P_3P_4 as an horizontal edge here)

y_{max}	$x_{y_{min}}$	$1/m$	

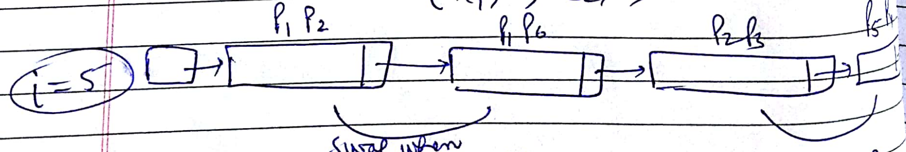


A horizontal edge can be between 2 y_{max}, 2 y_{min} b/w a y_{max} and y_{min}.

If a horizontal edge in the global edge table is at even posⁿ, simply delete it before drawing the line.



(x_1, i, x_2, i)



swap when horizontal edge comes

{ Checking for horizontal edge & its posⁿ

{ If odd posⁿ, swap rules ④ & ③. draw first, then delete