Definitions

Academic

Discipline which involves drawing / displaying graphical objects on graphic devices like Video Display unit OR a computer or workstation.

- graphical objects gen. using lines / points / curves / surfaces / solids & displayed on VDUs / workstations.

Commercial:

- Tools used to make pictures or objects.
- Can be both hardware / software.

H/W tools

↳ video monitor, graphics card, printer, mouse, pen, trackball, joystick.

S/W tools

↳ Prog. langs. have a collection of graphic entities that produce pictures OR dedicated languages for graphics.

- A POINT in CG is the smallest displayable object or a pixel.

- OpenGL - open source graphics libraries.
  - Device independent API.

Libraries in open GL -

① Basic GL ② GLU (Utility lib).

③ GLUT (Utility toolkit) ④ GLUI (User interface).

- Basic GL provides f^{ns} which are a permanent part of open GL.

Every f^n starts with GL prefix.

- GLU → manages WINDOWS, EVENTS, FULL SCREEN RENDERING.

- GLUT → manages high level processing reqd. for drawing of quadratic surfaces.

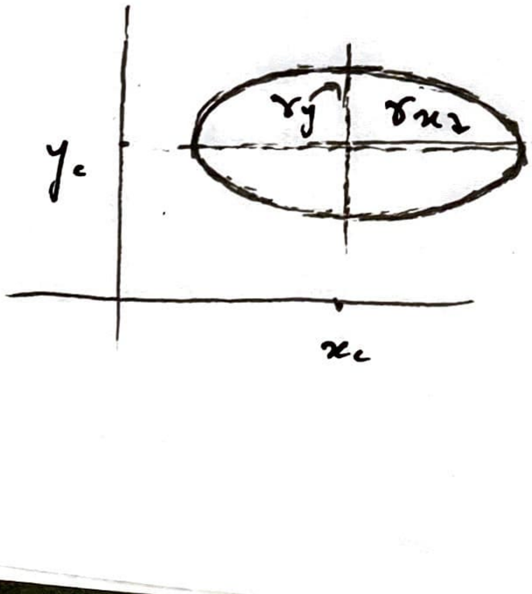
→ These involve matrix ops.

→ GLUI → provides sophisticated controls to open GL.

→

General Ellipse Eq^n

$$\frac{(x - x_c)^2}{r_x^2} + \frac{(y - y_c)^2}{r_y^2} = 1$$

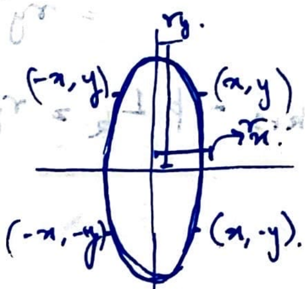


→ when you plot for ellipse centered at x_c, y_c, you shift $x, y \to (x + x_c, y + y_c)$

→ That's it!

Raster Scan Ellipse:

① Mid Point Algo.

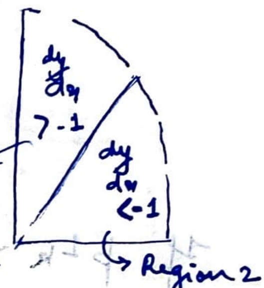


equation $\frac{x^2}{r_x^2} + \frac{y^2}{r_y^2} = 1$

symmetrical in quadrants.

$$\boxed{f_{ellipse}(x, y) = x^2 r_y^2 + r_x^2 y^2 - r_x^2 r_y^2}$$

$$\boxed{\frac{dy}{dx} = -\frac{r_y^2 x}{r_x^2 y}}$$



We need to move DOWN.

Region 1

$(x_k, y_k) \longrightarrow x_{k+1}, y_k$

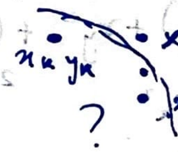
OR

x_{k+1}, y_{k-1}

$x_{mid} \to \frac{x_{k+1} + x_{k+1}}{2}$

$y_{mid} \to \frac{y_k + y_{k-1}}{2}$

$\Rightarrow (y_k - \frac{1}{2})$.



Decision parameter in region 1 -

$$P1_k = f_{ellipse}(x_{mid}, y_{mid})$$

$$= f_{ellipse}(x_{k+1}, y_k - \tfrac{1}{2})$$

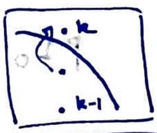
$$= r_y^2 \times (x_{k+1})^2 + r_x^2 (y_k - \tfrac{1}{2})^2 - r_x^2 r_y^2$$

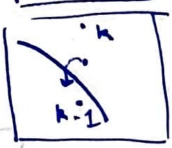
Now, if mid point inside ellipse, plot y_k

else plot y_{k-1}.

$\Rightarrow p1_k < 0$, plot x_{k+1}, y_k

else, plot x_{k+1}, y_{k-1}





Now, $p1_{k+1} = r_y^2(x_{k+1}+1)^2 + r_x^2(y_{k+1}-\frac{1}{2})^2 - r_x^2 r_y^2$

$$= r_y^2(x_k+1+1)^2 + r_x^2(y_{k+1}-\tfrac{1}{2})^2 - r_x^2 r_y^2$$

$$p1_{k+1} - p1_k = r_y^2(x_k+1+1)^2 + r_x^2(y_{k+1}-\tfrac{1}{2})^2 - \cancel{r_x^2 r_y^2} - r_y^2(x_k+1)^2 - r_x^2(y_k-\tfrac{1}{2})^2 + \cancel{r_x^2 r_y^2}$$

$$\Rightarrow p1_{k+1} - p1_k = r_y^2[\cancel{x_k}+\cancel{2}-\cancel{x_k}-1][2x_k+3] + r_x^2(y_{k+1}-y_k)(y_{k+1}+y_k-1)$$

$$= 2r_y^2\cdot[x_k+1] + r_y^2 \cancel{+ r_x^2[1]\cdot 2y_k} + (y_{k+1}-y_k)\cdot(y_{k+1}+y_k-1)$$

If $p1_k > 0$, means mid point outside,

$$\boxed{y_{k+1} = y_k - 1}$$

$$\Rightarrow p1_{k+1} - p1_k = 2r_y^2(x_k+1) + r_y^2 + (-1)r_x^2\, 2(y_k-1).$$

$$\Rightarrow \Delta p = 2r_y^2 x_k + 3r_y^2 - 2y_k r_x^2 + 2r_x^2$$

$$\boxed{\Delta p = 2r_y^2 x_{k+1} + r_y^2 - 2r_x^2(y_k-1)}$$

If $p1_k < 0$, means mid point inside,

$$\boxed{y_{k+1} = y_k}$$

$$\boxed{\Delta p = 2r_y^2 x_{k+1} + r_y^2}$$

At STARTING POINT →  $(0, r_y)$

$$p1_0 = f_{ellipse}(x_k+1, y_k-\tfrac{1}{2}) = (1, r_y-\tfrac{1}{2}).$$

$$= r_y^2 + r_x^2(r_y-\tfrac{1}{2})^2 - r_x^2 r_y^2$$

$$= r_y^2 + \cancel{r_x^2 r_y^2} + \frac{r_x^2}{4} - r_y r_x^2 - \cancel{r_x^2 r_y^2}$$

$$\boxed{p1_0 = r_y^2 + \frac{r_x^2}{4} - r_y r_x^2}$$

$\boxed{\text{Region 2}}$ → Start point where $\boxed{\frac{dy}{dx} = -1.}$

$\boxed{\text{UNIT STEPS in Y}}$.

↓ Here we are moving ↓, so next cell has $y \to y_k - 1$

→ Choosing x here!

x_k, y_k

?

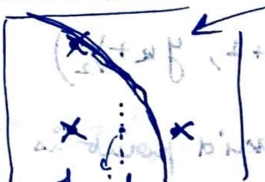
x_k, y_{k-1} $\quad$ x_{k+1}, y_{k-1}

mid point $= \left(x_k + \frac{1}{2}, y_k - 1\right)$

Again same treatment,

$$p2_k = r_y^2 \left(x_k + \tfrac{1}{2}\right)^2 + r_x^2 (y_k - 1)^2 - r_x^2 r_y^2$$

MID is INSIDE



Inside, Draw x_{k+1}.

$p2_k < 0$,

$\boxed{x_{k+1} = x_k + 1}$

MID is OUTSIDE

Outside, Draw x_k.

$p2_k > 0$,

$\boxed{x_{k+1} = x_k}$

$y_{k+1} \to (y_k - 1)$ → next iter value

$$p2_{k+1} = r_y^2 \left(x_{k+1} + \tfrac{1}{2}\right)^2 + r_x^2 \left((y_k - 1) - 1\right)^2 - r_x^2 r_y^2$$

$p2_k \leq 0$

$\boxed{x_{k+1} = x_k + 1}$

$$p2_{k+1} = r_y^2 \left(x_k + 1 + \tfrac{1}{2}\right)^2 + r_x^2 [y_k - 2]^2 - r_x^2 r_y^2$$

solve Δp in this case

$p2_k > 0$,

$\boxed{x_{k+1} = x_k}$

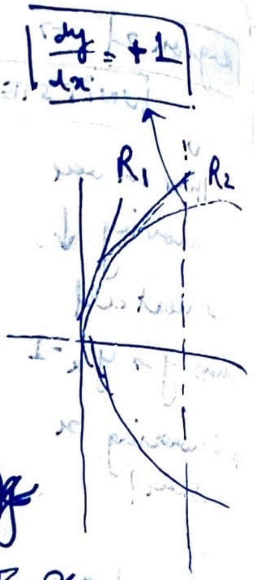
$$p2_{k+1} = r_y^2 \left(x_k + \tfrac{1}{2}\right)^2 + r_x^2 (y_k - 2)^2 - r_x^2 r_y^2.$$

solve Δp in this case.

$$p2(0) = f_{ellipse}\left(x_0 + \tfrac{1}{2}, y_0 - 1\right)$$

RASTER SCAN for Hyperbola

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$$

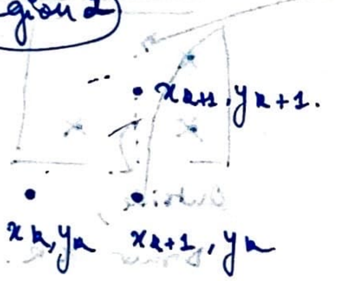


$$f_{hyperbola} = b^2x^2 - a^2y^2 - a^2b^2, \quad a > b$$

In Region 1, $\frac{dy}{dx} > 1$, unit step in y

In Region 2, $\frac{dy}{dx} < 1$, unit step in x

Region 2



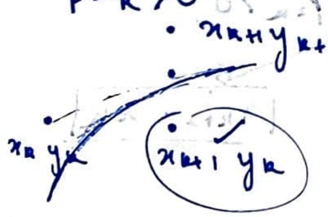
Mid point $= (x_{k+1}, y_{k+1/2})$

Again, if mid point is lying inside, then choose y_{k+1}, else choose y_k.

$$p2_k = b^2 \cdot (x_{k+1})^2 - a^2(y_k + \tfrac{1}{2})^2 - a^2b^2$$

$$= b^2(x_k + 1)^2 - a^2(y_k + \tfrac{1}{2})^2 - a^2b^2$$

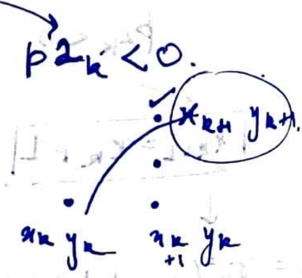
$p2_k > 0$



$$p2_{k+1} = b^2(x_{k+1}+1)^2 - a^2(y_{k+1} + \tfrac{1}{2})^2 - a^2b^2$$

$$= b^2(x_k+2)^2 - a^2(y_k + \tfrac{1}{2})^2 - a^2b^2$$

$p2_k < 0$



$$p2_{k+1} = b^2(x_{k+1}+1)^2 - a^2(y_{k+1} + \tfrac{1}{2})^2 - a^2b^2$$

$$= b^2(x_k+2)^2 - a^2(y_k + 1 + \tfrac{1}{2})^2 - a^2b^2$$

$\Delta p = p2_{k+1} - p2_k$

when $p2_k > 0$,

$\Delta p = b^2(1)(2(x_k+1)+1) - a^2((y_k+1/2)^2 - (y_k+1/2)^2)$

$$\boxed{\Delta p = 2b^2(x_k+1) + b^2}$$

$\Delta p = p2_{k+1} - p2_k$

when $p2_k < 0$,

$\Delta p = b^2(1)(2(x_k+1)+1) - a^2((y_k+1+1/2)^2 - (y_k+1/2)^2)$

$\Delta p = 2b^2(x_k+1) + b^2 - 2a^2[1][y_k+1]$

$$\boxed{\Delta p = 2b^2(x_k+1) + b^2 - 2a^2(y_k+1)}$$

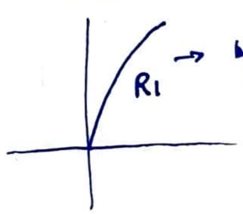
Region 1 unit step in y.

x_k OR x_{k+1}

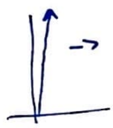
1 unit in y.

mid $\Rightarrow (x_k + 1/2, y_k + 1)$.

same treatment.



R_1 → big slope region –



→ need to take unit steps in y.

Mid Point → $(x_k + 1/2, y_k + 1)$.

$x_k, y_k + 1$

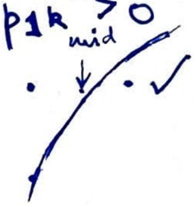
$x_{k+1}, y_k + 1$.

(x_k, y_k)

$$p1_k = b^2(x_k + 1/2)^2 - a^2 \times (y_k+1)^2 - a^2 b^2$$

$p1_k > 0$

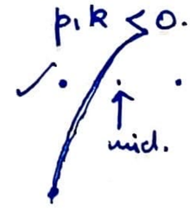
mid



→ x_{k+1}, y_{k+1}

$p1_{k+1} = b^2(x_k+1+1/2)^2 - a^2(y_k+1)^2 - a^2b^2$

$p1_k \leq 0$.



mid.

→ x_k, y_{k+1}.

$p1_{k+1} = b^2(x_k + 1/2)^2 - a^2(y_k+1)^2 - a^2b^2$

$$p1_{k+1} = b^2(x_k + 1 + 1/2)^2 - a^2(y_k + 1)^2 - a^2 b^2$$

Δp, $p1_k > 0$

$$\Delta p = b^2(1).(2)(x_k + 1)$$

$$\boxed{\Delta p = 2b^2(x_k + 1)}$$, then

x_{k+1}, y_{k+1}.

$$p1_{k+1} = b^2(x_k + 1/2)^2 - a^2(y_k+1)^2 - a^2 b^2$$

Δp, $p1_k < 0$.

$\boxed{\Delta p = 0}$, then x_k, y_{k+1}

PARABOLA

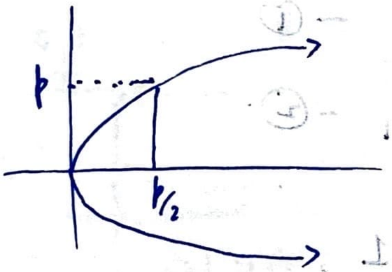
SCAN CONVERSION of Parabola

using Bresenham Algo,

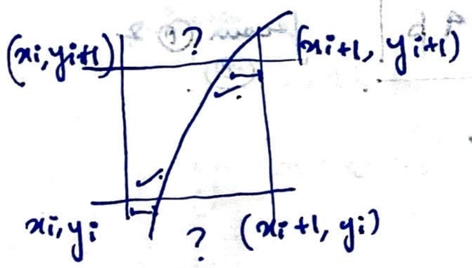
$$y^2 = 2px \quad , \quad \cancel{2}y\frac{dy}{dx} = \cancel{2}p$$

$$\frac{dy}{dx} = \frac{p}{y} \quad \text{if } y = p,$$

$$\boxed{\left(\frac{dy}{dx}\right)_{(p/2,\, p)} = 1}$$



we are here



if $d_i \geq 0$, then $x_{i+1} = x_i + 1$

else $d_i < 0$, then $x_{i+1} = x_i$

$$d_i = (y_i+1)^2 - px_i - px_{i+1}$$

if $d_i \geq 0$, then $d_{i+1} = (y_{i+1}+1)^2$

Region 2, below the x axis

$$d_1 = (y_i+1)^2 - y^2$$

$$d_2 = y^2 - y_i^2 ,$$

$$d_{2i} = d_1 - d_2 \qquad y^2 = 2p(x_i)$$

$$d_{2i} = (y_i+1)^2 + y_i^2 - 2y^2$$

$$d_{2i} = (y_i+1)^2 + y_i^2 - 2 \cdot 2px_i$$

$$d_{2i} = (y_i+1)^2 + y_i^2 - 4px_i \quad - (1)$$

$$d_{2_{i+1}} = (y_{i+1}+1)^2 + y_{i+1}^2 - 4px_{i+1} \quad - (2)$$

If $d_{2i} < 0$, then $y_{i+1} = y_i + 1$,

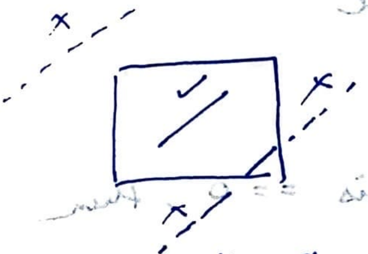
$$\boxed{d_{2i+1} = d_{2i} + 4(y_i + 1) - 4p} \quad \text{from (1) \& (2).}$$

If $d_{2i} > 0$, then $y_{i+1} = y_i$,

$$\boxed{d_{2i+1} = d_{2i} - 4p.}$$

$d1_i\,(0,0) = 1 - p$
$d1_i\,(p/2, p) = 1 + p$
$d2_i\,(p/2, p) = 1 - 2p$

LINE CLIPPING ALGORITHMS.



How to know if the frame buffer has points which are valid?

How to clip?

Why don't we have circular display?
eg. circular - quad. eq
=> line -> linear,

lin - lin eq" solve

quad-line solve

COHEN - SUTHERLAND Algo.

$b_1 b_2 b_3 b_4$	$b_1 b_2 b_3 b_4$	$b_1 b_2 b_3 b_4$	
1001	1000	1010	
			y_{Top}
0001	0000 screen	0010	
			y_{Bottom}
0101	0100	0110	
x_{Left}	x_{Right}		

4 bit codes are used to represent the regions.

=> x_L, x_R & y_B y_T are co-ordinates of screen.

$b_1 b_2 b_3 b_4$

If $x < x_L$, then $b_4 = 1$, else $b_4 = 0$.

If $x > x_R$, then $b_3 = 1$, else $b_3 = 0$.

If $y < y_B$ then $b_2 = 1$, else $b_2 = 0$.

If $y > y_T$ then $b_1 = 1$, else $b_1 = 0$.

Nice! each region is identified uniquely by a combo. of 4 bits.

Each point has a CODE, if $x_L - x_i > 0$, $b_4 = 1$, else 0,
(sign$(x - x_L)$. etc

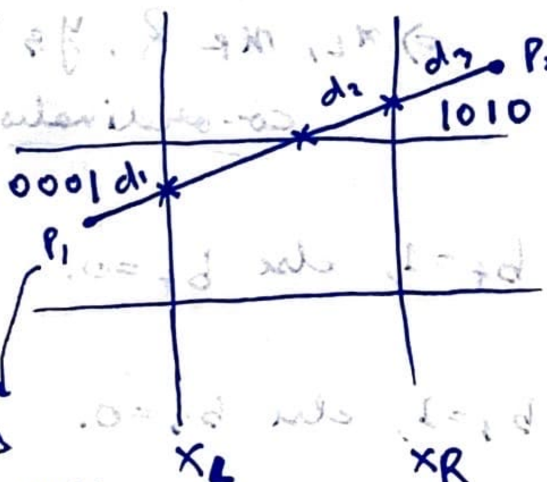
- Take end points P_1 & P_2 of line
- Find their region codes,
- If logical AND of regions is $== 0$, then it is visible.
- If code $P_1 = 0000$

  code $P_2 = 0000$

  AND $= 0 \Rightarrow$ Fully visible line
- If code $P_1 = 1011$

  $P_2 = 1001$

  AND $\neq 0$, Not visible line.



P_2 has code 1010

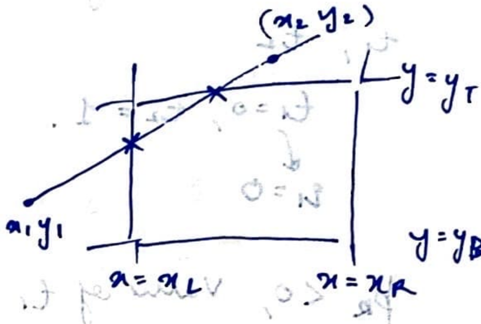
P_1 has code 0001

AND is 0, & P_1 & P_2 non zero then d_1, d_2 & d_3 need to be discarded.

Liang - Barsky - Line - Clipping

$x = t x_1 + (1-t) x_2$, $y = t y_1 + (1-t) y_2$

$0 \leq 1 \leq t$



$\Delta x = x_2 - x_1$

$\Delta y = y_2 - y_1$

$q_1 = x_1 - x_L$

$q_2 = x_R - x_1$

$q_3 = y_1 - y_B$

$q_4 = y_T - y_1$

$p_1 = \Delta x$, $p_2 = -\Delta x$

$p_3 = -\Delta y$, $p_4 = \Delta y$

$t_k = \dfrac{q_k}{p_k}$, $k = 1, 2, 3, 4$

If $p_k = 0$, then line is $\parallel$ to edge.

If $q_k < 0$, the line is completely outside the clipping window.

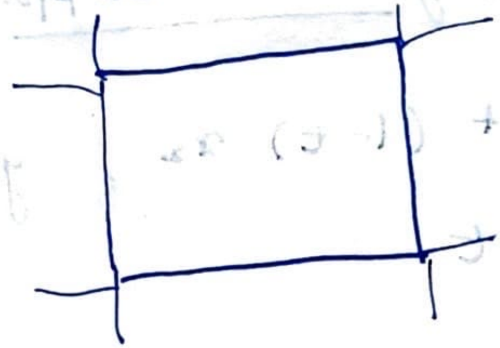
If $p_k < 0$, the line precedes from outside to the inside of particular clipping window.

If $q_k \geq 0$ the line is parallel to clipping edges.

If $p_k > 0$ the line precedes from inside to outside.

The value $t = \dfrac{q_k}{p_k}$, $k = 1, 2, 3, 4$

give us point of intersection.



Consider the values of t as

t_1, t_2

$t_1 = 0$, $t_2 = 1$.

$q_1 = 0$

$p_k < 0$, value of t_1 is found

$p_k > 0$, value of t_2 is found

$$t_1 = \max(0, r_k)$$

entering point

$k = 1, 2, 3, 4$.

How ?

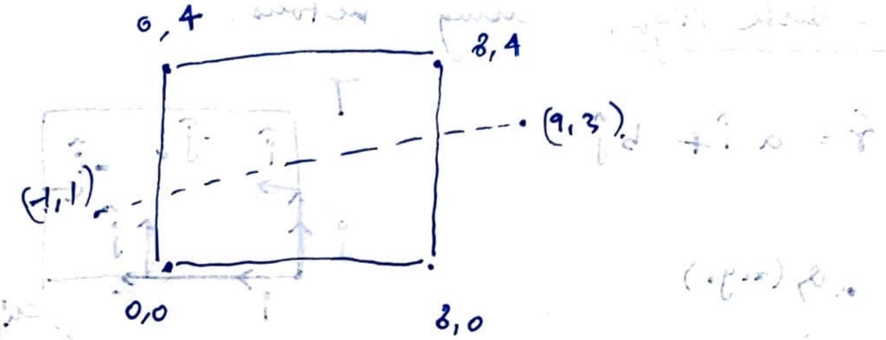
$$t_2 = \min(1, r_k)$$

$k = 1, 2, 3, 4$.

leaving point of the line from window

⋮

8)



$p_1 = -\Delta x = -10$ $\quad q_1 = x_1 - x_L = -1$

$p_2 = 10$ $\quad q_2 = x_R - x_i = 9$

$p_3 = -2$ $\quad q_3 = y_1 - y_B = 1$

$p_4 = 2$ $\quad q_4 = y_T - y_i = 3$

$r_1 = 1/10$ $\quad p_1 < 0, \; q_1 < 0,$

$r_2 = 9/10$ $\quad t_1 = \text{Max}(0, r_1)$

$r_3 = -1/2$ $\quad = \text{Max}(0, 1/10)$

$r_4 = 3/2$ $\quad \boxed{t_1 = 1/10}$.

$p_2 > 0$, find t_2 now.

$t_2 = \text{Min}(1, \frac{9}{10})$

$\boxed{t_2 = 9/10}$

$p_3 < 0$, find t_1 now,

$t_1 = \text{Max}(t_1, r_k) = \text{Max}(\frac{1}{10}, -\frac{1}{2})$

$\boxed{t_1 = \frac{1}{10}}$.

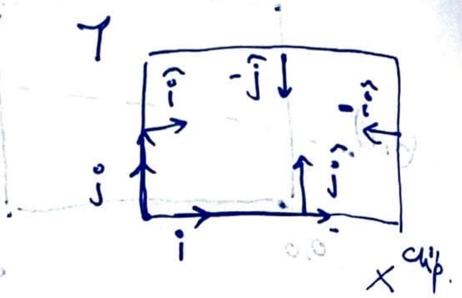
$p_4 > 0$, find t_2 now,

$t_2 = \text{Min}(t_2, r_4)$

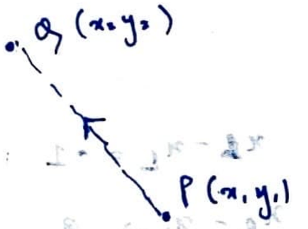
$= \text{Min}(\frac{9}{10}, 3/2) = \boxed{\frac{9}{10} = t_2}$.

find (x, y) for edges by replacing t with t_1 & t_2 & find intersections with edges.

Cyrus - Beck Algo, using vectors.



$\hat{r} = a\hat{i} + b\hat{j}$



$\vec{PQ} = (x_2 - x_1)\hat{i} + (y_2 - y_1)\hat{j}$

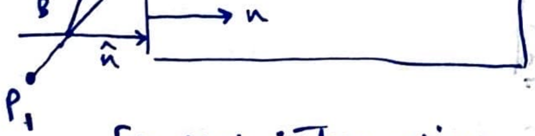
If we have $\hat{r} = a\hat{i} + b\hat{j}$,

normal to it would be

$r_\perp = b\hat{i} - a\hat{j}$.

OR

$r_\perp = -b\hat{i} + a\hat{j}$.



Goal ?

- Find parameters where point is inside, on & outside clip window.

For each intersection of the edges, we find dot product with edge perpendiculars.

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

$(P(t) - f) \cdot \hat{n} > 0$ at A

& $(P(t) - f) \cdot n < 0$ at B

okay, so for all intersections of edges, we find →

$$(P(t) - f_i) \cdot \hat{n_i}$$

$$\downarrow$$

$$\hat{n_i} \cdot (P_1 + (P_2 - P_1) t - f_i]$$

$$\hat{n_i} \cdot \underbrace{(P_1 - f_i)}_{W} + \hat{n_i} \cdot \underbrace{(P_2 - P_1)}_{D} t$$

$$\boxed{\hat{n_i} \cdot \vec{w} + \hat{n_i} \cdot D t}$$

↳ depending on the different values of this product, we will draw the points.

$$\hat{n_i} \cdot \vec{w} + \hat{n_i} \cdot D t = 0$$

if the point is on the edge of clip window.

$$\Rightarrow \boxed{t = - \frac{\hat{n_i} \cdot \vec{w}}{\hat{n_i} \cdot D}}.$$

what if $\bar{n_i} \cdot D = 0$?

1) $D = 0 \Rightarrow P_1 = P_2$

2) $\bar{n_i} \cdot D = 0$,

$D \perp n_i$, means line does not intersect the edges.

If $D = 0$, how to know where the point P lies?

$\vec{w_i} \cdot \vec{n_i} < 0$, point is outside window

$\vec{w_i} \cdot \vec{n_i} > 0$, point is inside the window

$\bar{w_i} \cdot \bar{n_i} = 0$, point is on the i^{th} edge.

$0 \le t < 1$, If $t > 1$, we reject the line.

$$t_1 = -\frac{\bar{w_i} \cdot \bar{n_i}}{D \cdot \bar{n_i}}, \quad i = [1, 4], \quad D \neq 0.$$

Values of t are grouped in 2 classes.

- One group: correspond to begining of line.
- Other group: " " end of line

- Near the begining of line, largest lower limit which is given by $\boxed{D_i \cdot N_i > 0}$.

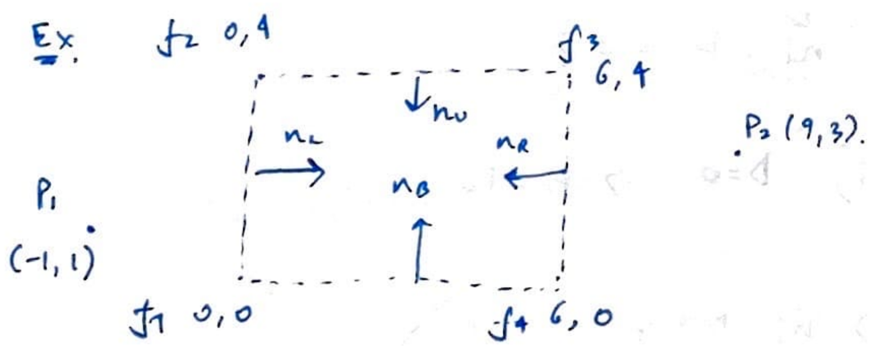
- Near the end of the line, smallest upper limit which is given by $\boxed{D_i \cdot N_i < 0}$.

$t_L = 0$, $t_U =$

$D_i \cdot n_i > 0$ $\quad$ $D_i \cdot n_i < 0$

$t_{Beg} = \text{Max}(0, t)$, $\quad$ $t_{end} = \text{Min}(1, t_{U})$

Ex.



$D = P_2 - P_1 = (10, 2)$

$w_i = P_t - f_i = (-1, 1) + (P_2 - P_1)\,t - (0,0)$

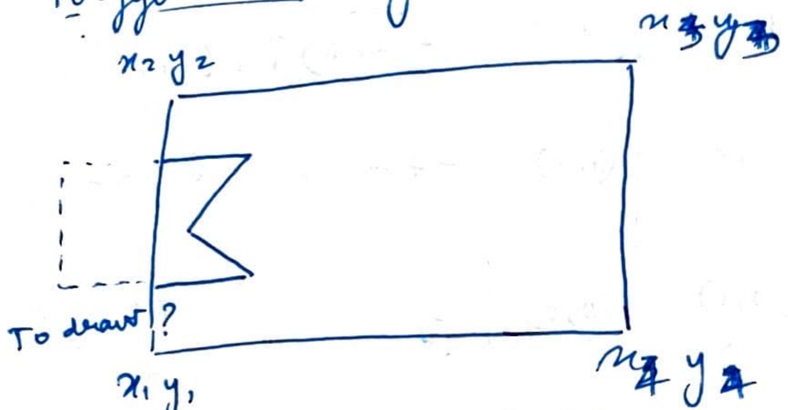
$\boxed{P(t) = (-1, 1) + (10, 2)\,t}$

Edge	Normal	f_i	$P(t) - f_i$
left	$\hat{i} = (1,0)$	$(0,0)$	$(-1+10t)\hat{i} + (1+2t)\hat{j}$
right	$-\hat{i} = (-1,0)$	$(6,4)$	$(-7+10t)\hat{i} + (-3+2t)\hat{j}$
bottom	$\hat{j} = (0,1)$	$(0,0)$	
up	$-\hat{j} = (0,-1)$	$(6,4)$	

$P_1 - f_i$ (w_i)	$w_i \cdot n_i$ ($i, -i, j, -j$)	$D \cdot n_i$	$t = -\dfrac{w_i \cdot n_i}{D \cdot n_i}$	t_{Low} t_{up}
$(-1,1)-(0,0)$ $(-1,1)$	-1	$10 > 0$] t_L	$\frac{1}{10}$	$\max(0, \frac{1}{10})$ $\boxed{\frac{1}{10}}$
$(-7,-3)$	7	$-10 < 0$	$7/10$	$\min(1, \frac{7}{10})$ $7/10$
$(-1,1)$	1	$2 > 0$] t_L	$-1/2$	$\max(-\frac{1}{2}, \frac{1}{10})$ $\boxed{\frac{1}{10}}$
$(-7,-3)$	3	$-2 < 0$	$3/2$	$\min(\frac{7}{10}, 3/2)$ $\Rightarrow 7/10$

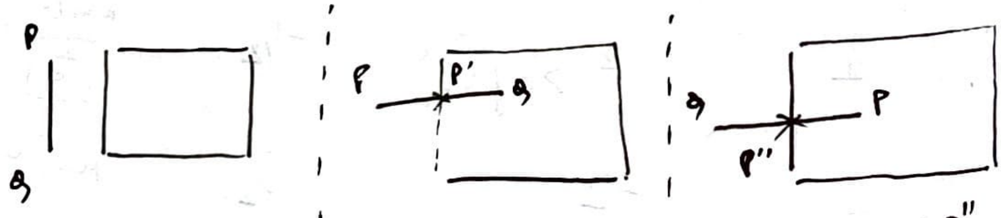
2 groups of values.

Polygon clipping -

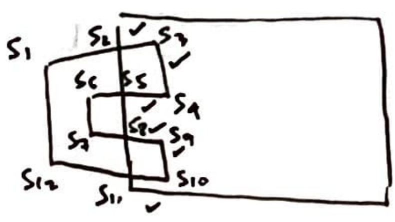


VERTEX	EDGES
$V_1 (x_1, y_1)$	$E_1 : V_1 V_2$
$V_2 (x_2, y_2)$	$E_2 : V_2 V_3$
$V_3 (x_3, y_3)$	$E_3 : V_3 V_4$
$V_4 (x_4, y_4)$	$E_4 : V_4 V_1$

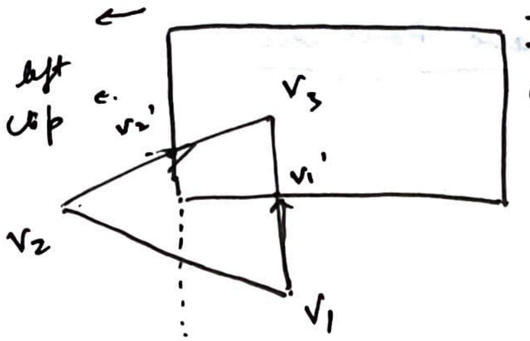
SUTHERLAND - HODGEMAN Algo.



Save none of the points | Save P'Q. | Save PP''.



$S_2 S_3$ ✓
$S_3 S_4$ ✓
$S_4 S_5$ ✓
$S_5 S_8$ ✓
$S_8 S_9$ ✓
$S_9 S_{10}$ ✓
$S_{10} S_{11}$ ✓

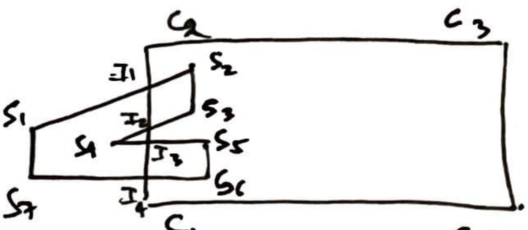


right clip	left clip	Right clip	Bottom clip	Top clip.
$V_1 \rightarrow$	V_1	V_1	V_1'	
$V_2 \rightarrow$	V_2'	V_2	V_2	
$V_3 \rightarrow$	V_3	V_3	V_3	

Read.

no intersection

WEILER ATHERTON ALGO.



1. Start from the first vertex of subject polygon  entering the clip polygon $C_1C_2C_3C_4$.

and go along the edge of the subject polygon as we enter the clipping polygon.

2. When we leave the clipping polygon (window).

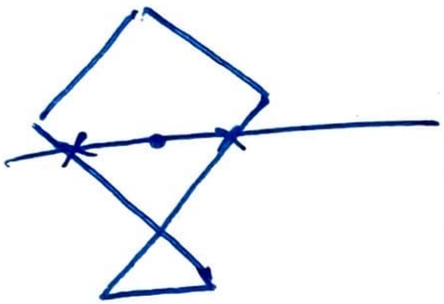
TURN RIGHT.

Now subject polygon becomes the clipping polygon and clipped polygon becomes SUBJECT.

Point inside Polygons.

How to know if a point lies inside a polygon?

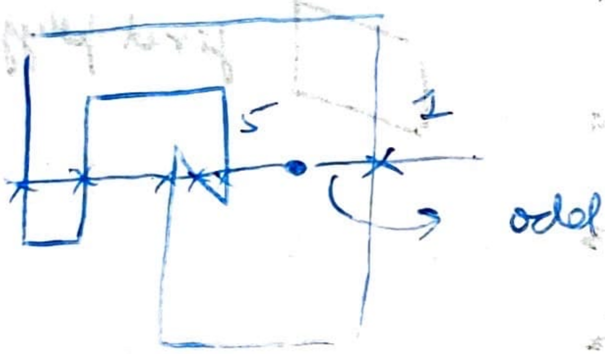
we draw lines, if there are -



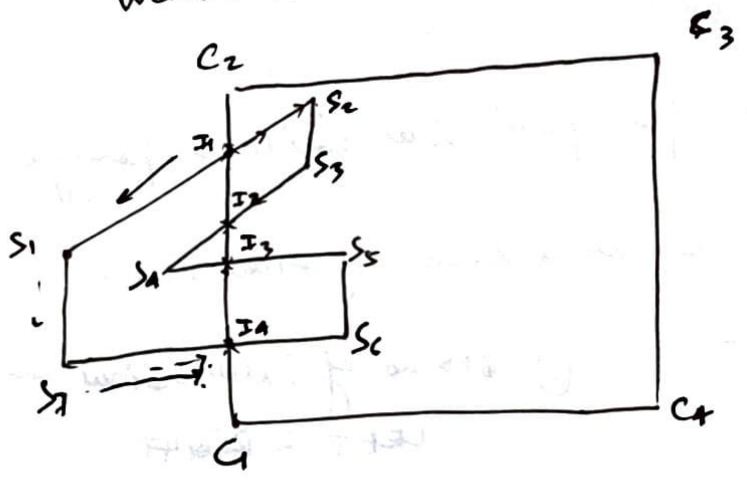
① ODD no of intersections on LEFT & RIGHT

- INSIDE

② EVEN no. of intersections on LEFT & RIGHT



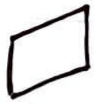
weiler-Atherton



Subject Polygon	Clip polygon.

Clockwise

I_1	C_1
S_2	I_4
S_3	I_3
I_2	I_2
S_4	I_1
I_3	C_2
S_5	C_3
S_6	C_4
I_4	
S_7	
S_1	

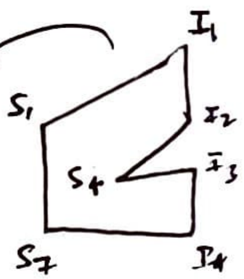


first polygon

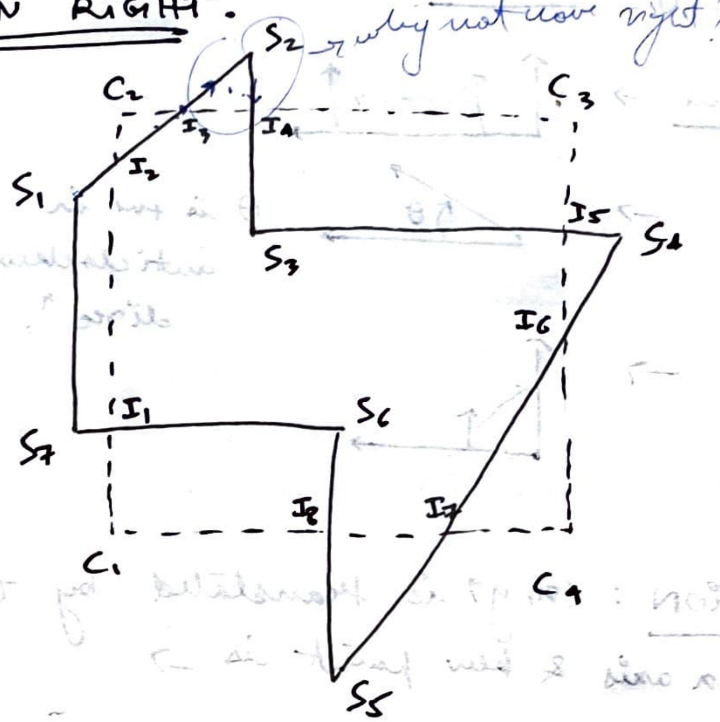
second polygon.

Anticlockwise

I_1	
S_2	C_1
S_3	I_4
I_2	I_3
S_4	I_2
I_3	I_1
S_5	C_2
S_6	C_3
I_4	C_4
S_7	



whenever you are about to leave the clip window, **TURN RIGHT**.

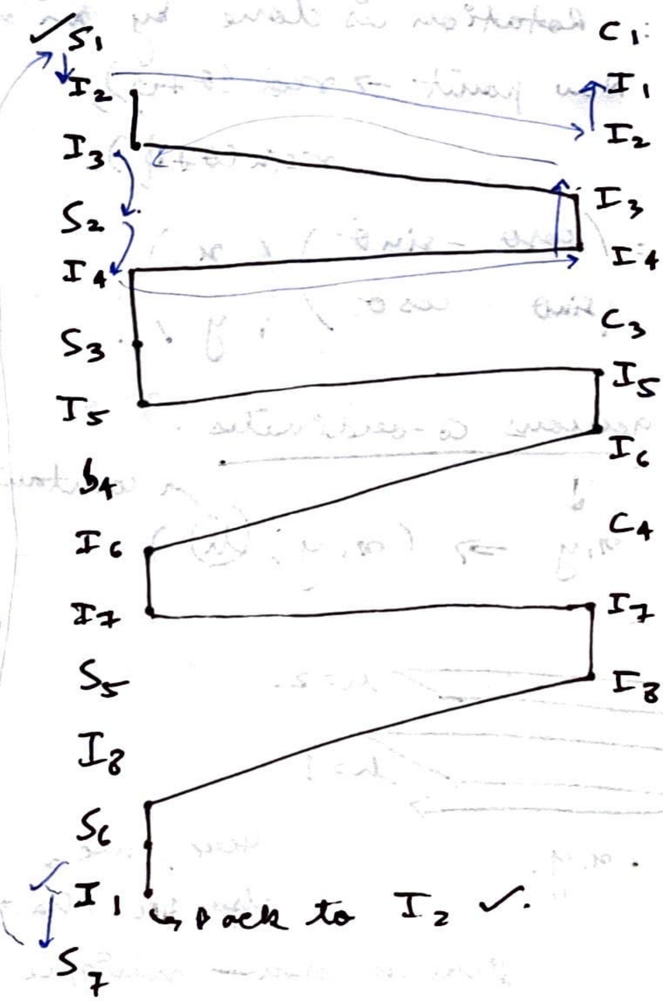


Subject Polygon — Clip Polygon.

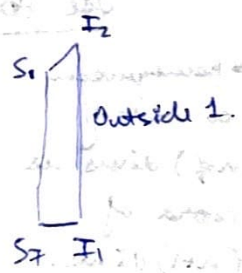
Inner polygon search.

Outer Polygon Search.

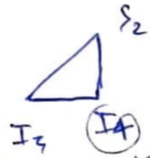
1. I_1 is the first intersection
2. I_2 is used.
3. Use I_3.



& so on



\# From a point, follow the table.



& so on

remove.

2-D Transformations.

Translation $\rightarrow$ 

Rotation $\rightarrow$  θ is +ve in anticlockwise direc^n.

Scaling $\rightarrow$ 

① TRANSLATION : (x, y) is translated by t_x along x axis & new point is $\rightarrow$

$$x' = x + t_x, \quad y' = y + t_y.$$

② ROTATION : Rotation is done by an angle ϕ.

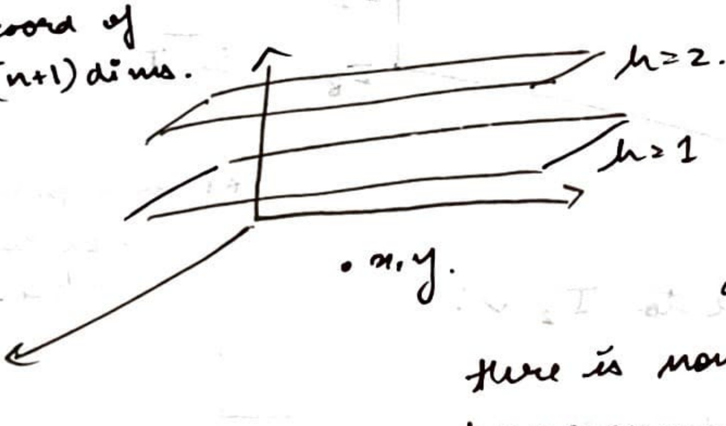
New point $\rightarrow r\cos(\theta+\phi)$,
$r\sin(\theta+\phi)$.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

Use Homogeneous co-ordinates.

$$x, y \Rightarrow (x, y, h)$$ constant

homogeneous co-ords of (n) dims is coord of (n+1) dims.



Here, we also see that there is non-unique homogeneous co-ords.

Translation in Homogeneous co-ords.

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \qquad T(x_1, y_1)$$

Rotation in Homogeneous co-ords.

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \qquad R(\theta)$$

Scaling in Homogeneous co-ords.

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} r_x & 0 & 0 \\ 0 & r_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \qquad S(r_x, r_y).$$

LINEARITY.

$$T(t_{x_1}, t_{y_1}) + T(t_{x_2}, t_{y_2}) = T(t_{x_1} + t_{x_2}, t_{y_1} + t_{y_2})$$

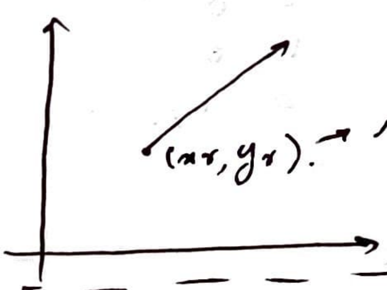
$$R(\theta_1) + R(\theta_2) = R(\theta_1 + \theta_2).$$

$$S(r_{x_1}, r_{x_2}, r_{y_1}, r_{y_2}) = S(r_{x_1}, r_{y_1})\, S(r_{x_2}\, r_{y_2}).$$

We can have combinations of these transformations

TR, ST, ...

Ex.



→ rotate something about this point?

1. Bring point to origin $T(-x_r, -y_r)$
2. Rotate $R(\theta)$
3. Translate back $T(x_r, y_r)$.

$$\left\{ T(x_r, y_r) \,.\, R(\theta) \,.\, T(-x_r, -y_r) \right\}$$

$$\begin{bmatrix} r_{11} & r_{12} & t_{x} \\ r_{21} & r_{22} & t_{y} \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow$$

2×2 matrix preserves orthogonality, length & angle.

↳ Rigid Body Transform

Affine : preserve || lines but not length or angle

Shear : ? 1

Reflection : ? 2

Assignment: find T-Matrix for ① & ② about the line $ax + by + c = 0$, $a \neq 0$, $b \neq 0$, $c \neq 0$

3D Transformations.

$$T = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 0 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$ Translation

$$S_{caling} = \begin{pmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$ Scaling.

$$R_z = \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

$x \rightarrow y \rightarrow z \rightarrow x$

replace x with

$$x' = x\cos\theta - y\sin\theta$$

$$y' = x\sin\theta + y\cos\theta$$

$$z' = z$$

$$x' = x$$

$$y' = y\cos\theta - z\sin\theta$$

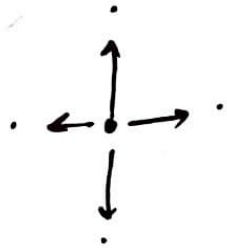
$$z' =$$

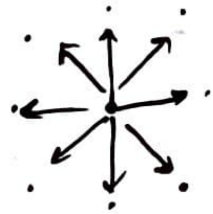
$$R_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

$$R_y = \begin{pmatrix} \cos\theta & 0 & -\sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

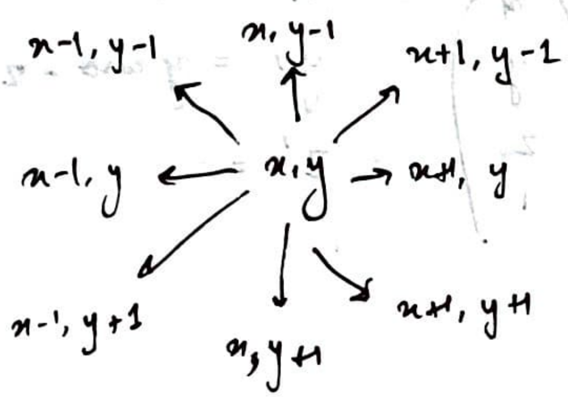
POLYGON FILL ALGOS.

4 connected & 8 connected algos.

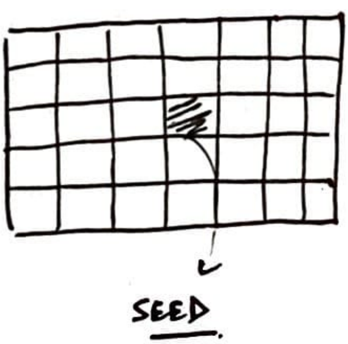




↳ more coherence, so this is chosen.



SEED FILL ALGORITHM.

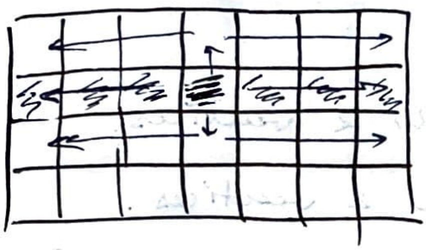


1. Pick a seed (x, y)
2. Pop seed from the stack
3. Shade the pixel.
4. Find the 4 neighbours of
   - if neighbour unshaded & not a boundary
   - push to stack.
5. Continue till stack empty.

$O(N^2)$ space

→ handling pixels in mem, instead of lines.

SCAN-LINE SEED FILL ALGO.



- Seed pixel pushed to the stack
- Now, top pixel is removed from the stack.

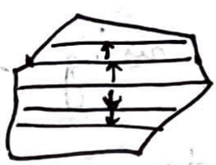
- The scan line containg the seed is filled till left boundary & right boundary.

- Save the left end & right of the currently filled scan line.

- Check if above & below lines are filled in [x left, x right].

- If not filled, push a seed, x seed on stack where $x_L \leq x$ seed $\leq x$right.

- continue for unfilled lines.



& so on

Scan line polygon fill algorithm.

even & odd parity - ?

y = c is the scan line

All EDGE Table,
GLOBAL EDGE ",
ACTIVE EDGE ".

Let (x_1, y_1), (x_2, y_2) be the edge ~~of~~ with these vertices.

i) Minimum y value of the 2 vertices.

ii) Maximum y value of the 2 vertices.

iii) x - value associated with min. y - value.

iv) The slope of the edge.

ALL EDGE TABLE (AET).

y_{min}	y_{max}	x val, min	$1/m$

GLOBAL EDGE TABLE

→ Fill the edges from bottom to top & left to right.

→ The GET should be inserted with edges GROUPED by INCREASING minimum y - value.

→ Edges with same min. value are sorted on minimum x value using steps →.

i) Place the first edge with a slope $\neq 0$ in GET, otherwise don't place it in GET.

ii) For every other edge start at index $= 0$,

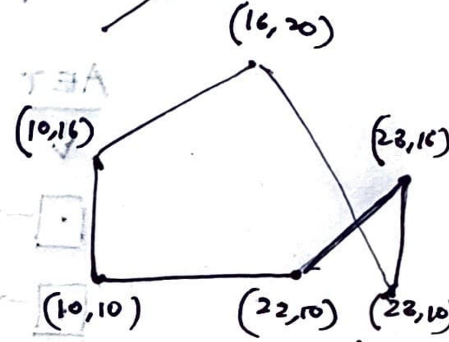
- the current edge's y value is <= of the edge at the current index in GET.
- Increase the index

⋮

Example.

ALL
AET - ~~global~~ Edge Table.

		y_{min}	y_{max}	x_{val}	$1/m$
10,10	0	10	16	10	0
10,16	1	16	20	10	1.5
16,20	2	10	20	28	-1.2
28,10	3	10	16	28	0
28,16	4	10	16	22	1
22,10.	5	10	10	10	∞



GLOBAL EDGE TABLE - sort first by y_{min}, then by x_{val}.

first by y. then by x.] careful
↳ use insertion sort.

0	0
2	5
3	4
4	
5	
1	

Final GLOBAL EDGE TABLE.

	y_{min}	y_{max}	x_{val}	$1/m$
0	10	16	10	0
1	10	16	22	1
2	10	16	28	0
3	10	20	28	-1.2
4	16	28	10	1.5
5				

$y_{min} = 10$

x val min = 10.

Active Edge Table. ⇒ y_{min} is dropped from the global Edge table.

↓

For 1 particular value of y_{min}, here 10, drop y_{min} column & then put the other cells here. ↪ y_{min} is the SCAN LINE.

AET

↓

[•] → | 16 | 10 | 0 |

[•] → | 16 | 22 | 1 |

[•] → | 16 | 28 | 0 |

[•] → | 20 | 28 | -1.2 |

GET (left)

16	20	10	1.5

Filling start with $x = 10$, parity is odd.

All pixels from $x = 10$ to $x = 22$ are filled.

Parity even, don't fill,

odd fill, even don't.

Parity? Intersection counts parity.