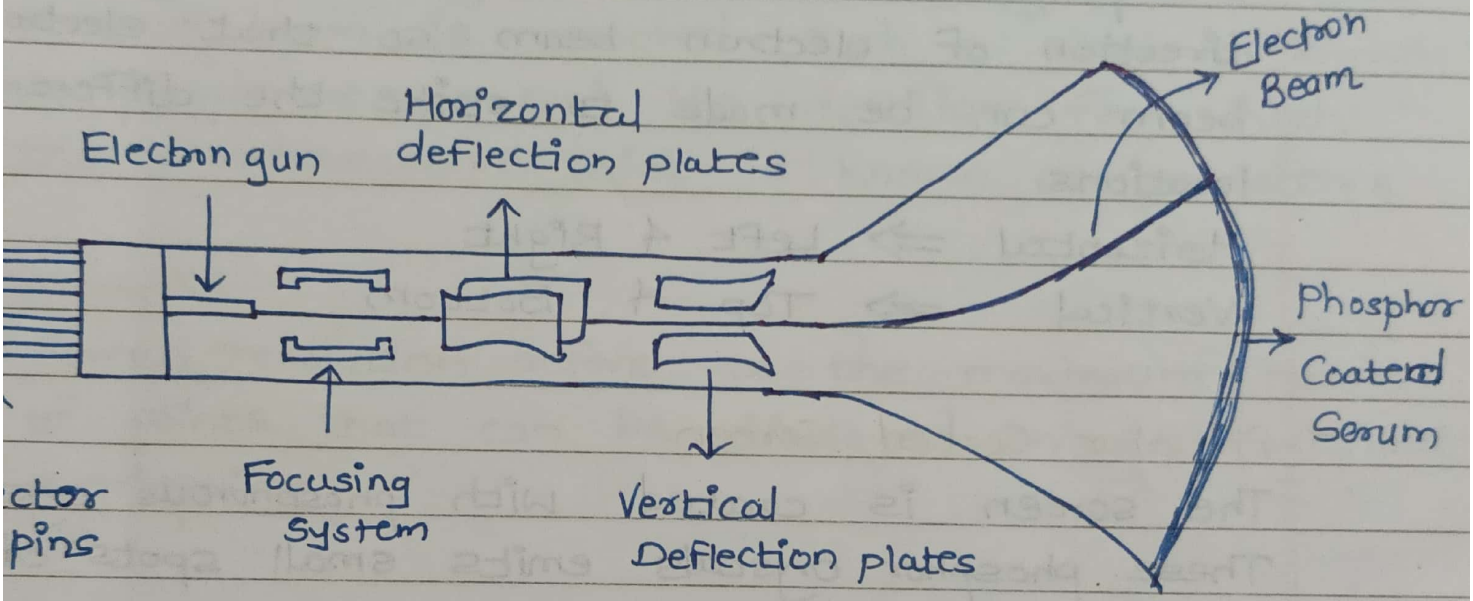
Introduction to Computer Graphics:

Cathode Ray Tube:

Just a vaccum glass bottle.

Converts an electrical signal into a visual display



Components of CRT:

Electron gun

- Generates negatively charged electrons.
- Electron gun consists of cathode and heating Filament.
- When heat is supplied to cathode by a heating filament it $\bar{e}$ become loose and gets emitted from cathode surface.

$\bar{e}$ - Electrons.

2. Focusing System -
The purpose of focusing system is to force the electron beam to converge into small spots and travel in straight line.

3. Deflection System
The purpose of deflection is to change the direction of electron beam so that electron beam can be made to strike the different locations.
Horizontal ⇒ Left & Right
Vertical ⇒ Top & Bottom

4 Phosphor Coated Screen
The screen is coated with phosphorous crystals. These phosphor crystals emits small spots of light when the electron beam strikes on them.

Various terms associated with CRT :

1. Pixel -
A smallest color ~~dro~~ spot that can be displayed on the screen.

2. Fluorescence -
It is the property of phosphor crystals that they glow when they hit by a high energy electron beam.

3. Phosphorescence -
The phosphor crystals continue to glow even after electron beam is removed. This light glow produced is called phosphoresence.

4. Persistence :
The phosphor crystals continue to glow even the electron beam is removed. Such a glow is known as phosphoreseence and the duration for which the phosphoreseence exist is known as persistence.

5. Resolution :
Screen resolution refers to the maximum number of points that can be displayed on CRT without overlapping.

6. Aspect Ratio :
It is defined as the ratio of horizontal point to the vertical points.

$$\text{Aspect Ratio (AR)} = \frac{\text{No. of horizontal points}}{\text{No. of vertical points}}$$

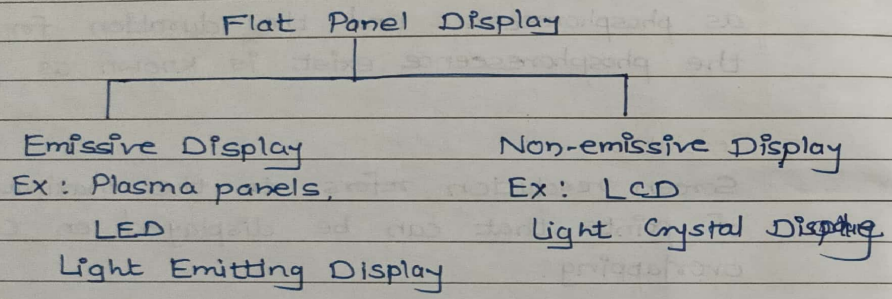
Example : Screen resolution is 800 × 600 then

$$AR = \frac{800}{600} = \frac{4}{3} = \boxed{4:3}$$

Flat Panel Display :

It refers to a class of video devices that have reduced volume, weight and power requirement compare to CRT.

Ex : Small T.V. monitor, calculator, laptop, advertisment board, computors



Emissive Display :

The devices that convert the electrical energy into light.

Ex : plasma panel, LED.

Non-emissive Display :

It uses optical effect to convert sunlight or light from some other source into graphics pattern.

Types of Flat Panel Display :

1. Liquid Crystal Display :
   - LCD Flat panel contains a liquid crystal solution that is pressed between two thin sheets of polarizing material.
   - When electricity is applied, the crystals in the solution align to prevent any light from passing throught.

2. Light Emitting Diode
   - It uses the light emitting diodes as a backlight instead of traditional Fluoreseent light

Line -

- Point is the fundamental element of picture representation.

If two points used to specify a line are (x_1, y_1) and (x_2, y_2)
Equation of line is as

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

$$y = \frac{y_2-y_1}{x_2-x_1}(x-x_1) + y_1$$

or $y = mx + b$

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

and $b = y_1 - mx_1$

- Line extends forever both forward and backward.
- In graphics we need to display only pieces of lines. When we consider the piece of line i.e. only those points which lie between two endpoints, then it is called line segment.

Line Drawing Algorithms -

- The process of 'turning on' the pixels for a line segment is called vector generation or line generation.
- The algorithms for them are known as line drawing algorithms.

Characteristics of line -

i) Line should appear as a straight line and it should start and end accurately.
ii) Line should be displayed with constant brightness along its length independent and orientation.
iii) The line should be drawn rapidly.

Digital Differential Algorithm -

1. Read all the line endpoints (x_1, y_1) and (x_2, y_2) such that they are not equal.
2. $dx = |x_2 - x_1|$
   $dy = |y_2 - y_1|$
3. if $(dx > dy)$ then
   length = dx
   else
   length = dy
4. Find increment factor
   $\Delta x = dx / length$
   $\Delta y = dy / length$

5. Plot (int(x), int(y))

6. i = 1
while (i ≤ length)
{
x = x + Δx
y = y + Δy
plot (int(x), int(y))
i = i + 1
}

7. stop

Example
Rasterize the line from (0,0) to (4,6) using DDA Algorithm

$x_1 = 0, \quad y_1 = 0, \quad x_2 = 4, \quad y_2 = 6$

$dx = x_2 - x_1 = 4 - 0 = 4$
$dy = y_2 - y_1 = 6 - 0 = 6$

as $dy > dx$
length = $dy = 6$.

$\Delta x = dx / length = 4/6 = 0.66 \approx 0.7$
$\Delta y = dy / length = 6/6 = 1$

plot 1st point (0,0)

i=1

i) $x = x + \Delta x \Rightarrow x = 0 + 0.7 = 0.7$
$y = y + \Delta y \Rightarrow y = 0 + 1 = 1$
plot (0,1)
plot(0,1)

ii) $x = x + \Delta x \Rightarrow 0.7 + 0.7 = 1.4$
$y = y + \Delta y \Rightarrow 1 + 1 = 2$
plot (1,2)

iii) $x = x + \Delta x \Rightarrow 1.4 + 0.7 = 2.1$
$y = y + \Delta y \Rightarrow 2 + 1 = 3$
plot (2,3)

iv) $x = x + \Delta x \Rightarrow 2.1 + 0.7 = 2.8$
$y = y + \Delta y \Rightarrow 3 + 1 = 4$
plot (2,4)

v) $x = x + \Delta x \Rightarrow 2.8 + 0.7 = 3.5$
$y = y + \Delta y \Rightarrow 4 + 1 = 5$
plot (3,5)

vi) $x = x + \Delta x \Rightarrow 3.5 + 0.7 = 4.2$
$y = y + \Delta y \Rightarrow 5 + 1 = 6$
plot (4,6)

i	plot	x	y
	(0,0)	0	0
1	(0,1)	0.7	1
2	(1,2)	1.4	2
3	(2,3)	2.1	3
4	(2,4)	2.8	4
5	(3,5)	3.5	5
6	(4,6)	4.2	6

stop

2. Rasterize the line with endpoints (20, 10) and (30, 18) using DDA Algorithm.

$x_1 = 20, \; y_1 = 10$
$x_2 = 30, \; y_2 = 18$

$dx = x_2 - x_1 = 30 - 20 = 10$
$dy = y_2 - y_1 = 18 - 10 = 8$

as $dx > dy$
length = $dx = 10$.

$\Delta x = dx / length = 10/10 = 1$
$\Delta y = dy / length = 8/10 = 0.8$

plot 1st point (20, 10)

$i = 1$

i) $x = x + \Delta x \Rightarrow x = 20 + 1 = 21$
$y = y + \Delta y \Rightarrow y = 10 + 0.8 = 10.8$
(21, 10)

ii) $x = x + \Delta x \Rightarrow 21 + 1 = 22$
$y = y + \Delta y \Rightarrow 10.8 + 0.8 = 11.6$
plot (22, 11)

iii) $x = x + \Delta x \Rightarrow 22 + 1 = 23$
$y = y + \Delta y \Rightarrow 11.6 + 0.8 = 12.4$
plot (23, 12)

iv) $x = x + \Delta x \Rightarrow 23 + 1 = 24$
$y = y + \Delta y \Rightarrow 12.4 + 0.8 \Rightarrow 13.2$
plot (24, 13)

v) $x = 24 + 1 = 25$
$y = 13.2 + 0.8 = 14$
plot (25, 14)

vi) $x = 25 + 1 = 26$
$y = 14 + 0.8 = 14.8$
plot (26, 14)

vii) $x = 26 + 1 = 27$
$y = 14.8 + 0.8 = 15.6$
plot (27, 15)

viii) $x = 27 + 1 = 28$
$y = 15.6 + 0.8 = 16.4$
plot (28, 16)

ix) $x = 28 + 1 = 29$
$y = 16.4 + 0.8 = 17.2$
plot (29, 17)

x) $x = 29 + 1 = 30$
$y = 17.2 + 0.8 = 18$
plot (30, 18)

$P_1 = (x_1, y_1) = (0,0)$
$P_2 = (x_2, y_2) = (-4,-6)$

i) $dx = |x_2 - x_1| = |-4+0| = 4$
$dy = |y_2 - y_1| = |-6-0| = 6$

ii) as $dy > dx$
length = $dy = 6$.

iii)
$\Delta x = dx / length = 4/6 = 0.66 \approx 0.7$
$\Delta y = dy / length = 6/6 = 1$

iv)
plot (0,0)

v)
$x = x + \Delta x \Rightarrow 0 + 0.7 = 0.7$
$y = y + \Delta y \Rightarrow$

Bresenhams line drawing algorithm.
For $|m = dy/dx| < 1$

1. Read the line end points (x_1, y_1) and (x_2, y_2) such that they are not equal
2. $dx = |x_2 - x_1|$ and $dy = |y_2 - y_1|$
3. Initialize starting point
   $x = x_1$
   $y = y_1$
   plot (x,y);
4. $e = 2dy - dx$
5. $i = 1$
6. While ($e \geq 0$)
   {
   $y = y + 1$
   $e = e - 2 * dx$
   }
   $x = x + 1$
   $e = e + 2 * dy$
7. plot (x,y)
8. $i = i + 1$
9. iF ($i \leq dx$) then goto step 6
10. Stop

Example -
Consider the line From (5,5) to (13,9). Use the Bresenhams algorithm to rasterize the line.

i) $(x_1, y_1) = (5, 5)$
$(x_2, y_2) = (13, 9)$

ii) $dx = |x_2 - x_1| = |13 - 5| = 8$
$dy = |y_2 - y_1| = |9 - 5| = 4$

iii) $x = 5$, $y = 5$.

iv) $e = 2 * dy - dx$
$= 2 * 4 - 8$
$= 8 - 8 = 0$
$e = 0$

v) $i = 1$

vi) While $(e \geq 0)$
a) $y = y + 1 \Rightarrow y = 5 + 1 = 6$ [y=6]
$e = e - 2 * dx \Rightarrow e = 0 - 2 * 8 = -16$

$x = x + 1 \Rightarrow 5 + 1 = 6$ [x=6]
$e = e + 2 * \Delta y \Rightarrow e = -16 + 2 * 4 = -8$

vii) plot (6,6)

viii) $i = i + 1 = 1 + 1 = 2$

ix) $i \leq dx$
$2 \leq 8$
goto step 6.

b) As $e \geq 0$ i.e. $(-8 \geq 0)$ not satisfied
value of y will not be incremented
y will be as it is i.e. [y = 6]
$x = 6 + 1 = 7$ [x = 7]
$e = e + 2 * dy \Rightarrow e = -8 + 2 * 4 = 0$

plot (7,6)

$i = i + 1 \Rightarrow 3$; $3 \leq 8$

c) as $e \geq 0$ i.e. $0 \geq 0$ satisfied
$y = y + 1 \Rightarrow y = 6 + 1 = 7$ [y=7]
$e = e - 2 * \Delta x \Rightarrow e = 0 - 2 * 8 = -16$

$x = x + 1 \Rightarrow 7 + 1 = 8$ [x=8]
$e = e + 2 * \Delta y \Rightarrow -16 + 2 * 4 = -8$

plot (8,7)

$i = 4$

d) as $-8 \geq 0$ not satisfied
y will be as it is i.e. [y=7]

$x = x + 1 \Rightarrow 8 + 1 = 9$ [x=9]
$e = e + 2 * dy \Rightarrow e = -8 + 2 * 4 = 0$

plot (9,7)

i=5

e) as e>=0 0>=0

y = y+1 ⇒ 7+1 = 8 y = 8

e = e-2*dy = -16

x = x+1 ⇒ 9+1 = 10 x = 10

e = e+2*dx = -16+8 = -8

plot (10,8)

i=6

f) as e>=0 -8>=0 not satisfied

y = 8

x = x+1 ⇒ 10+1 = 11 x = 11

e = e+2*dx = -8+8 = 0

plot (11,8)

i=7

g) as e>=0 , 0>=0

y = y+1 = 8+1 = 9 y = 9

e = e-2*dy = -16

x = x+1 = 11+1 = 12 x = 12

e = e+2*dx = -16+8 = -8

plot (12,9)

i=8

as e>=0 , -8>=0 not satisfied

y = 9

x = x+1 ⇒ 12+1 = 13 x = 13

e = e+2*dx ⇒ e = 0

plot (13,9)

i = 8+1 = 9

i <= dx 9 <= 8 not satisfied.

stop

i	plot	x	y	e
		5	5	0
	(5,5)	6	6	-8
1	(6,6)	7	6	0
2	(7,6)	8	7	-8
3	(8,7)	9	7	0
4	(9,7)	10	8	-8
5	(10,8)	11	8	0
6	(11,8)	12	9	-8
7	(12,9)	13	9	0
8	(13,9)	14	10	-8

Consider the line from (4,9) to (7,7) Draw a line using Bresenham's line drawing algorithm.

$(x_1, y_1) = (4,9)$
$(x_2 - y_2) = (7,7)$

$dx = |x_2 - x_1| = |7-4| = 3$
$dy = |y_2 - y_1| = |7-9| = 2$

Consider the line from (1,1) to (5,3)

$(x_1, y_1) = (1,1)$
$(x_2, y_2) = (5,3)$

$dx = |x_2 - x_1| = |5-1| = 4$
$dy = |y_2 - y_1| = |3-1| = 2$

$x = 1$, $y = 1$

$e = 2 * dy - dx$
$= 2 * 2 - 4$
$= 4 - 4 = 0$

a) $i = 1$, $e \geq 0$

$y = y + 1 \Rightarrow y = 1 + 1 = 2$ → $y = 2$
$e = e - 2 * dx \Rightarrow e = 0 - 8 = -8$

$x = x + 1 \Rightarrow 1 + 1 = 2$ → $x = 2$
$e = e + 2 * dy \Rightarrow e = -8 + 4 = -4$

b) $i = 2$, $e \geq 0$ $-4 \geq 0$ not satisfied

$y = 2$.

$x = x + 1 = 2 + 1 = 3$ → $x = 3$
$e = e + 2 * dx \Rightarrow e = 0$

plot (3,2)

c) $i = 3$, $e \geq 0$ $0 \geq 0$

$y = y + 1 \Rightarrow 2 + 1 = 3$ → $y = 3$
$e = e - 2 * dy \Rightarrow e = -8$

$x = x + 1 \Rightarrow 3 + 1 = 4$ → $x = 4$
$e = e + 2 * dx \Rightarrow e = 0 - 4 = -4$

plot (4,3)

d) $i = 4$, $e \geq 0$ $-4 \geq 0$ not satisfied

$y = 3$ → $y = 3$

$x = x + 1 \Rightarrow 4 + 1 = 5$ → $x = 5$
$e = e + 2 * dx \Rightarrow e = 0$

plot (5,3)

i	Plot	x	y	e
		1	1	0
1	(1,1)	2	2	-4
2	(2,2)	3	2	0
3	(3,2)	4	3	-4
4	(4,3)	5	3	0
5	(5,3)			

$(x_1, y_1) = (0,0)$

$(x_2, y_2) = (4,6)$

$dx = |x_2 - x_1| = |4-0| = 4$

$dy = |y_2 - y_1| = |6-0| = 6$

plot (0,0)

$e = 2*dy - dx$

$= 2*6 - 4$

$= 12 - 4 = 8$

a) $i = 1$, $e \geq 0$

$y = y + 1 \Rightarrow y = 0 + 1 = 1$

$e = e + 2 * dy$

$= -5 + 2 * 2$

$= -5 + 4$

$e = -1$

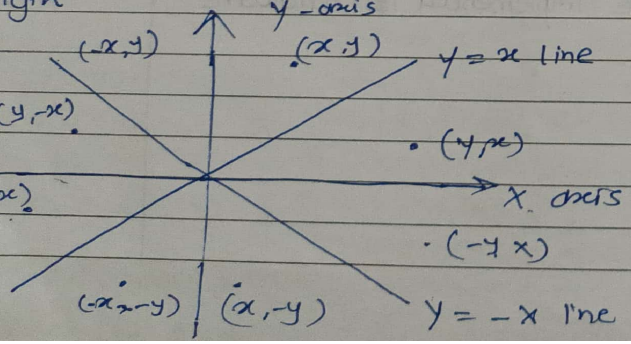
i	Set Pixel	e	X	Y
		1	2	7
1.	(2,7)	-1	3	6
2.	(3,6)	3	4	6
3.	(4,6)	2	5	5

Vector Generation	Bresenham's Algorithm
1. Uses floating point arithmetic	1. Uses only integer addition, sub, multiplicⁿ
2. Takes more time	2. Quicker than vector generation
3. Less efficient	3. More efficient
4. Speed is important, needs to be implemented in hardware	4. Hardware implementation is not required.

Bresenham's Circle drawing algorithm -

1. Read the radius of circle (r).
2. $d = 3 - 2r$
3. $x = 0, y = r$
4. do
   {
   plot (x,y); $x = x+1$
   if $(d < 0)$ then
   {
   $d = d + 4x + 6$
   }
   else
   {
   $d = d + 4(x-y) + 10$
   $y = y - 1$
   }
   ~~$x = x + 1$~~
   } while $(x < y)$
5. Stop

Remaining part of circle can be drawn by reflecting point about y axis, x axis and about origin

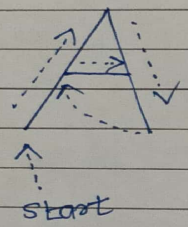


plot (x,y) — x, y
plot $(-x,y)$ — $x, -y$
plot $(x,-y)$ — y, x
plot $(-x,-y)$ — $y, -x$
plot $(-x,-y)$ — $-x, -y$
plot $(-y,x)$ and — $-y, -x$
plot $(-x,y)$ — $-x, y$
$-y, x$

Character Generation -

1 Stroke Method -

- Uses small line segments to generate a character.
- The small series of line segments are drawn like a stroke of pen to form a character.
- We can generate our own stroke method by calling line drawing algorithm.
- It supports scaling by changing length of line segment.



2 Starbust Method -

- A fix pattern of line segments are used to generate characters.
- There are 24 line segments.
- Out of these 24 linesegments, highlight the line segments those are necessary to draw a particular character.
- It is known as starbust Method.
- Pattern of particular characters are stored in the form of 24 bit code.
- Each bit represents one line segment. Bit is set to one to highlight line segment; otherwise set to zero.

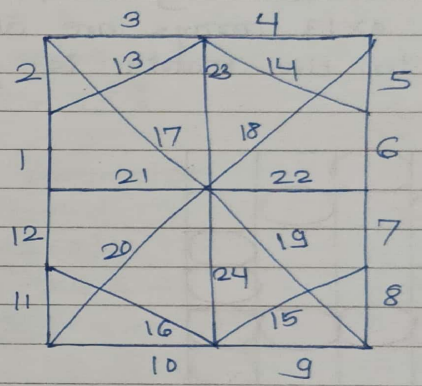
Character A

0011 0000 0011 1100 1110 0001

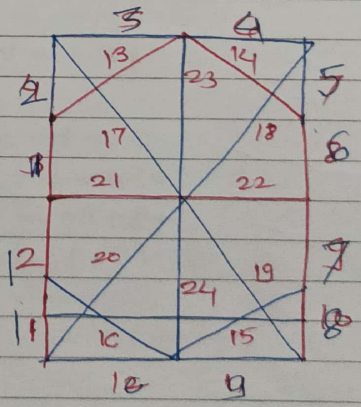
24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

Character M

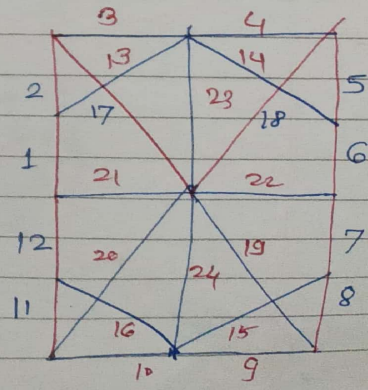
0000 0011 0000 1100 1111 0011



Character A

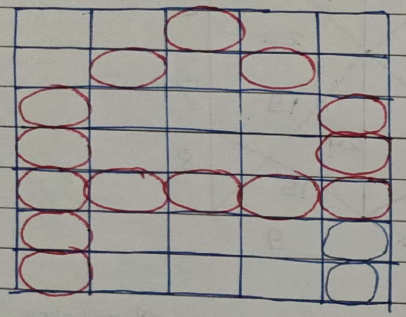


Character M



Bitmap Method -

- Also called as dot matrix method.
- The character is represented by array of dots in the matrix form.
- It is two dimensional array having rows and columns.
- 5X7 array is commonly used.
- 7X9 and 9X13 arrays are also used.
- Each dot in the matrix is pixel.



Bresenhams Circle Drawing algorithm Example

Given the centre point coordinates (0,0) and radius as 8, generate all the points to form a circle.

$(x_c, y_c) = (0,0)$

radius = 8

Assign starting point

$x = 0, y = 8$.

Decision parameter

$d = 3 - 2r$

$= 3 - 2(8)$

$= 3 - 16 = -13$

$\boxed{d = -13}$

$x = x + 1$

$= 0 + 1$

$\boxed{x = 1}$

As $d < 0$

$d = d + 4(x) + 6$

$= -13 + 4(1) + 6$

$= -13 + 10$

$\boxed{d = -3}$

$x = x + 1$

$= 1 + 1$

$\boxed{x = 2}$

$\boxed{y = 8}$

As $d < 0$

$d = d + 4(x) + 6$

$= -3 + 8 + 6$

$\boxed{d = 11}$

$x = x + 1$

$= 2 + 1$

$\boxed{x = 3}$

$\boxed{y = 8}$

Aliasing and Antialising :

Aliasing :

- The process by which smooth curves and other lines become jagged because the resolution of graphics device or file not enough to represe a smooth curve.
- It is known as alising.
- It can be reduced by adjusting intensities of pixels.
- Can be minimized by increasing resolution.

Antialising -

- It is the smoothing of jagged images / edges in digital images by averaging the colors of pixels at a boundary.

Methods -

1. Using high resolution display -

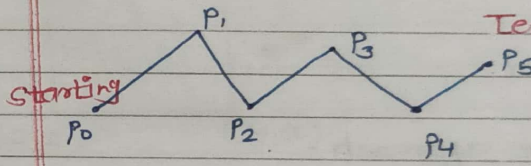
Displaying object at a greater resolution is a technique to decrease aliasing effect.

Ex: OLED and Retina displays in Apple produc both have high pixel densities which results in jaggies they are blurry and invisible to the human eye.

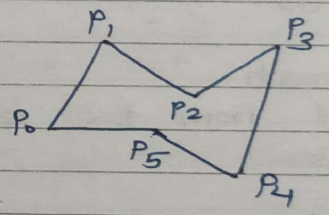
Post filtering / Super Sampling :
Reduce the pixel size while improving the sampling resolution

Unit II
Polygons, 2D Transformations

Polyline -
- A chain of connected line segments.
- Specified by the vertices P_0, P_1, P_2 and so on
- First vertex ⇒ starting point
  Last vertex ⇒ Final / terminal point.
- When starting and endpoint of polyline are same, then it is known as polygon.

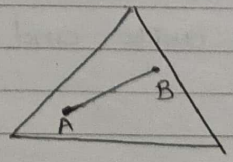


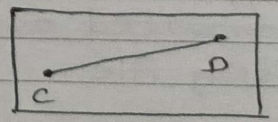
Polyline

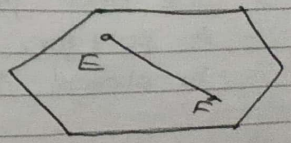


Polygon

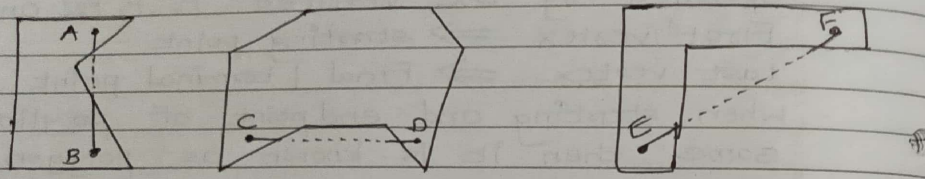
Types of Polygon -
1. Convex polygon.
   Line segment joining any two points wit
   the polygon lies completely inside the poly







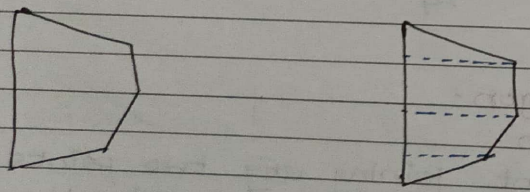
Concave polygon -
Line segment joining any two points within the polygon may not lie completely inside the polygon.



Polygon Representation -

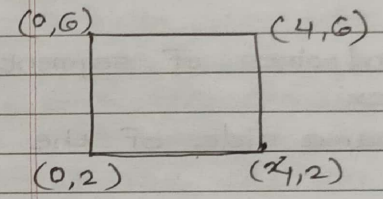
1. Polygon drawing primitive approach -
Directly draw the polygon shapes.

2. Trapezoid primitive approach.
Trapezoids are formed from two scan lines and two line segments.



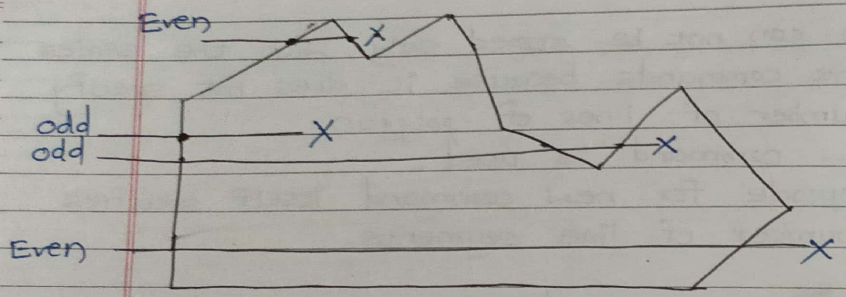
3. Line and point approach.
- A polygon is represented as a unit and it is stored in display file.
- Polygon can not be stored only with the series of line commands because it does not specify the number of lines of polygon.
- A new command is used.
- The opcode for new command itself specifies the number of line segments.

Opcode operand 1 operand 2

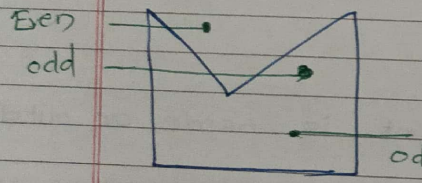


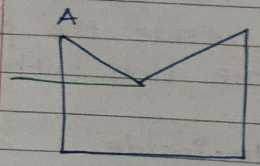
DF_OP	DF_X	DF_Y
4	0	2
2	0	6
2	4	6
2	4	2
2	0	2

An Inside Test -
1. Even-odd test
- To check whether the point is inside or outside polygon.
- Draw a line segment between the point in question and a point known to be outside the polygon.
- Count no. of intersections of the line with the polygon boundary.
- Odd intersections ⇒ ~~outside~~ inside
Even intersections ⇒ outside.

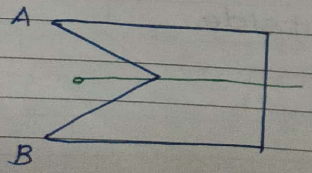


- If the intersection point is the vertex of the polygon :
  * Look the other two end points of segments which meet at the vertex.
  * If they lie on the same side of the constructed line



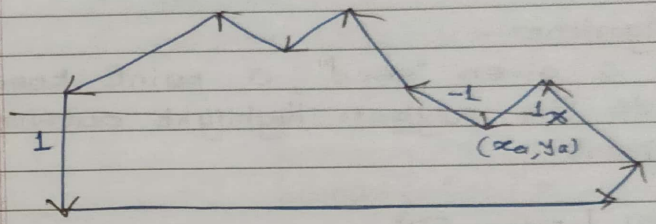


On same side of line
no. of intersections = even = 2
2+1 = 3
odd number ⇒ inside

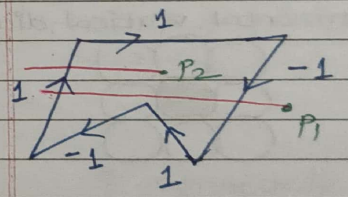


not on same side of line
no. of intersections = odd = 1
1+1 = 2
even number ⇒ outside

Winding Number Method -



- Consider the point in question and point on polygon boundary.
- Give the direction numbers to each boundary line crossed, sum these direction numbers.
- If the addition is non-zero number is inside.



For P1 ⇒ -1 + 1 = 0
point is outside
For P1 ⇒ 1
point is inside

Polygon Filling -

- Highlighting all the pixels which lie inside the polygon with color.

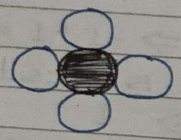
Seed Fill Algorithm -

- Start from a given "seed", a point known to be inside the polygon. Highlight outward from this point.

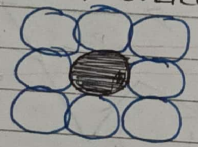
A. Boundary Fill / Edge Fill

- Starts with seed, a point inside the polygon.
- Examine the neighbouring pixels to check boundary pixel is reached or not.
- If not found continue the process.

4 - connected
Left, Right, Up, Down



8 - connected
Combination of two horizontal, vertical, diagonal



8-connected algorithm is more accurate.

F-colour ⇒ Fill colour
b-colour ⇒ Boundary colour

getpixel function gives the colour of specified pixel.
putpixel ⇒ draws with specified colour.

```
boundary_fill (x, y, f_colour, b_colour)
{
  if(getpixel(x,y) != b_colour &&
     getpixel(x,y) != f_colour)
  {
      putpixel (x, y, F_colour)
      boundary_fill (x+1, y, f_colour, b_colour)
      boundary_fill (x, y+1, F_colour, b_colour)
      boundary_fill (x-1, y, f_colour, b_colour)
      boundary_fill (x, y-1, F_colour, b_colour)
  }
}
```

4-connected Method

	x, y+1	
x-1, y	x, y	x+1, y
	x, y-1	

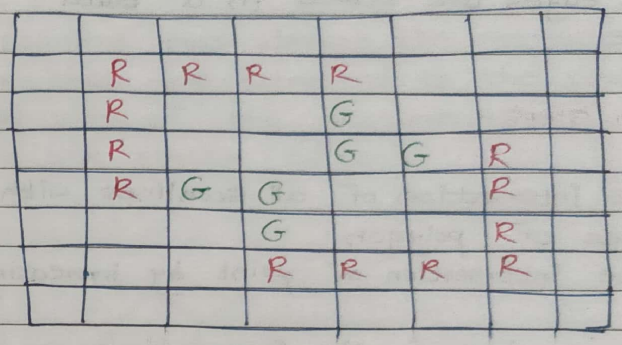
8-connected Method

x-1, y+1	x, y+1	x+1, y+1
x-1, y	x, y	x+1, y
x-1, y-1	x, y-1	x+1, y-1

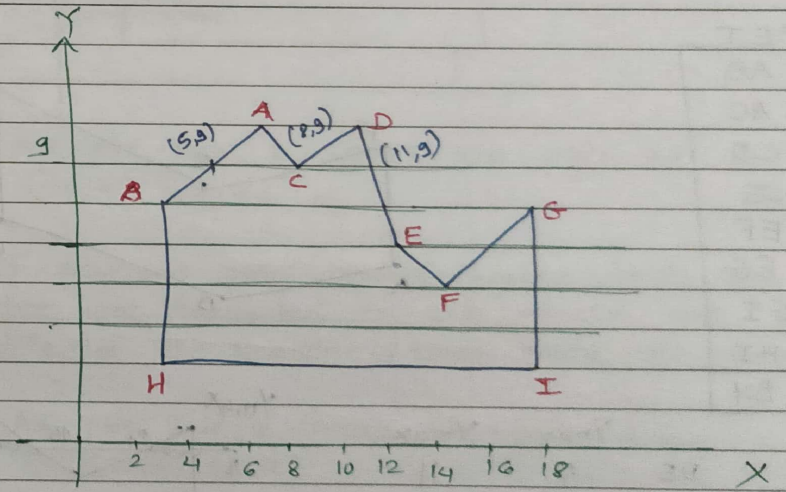
Flood Fill Algorithm -

- If such polygon has to be filled whose not defined in a single colour boundary.
- Fill areas by replacing a specified interior colour instead of searching for boundary colour.
- It is known as Flood Fill algorithm.
- Start with some seed and examine the neighbouring pixels, replace it with a new colour.

```
Flood-Fill (x,y,old-colour,new-colour)
{
    IF (getpixel (x,y) = old-colour)
    {
        putpixel (x,y,new-colour)
        Flood-Fill (x+1,y,old-colour,new-colour)
                   (x-1,y                      )
                   (x,y+1                      )
                   (x,y-1                      )
                   (x+1,y+1                    )
                   (x-1,y-1                    )
                   (x+1,y-1                    )
                   (x-1,y+1                    )
    }
}
```



Scan Line Algorithm



Active Edge Table (AET)

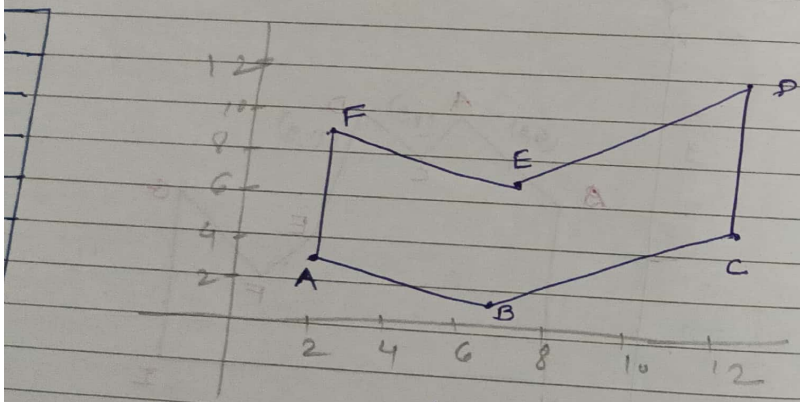
All the edges are stored in a data structure.

Algorithm Steps -

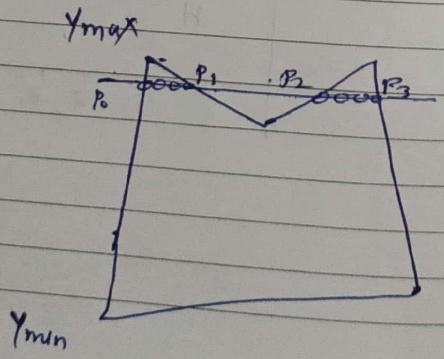
...d the intersection of all scanlines with ... edges of polygon.

...t the intersection of point by increasing ...er.

...e the pair of the intersection, ... color within all the pixels inside the pair.

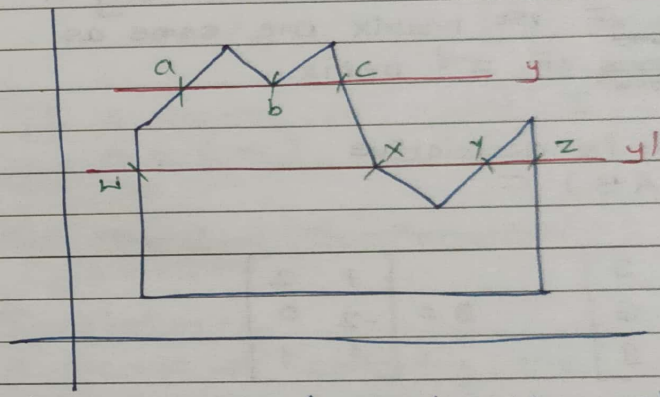


	Ymin	Ymax
E	7	11
F	7	9
D	5	11
	3	9
	1	5
	1	3



Rule :

1. If scanline pass through the vertex, both the edges that are connected to the vertex are on same side of the scanline then we need to count the vertex twice.



As per Rule 1, Make the pair as (a,b) (b,c).

2. If scanline pass through vertex, both the edges that are connected to the vertex are on opposi side of the scanline then count the vertex once

As per Rule 2, Make the pair as (W,X) (Y,Z)

2D Transformation

Matrix :
No. of rows & columns.

Matrix Multiplication.
Multiplication can be done if and only if no. of ~~rows~~ columns of 1st Matrix are same as no. of ~~columns~~ rows of 2nd matrix.

Multiplication is associative.
$A(BC) = (AB)C$

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 2 \\ -2 & 0 \\ 3 & 1 \end{bmatrix}$$

$$C(1,1) = A(1,1)B(1,1) + A(1,2)B(2,1) + A(1,3)B(3,1)$$
$$= (1)(1) + (2)(-2) + 3(3)$$
$$= 1 - 4 + 9$$
$$= 6$$

$$C = \begin{bmatrix} 6 & 5 \\ 12 & 14 \\ 18 & 23 \end{bmatrix}$$

Identity Matrix
Square matrix which has all the elements equal to 0 except the diagonal elements, which are 1, is known as Identity Matrix.

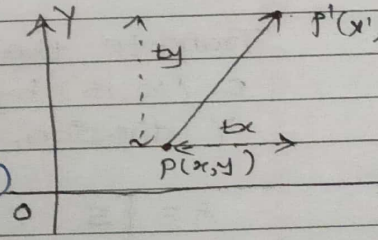
$$\begin{bmatrix} 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

It is denoted as matrix I.
$A = A * I$.

* Two Dimensional Transformation -

1. Translation :
- A process of changing the position of an object in a straight line path from one coordinate location to another.

$$x' = x + t_x$$
$$y' = y + t_y$$



The translation distance (t_x, t_y) is called a translation vector or shift vector.

$$P = \begin{bmatrix} x \\ y \end{bmatrix} \qquad P' = \begin{bmatrix} x' \\ y' \end{bmatrix} \qquad T = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$P' = P + T$$

Example

Translate a polygon with co-ordinates A(2,5), B(7,10), C(10,2) by 3 units in x direction and 4 units in y direction.

$A' = A + T$

$$A = \begin{bmatrix} 2 & 5 \\ 7 & 10 \\ 10 & 2 \end{bmatrix}, \quad T = \begin{bmatrix} 3 \\ 4 \end{bmatrix} \quad [3 \;\; 4]$$

$$= \begin{bmatrix} 2+3 & 5+4 \\ 7+3 & 10+4 \\ 10+3 & 2+4 \end{bmatrix}$$

$$= \begin{bmatrix} 5 & 9 \\ 10 & 14 \\ 13 & 6 \end{bmatrix}$$

Translate the polygon A(5,7), B(7,11) and C(12,15) by 4 units in X and 6 units in Y direction.

$$A = \begin{bmatrix} 5 & 7 \\ 7 & 11 \\ 12 & 15 \end{bmatrix}, \quad T = \begin{bmatrix} 4 & 6 \end{bmatrix}$$

$$A' = \begin{bmatrix} 9 & 13 \\ 11 & 17 \\ 16 & 21 \end{bmatrix}$$

Translate the polygon with coordinates A(0,0), B(0,4), C(4,4), D(4,0) by 2 units X and 3 units in y direction.

$$A = \begin{bmatrix} 0 & 0 \\ 0 & 4 \\ 4 & 4 \\ 4 & 0 \end{bmatrix} \quad T = \begin{bmatrix} 2 & 3 \end{bmatrix}$$

$$A' = \begin{bmatrix} 2 & 3 \\ 2 & 7 \\ 6 & 7 \\ 6 & 3 \end{bmatrix}$$

Generalized Bresenhams Algorithm -

1. Read the line end point (x_1, y_1) and (x_2, y_2) such that they are not equal.
2. $dx = |x_2 - x_1|$ and $dy = |y_2 - y_1|$
3. Initialize starting point
   $x = x_1$
   $y = y_1$
   plot (x, y)
4. $s1 = \text{sign}(x_2 - x_1)$
   $s2 = \text{sign}(y_2 - y_1)$
5. if $(dy > dx)$ then
   Exchange dx and dy
   Exchange = 1
   else
   Exchange = 0
   end if
6. $e = 2 * dy - dx$
7. $i = 1$
8. while $(e >= 0)$
   {
   if (Ex-change = 1) then
   $x = x + s1$
   else
   $y = y + s2$
   end if
   $e = e - 2 * dx$
   }

$m = \frac{dy}{dx}$

9. if (Ex-change == 1) then
   y = y + s2
   else
   x = x + s1
   end if
   e = e + 2 * dy
10. plot (x, y)
11. i = i + 1
12. if (i ≤ dx) then goto step 8.
13. stop.

Consider the line From (4,9) to (7,7)

$(x_1, y_1) = (4, 9)$
$(x_2, y_2) = (7, 7)$

$dx = |x_2 - x_1| = |7-4| = 3$
$dy = |y_2 - y_1| = |7-9| = 2$

$s1 = sign(x_2 - x_1) = 1$
$s2 = sign(y_2 - y_1) = -1$

as dy is not greater than dx
Ex_change = 0.

$e = 2 * dy - dx$
$= 2 * 2 - 3$
$= 1$

A) i = 1.
e = 1 ≥ 0
As Ex_change = 0
y = y + s2 ⇒ y = 9 − 1 = 8 → y = 8
e = e − 2 * dx ⇒ e = 1 − 2 * 3 = −5

x = x + s1 = 4 + 1 = 5. → x = 5
e = e + 2 * dy = −5 + 2 * 2 = −5 + 4 = −1.

plot (5,8)

B) i = 2
e = −1 ≥ 0 not satisfied & Ex_change = 0 not satisfied
value of y will not change → y = 8
x = x + s1 = 5 + 1 = 6 → x = 6
e = e + 2 * dy = −1 + 4 = 3

plot (6,8)

c) i = 3
e = 3 > 0
As Ex_change = 0
y = y + s2 ⇒ 8 − 1 = 7 → y = 7
e = e − 2 * dx ⇒ 3 − 2 * 4 = −5

x = x + s1 ⇒ 6 + 1 = 7 → x = 7
e = e + 2 * dy ⇒ −5 + 2 * 4 = −1

plot (7,7)

Consider the line from (2,7) to (5,5). Use Bresenham's line drawing algorithm to rasterize the line.

$(x_1, y_1) = (2,7)$
$(x_2, y_2) = (5,5)$

$\Delta x = |5-2| = 3$ $\quad s1 = 1$
$\Delta y = |5-7| = 2$ $\quad s2 = -1$

As $dx > dy$, Ex_change = 0

$e = 2 * dy - dx$
$= 2 * 2 - 3$
$= 1$

$i = 1$

i) $e \geq 0 \; ; \; 1 \geq 0$
Ex_change = 0
$y = y + s2$
$= 7 + (-1)$
$y = 6$ $\quad$ y=6

$e = e - 2 * dx$
$= 1 - 2 * 3$
$= 1 - 6$
$e = -5$

$x = x + s1$
$= 2 + 1$
$x = 3$ $\quad$ x=3