$= 30 + 10 \times \frac{1}{2} = 35$

$y_0 = y_{vmin} + (y_w - y_{wmin}) S_y$

$= 40 + (80 - 40) S_y$

$= 40 + 40 S_y = 40 + 40 \times \frac{1}{2} = 60$

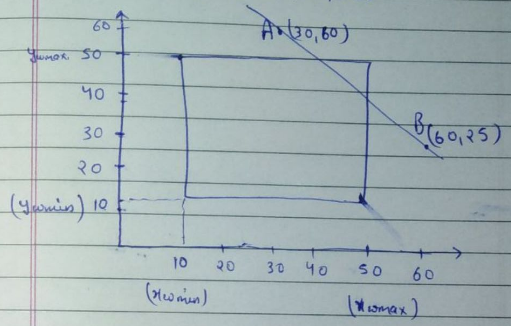
$(x_0, y_0) = (35, 60)$

* Line clipping -

⊙ Algorithm (cohen sutherland) → (TBRL)

1) Assign a region code for each endpoint.
2) If both end points have a region code 0000 then accepts this line.
3) Else perform the logical AND operation for both region code.
   - i) If result is not 0000, reject.
   - ii) Else you need clipping.
     - a) choose an end point of the line that is outside the window.
     - b) Find the intersection point at the window boundary based on region code.
     - c) Replace end point with the intersection point and update the region code.
     - d) Repeat step 2 until we find a clip line either partially accepted or rejected.
4) Repeat step 1 for other lines

for ex - Apply when cohen sutherland line algo. to clip the line segment with coordinates (30,60) (60,25) against the window (10,10) & (50,50)



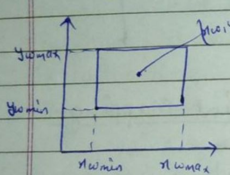
	Endpoint	AND
A	1000	0000
B	0010	

This is partially visible.

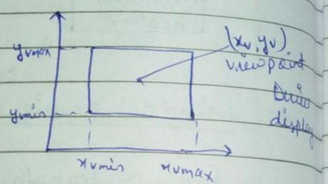
13/02/2020

Point clipping –

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



world coordinate



viewing coordinates

~~world coordinate~~

Normalised point on window –

$$\rightarrow \frac{x_w - x_{wmin}}{x_{wmax} - x_{wmin}}, \quad \frac{y_w - y_{wmin}}{y_{wmax} - y_{wmin}}$$

Normalised point at viewport –

$$\Rightarrow \frac{x_v - x_{vmin}}{x_{vmax} - x_{vmin}}, \quad \frac{y_v - y_{vmin}}{y_{vmax} - y_{vmin}}$$

Now the relative portion of the object in the viewport are same.

for x coordinate,

$$\frac{x_w - x_{wmin}}{x_{wmax} - x_{wmin}} = \frac{x_v - x_{vmin}}{x_{vmax} - x_{vmin}}$$

For y coordinate,

$$\frac{y_w - y_{wmin}}{y_{wmax} - y_{wmin}} = \frac{y_v - y_{vmin}}{y_{vmax} - y_{vmin}}$$

So after calculating for x, y coordinates, we get

$$x_v = x_{vmin} + (x_w - x_{wmin})$$
$$y_w = y_{vmin} + (y_w - y_{wmin})$$

$$S_x = \frac{x_{vmax} - x_{vmin}}{x_{wmax} - x_{wmin}}$$

$$S_y = \frac{y_{vmax} - y_{vmin}}{y_{wmax} - \cancel{y_{min}}\, y_{wmin}}$$

S_x is the scaling factor of x
S_y is the scaling factor of y

Ques: window viewport $x_{wmin} = 20$, $x_{wmax} = 80$, $y_{wmin} = 40$, $y_{wmax} = 80$.
viewport $x_{vmin} = 30$ $x_{vmax} = 60$, $y_{vmin} = 40$, $y_{vmax} = 60$

Now in point (x_w, y_w) be $(30, 80)$ on the window you have to calculate that point or viewport.

Ans.

$$S_x = \frac{60-30}{80-20} = \frac{30}{60} = \frac{1}{2}$$

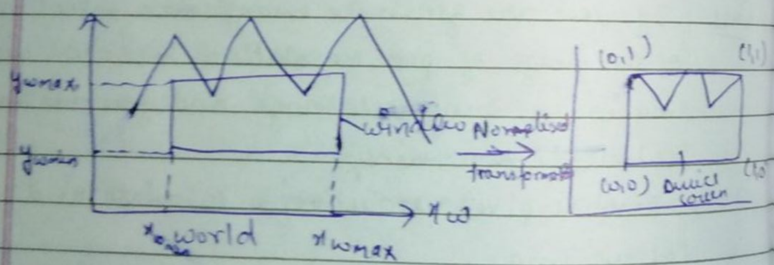
$$S_y = \frac{60-40}{80-40} = \frac{20}{40} = \frac{1}{2}$$

$$X_0 = X_{vmin} + (x_w - x_{wmin}) S_x$$
$$= 30 + (30-20) S_x$$
$$= 30 + 10 S_x$$

The viewing pipeline –

A world coordinate area selected for display is called a window. The window defines what to be viewed. An area on a display device to which a window is mapped is called a view port.

The view port defines where it is to be displayed.
The mapping of a part of a world coordinate scene to the device coordinates is referred to as viewing transformation. Sometimes the 2D viewing transformation simply referred to as the window-to-view port transformation or the windowing transformation.



The 2D viewing transformation –

MC (modelling coordinates) → [Construct world coordinate scene using modelling coordinate Transformation] $\xrightarrow{WC}$ (world) → [convert world coordinates to viewing Coordinates] $\xrightarrow{VC}$ (viewing coordinates) → [map viewing coordinates to normalised viewing coordinates using window view point specification] $\xrightarrow{NVC}$ (Normalised) → [Map normalised viewpoint to Device Coordinates] $\xrightarrow{DV}$ (Device coordinates)

* Window to viewpoint Mapping –

formula transformation :

$$\frac{x_v - x_{vmin}}{x_{vmax} - x_{vmin}} = \frac{x_w - x_{wmin}}{x_{wmax} - x_{wmin}}$$

$$\frac{y_v - y_{vmin}}{y_{vmax} - y_{vmin}} = \frac{y_w - y_{wmin}}{y_{wmax} - y_{wmin}}$$

$$x_v - x_{vmin} = \left(\frac{x_w - x_{wmin}}{x_{wmax} - x_{wmin}}\right) \times (x_{vmax} - x_{vmin})$$

$$x_v - x_{vmin} = \underbrace{S_x}_{\text{scaling factor}} \times (x_{vmax} - x_{vmin})$$

This process is called windowing.

The major task of display processor is digitising a picture definition given in an application program into a set of pixel intensity value for storage in the frame buffer. This digitisation process is called scan conversion.

* <u>Resolution</u>–

Resolution refers to the sharpness & clarity of an image. The term is used to describe monitor or printer. In case of dot matrix & laser printer resolution indicates no of dots/inch.
for ex– 300 DPI

In case of graphics monitor, the screen resolution signifies no. of pixels on the entire screen or max. points that can be displayed without overlap on a CRT 640 by 480 [640 x 480]. Pixel screen is capable of displaying 640 distinct pixels on each of 480 lines or about 3 lakh pixels.

Printers, monitors, scanner another I/O device are classified as high, low, medium resolution.

If image has m rows and n columns, resolution is defined as m×n.

* <u>Aspect Ratio</u>–

The no gives the ratio of vertical points to horizontal points necessary to produce equal length line in both direction on the screen.

Aspect ratio of 3 by 4 means that a vertical line plotted with 3 points has the same length as a horizontal line plotted with 4 points.

When resizing graphics it is imp to maintain the aspect ratio to avoid stretching the graphics out of proportion.
The resolution of 800 by 600 has an aspect ratio of 4:3

<u>Ques.</u> If you are given an image with aspect ratio 6:2 of an image of pixel resolution 480000. Given the img is an grey scale image and you are asked to calculate two things

i) Resolve pixel resolution to calculate the dimension of an img.

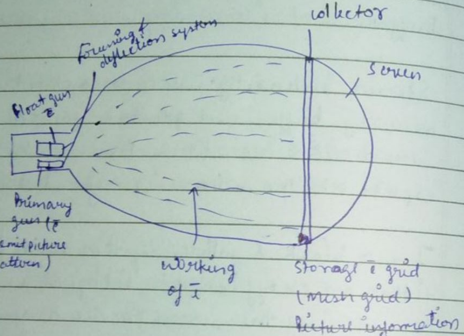
ii) Calculate the size of an image.

i)
$$R = m \times n$$
$$480000 = m \times n$$
$$\frac{m}{n} = \frac{\cancel{6}\,3}{\cancel{2}\,1} = \frac{3}{1} \qquad 480000 = 3n^2$$
$$n^2 = 160000$$

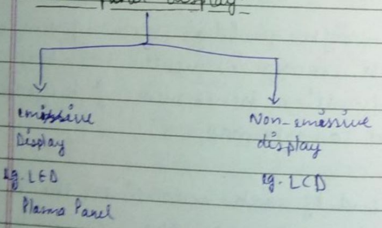
Row $m = 3n = 1200$ column $n = 400$

ii) Size = Row x column x 8
= 1200 x 400 x 8 = 3840000 bits
~~3840~~ = 480000 byte

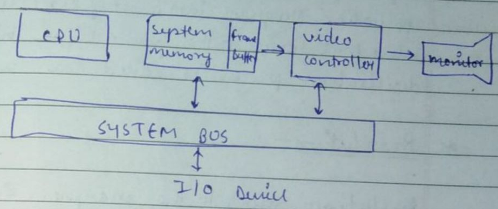
DVST (Direct view storage tube)-



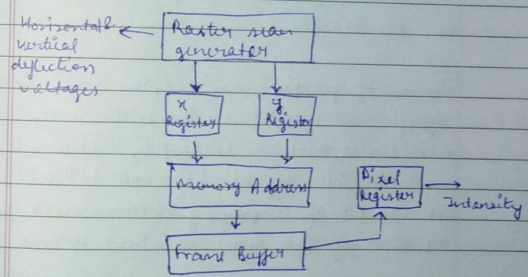
Flat panel display-

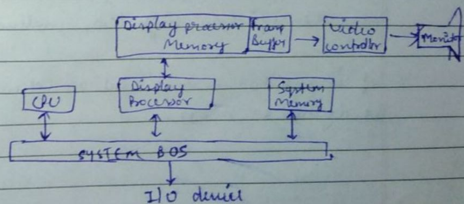


* Architecture of simple raster graphic system



* Basic video controller refresh operation-





High Resolution

- Resolution of random scan display is higher.
- Electron beam falls on those part of the screen where a picture is to be drawn.

Disadvantages-

- They don't produces real & shadow image.
- Different colours are not possible with this approach.

★ Raster scan display -

That is commonly used in monitors. In Raster scan system, the E beam is swepes across the screen horizontaly from left to right at a time from top to bottom.

At the end of scan line, the E beam turned to the left corner of the screen to start the next line, next scanned line. This is called horizontal retrace

At the end of each frame, the E beam go to the top left corner to continue the next frame. This is called vertical retrace

- The beam intensity is turned on & off to create a pattern of pixels. Picture definition is stored in memory space is called memory buffer

- Stored intensity, values extract from the memory buffer and ~~printed~~ painted on the screen one row at a time. Each illuminated spot on the screen is referred as the pixel.

- In a black & white system each ~~picture~~ pixel is either on or off, so one bit / pixel is needed for storing the screen position so the memory buffer is called a bit map.

- But in a high quality color system several megabyte storage is needed to store the picture definition hence memory buffer is a referred as pix map.

- At the end of each scan line, the E beam returns to the left corner of the screen & starts scanning for the next line and at the end of each frame, the E beam returns to the top of the left corner.

- The random scan display, ē beam directed only that part of the screen :- where the picture is to be drawn.

- Random scan monitors draw a picture one line at a time and for this reason also referred to as vector display or stroke writing or calligraphic displays. The component lines of a picture can be drawn and refreshed by a random scan system in any specified order

- In random display picture definition is stored as the line drawing command in an area of memory knlas refresh buffer or refresh display file or display list or display program

- The example of random display is pen plotter.

- In the above picture the typical random display architecture.

- It consist of display controller central processing unit, display buffer memory & a cathode ray tube.

- The display buffer memory store the computer produce display list or display program. The program contains points & line ~~pointer~~ plotting command with end point.

- The display controller interprets commands for plotting points, lines and characteristic & sends digital and point coordinates to a vector generator.

- The vector generator converts the digital coordinates values ~~of~~ to analog voltages for beam deflection circuits that displays an ē beam writing on the CRT phosphorous coating

- Random scan system are designed for line drawing applications and can't display realistic shaded scenes. displays such as plasma panel supports only line drawing.

Advantages -

- Vector display produce smooth line drawing because the CRT being directly follows the line path.
- Used by both analog & digital computers

~~Que~~ $\cancel{r_x = 8}$

$\cancel{r_y = 6}$

$P_{2k+1} - P_k =$

$P_{2k+1} = P_{2k} + r_x^2 - 2r_x^2(y_x - 1) + r_y^2(x^2_{k+1} - r_x^2) + r_y^2($
$(x_{k+1} - x_k)$

Region 1 : $m < 0$

$P_k \geq 0$

Next coordinate $(x_k + 1, y_k - 1)$

$P_{1k} < 0$

$N.C. = (x_k + 1, y_k)$

Stop $2r_y^2$ ~~x~~ $x_{k+1} \geq 2x^2 + y_k + 1$

Region 2 $m > 1$

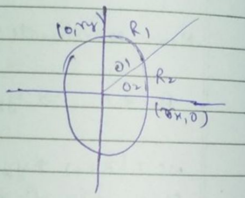
5/02/2020

★ Random Scan display / Raster scan display -

(3,7)
(4,7)
(5,6)
(6,5)
(7,4)
(7,5)

☆ Midpoint ellipse Drawing Algorithm-

eq.ⁿ of ellipse



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

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

$$x^2r_y^2+y^2r_x^2-r_x^2r_y^2=0$$

$$dy=2yr_x^2$$

$$dx=2xr_y^2$$

Octant 1:

$m<1$ — x - unit interval
y - y_k or y_{k-1}

(x_{k+1}, y_k) (x_{k+1}, y_{k-1})

midpoint $(x_{k+1}, y_k-\frac{1}{2})$

Apply midpoint in eq. (1) -

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

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

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

$P_{k+1}-P_k$

$$P_{k+1}=P_k+r_y^2+2(x_{k+1})r_y^2+r_x^2\left[(y_{k+1}^2-y_k^2)-(y_{k+1}-y_k)\right]$$

Initial decision parameter
$(0, r_y)$ in P_k

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

$m<1$

$$P_k=r_y^2+r_x^2-r_yr_x^2$$

if $P_k\geq 0$ (x_k+1, y_k+1)

If $P<0$ (x_k+1, y_k)

$m>1$

y - unit interval
x - x_k or x_{k+1}

midpoint $(x_k+\frac{1}{2}, y_k-1)$

Apply midpoint in eq. (1) -

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

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

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

✱ Mid point circle algorithm-

x - unit interval

y = ?

y_k or $y_k - 1$

(x_{k+1}, y_k) or (x_{k+1}, y_{k-1})

midpoint $= \left(\frac{x_{k+1} + x_{k+1}}{2} , \frac{y_k + y_{k-1}}{2} \right)$

✱ Circle formula-

$$x^2 + y^2 = r^2$$

mid $x_{k+1}, y_k - \frac{1}{2}$

$$P_k = (x_k + 1)^2 + \left(y_k - \frac{1}{2}\right)^2 - r^2$$

$$P_{k+1} = (x_{k+1} + 1)^2 + \left(y_{k+1} - \frac{1}{2}\right)^2 - r^2$$

$$P_{k+1} - P_k = \left((x_k + 1) + 1)^2 + \left(y_{k+1} - \frac{1}{2}\right)^2\right) - (x_k + 1)^2 - \left(y_k - \frac{1}{2}\right)^2$$

$$= (x_k + 1)^2 + 1 + 2(x_k + 1) + y_{k+1}^2 + \frac{1}{4} - y_{k+1} - (x_k + 1)^2 - y_k^2 - \frac{1}{4} + y_k$$

$$P_{k+1} - P_k = 2(x_k + 1) + (y_{k+1}^2 - y_k^2) - (y_{k+1} - y_k) + 1$$

$$[P_{k+1} = P_k + 2(x_k + 1) + (y_{k+1}^2 - y_k^2) - (y_{k+1} - y_k) + 1$$

$$P_k = (0 + 1)^2 + \left(r - \frac{1}{2}\right)^2 - r^2$$

$$= \frac{5}{4} - r$$

$$= 1 - r \quad \text{[Taking integer]}$$

$$\boxed{P_0 = 1 - r}$$

$P_k \geq 0 \Rightarrow y_{k+1} = y_k - 1$

Next coordinate: (x_{k+1}, y_{k-1})

$P_k < 0 \Rightarrow y_{k+1} = y_k$

NC $= (x_{k+1}, y_k)$

Stop when $(x \geq y)$

Ques. $r = 8$

$P_0 = 1 - r = 1 - 8 = -7$

Iterative:

Initial point → $(0, r)$

Iteration	(x_k, y_k)	decision	x_{k+1}, y_{k+1}
	(0, 8)	−7	(1, 8)

$P_{k+1} = -7 + 2(0+1) + (0) - (0) + 1 = -4$

Iteration	(x_k, y_k)	decision	x_{k+1}, y_{k+1}
	(1, 8)	−4	(2, 8)

$P_{k+1} = -4 + 2(1+1) + 0 - 0 + 1 = 1$

Iteration	(x_k, y_k)	decision	x_{k+1}, y_{k+1}
	(2, 8)	1	(3, 7)

Ques. (35, 40) (43, 45)

Initial decision parameter

	Initial decision parameter	Initial coordinates (x_k, y_k)	P_k	x_{k+1}, y_{k+1}
	~~Iwation~~ $k=0$	35, 40	2	(36, 41)
$k=1$	$P_{k+1} = 2+10-16(1) = -4$	36, 41	-4	(37, 41)
$k=2$	$P_{k+1} = -4+10-16(0) = 6$	37, 41	6	(38, 42)
$k=3$	$P_{k+1} = 6+10$			

Bresenham's Algo. for circle drawing –

i) Input the two line endpoints and store the left end point in (x_0, y_0)

ii) load (x_0, y_0) into the frame buffer i.e. plot the first point.

iii) Calculate constants Δx, Δy, $2\Delta y$ and $2\Delta y = 2\Delta x$ and obtain the starting value for the decision parameter as $P_0 = 2\Delta y - \Delta x$

iv) At each x_k along the line, starting at $k=0$, perform the following test

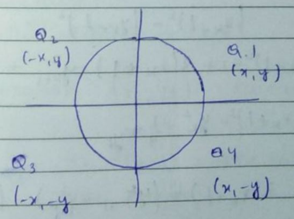
if $P_k < 0$, the next point to plot is (x_{k+1}, y_k).

$P_{k+1} = P_k + 2\Delta y$

Otherwise, the point to plot is (x_{k+1}, y_{k+1})

$P_{k+1} = P_k + 2\Delta y - 2\Delta x$

v) Repeat step 4 Δx times.



$d_1 - d_2 = 2m(x_k + 1) - 2y_k + 2c - 1$

$\Delta x(d_1 - d_2) = \Delta x\left[2\frac{dy}{dx}(x_k+1) - 2y_k + 2c - 1\right]$

$P_k = 2\Delta y x_k + 2\Delta y - 2\Delta x y_k + 2\Delta x c - \Delta x$

PK decision Parameter —

$P_{k+1} = 2\Delta y x_{k+1} + 2\Delta y - 2\Delta x y_{k+1} + \Delta x(2c-1)$

$P_{k+1} - P_k = 2\Delta y x_{k+1} + 2\Delta y - 2\Delta x y_k + \Delta x(2c-1)$

$\cancel{2\Delta y(x_{k+1} - x_k) - 2\Delta x(y}$

$-(2\Delta y x_k + 2\Delta y - 2\Delta x y_k + 2\Delta x c - \Delta x)$

$= 2\Delta y(x_{k+1} - x_k) - 2\Delta x(y_{k+1} - y_k)$

m<1

$P_{k+1} = P_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k)$

Initial (x_k, y_k)

$P_k = 2\Delta y x_k + 2\Delta y - 2\Delta x y_k + \Delta x(2c-1)$

$y = mx + c$

$c = y - mx$

$= 2\Delta y x_k + 2\Delta y - 2\Delta x y_k + \Delta x\left(2y - \frac{2\Delta y}{\Delta x} - 1\right)$

$= 2\Delta y - \Delta x$

$P_{k+1} = P_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k)$

If $P_k \geq 0$	If $P_k < 0$
$x_{k+1} = x_k + 1$	$x_{k+1} = x_{k+1}$
$y_{k+1} = y_k + 1$	$y_{k+1} = y_k$
Next coordinate: $y_{k+1} = y_{k+1}$	(x_{k+1}, y_k)
(x_{k+1}, y_{k+1})	

* If m<1 —

$P_k = 2\Delta y - \Delta x$

$P_{k+1} = P_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k)$

* If m>1 —

$P_k = 2\Delta x - \Delta y$

$P_{k+1} = P_k + 2\Delta y - 2\Delta x(x_{k+1} - x_k)$

If $P_k \geq 0$	If $P_k < 0$
$y_{k+1} = y_{k+1}$	$y_{k+1} = y_k + 1$
$x_{k+1} = x_k + 1$	$x_{k+1} = x_k$

① ②

x	9	8	7	6	5	4	3	2	1	0
y	7	6	5	4	3	2	1	0	0	0

② $x = x + \Delta x = 9 - 1 = 8$

$y = y + \Delta y = 7 - 0.8 = 6.2$

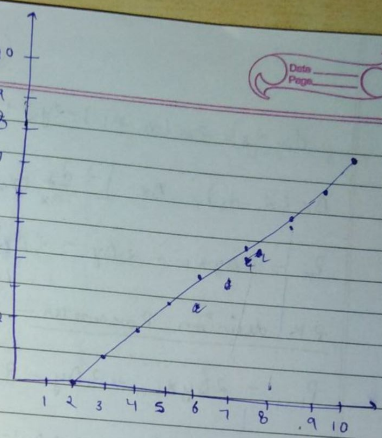
③ $x = 8 - 1 = 7$

$y = 6 - 0.8 = 5.2$

④ $x = 7 - 1 = 6$

$y = 5 - 0.8 = 4.2$

Iteration	(round x, round y) Plotted point	Incremental value ($x+\Delta x$, $y+\Delta y$) (before round x, round y)
0	(10, 8)	10-1=9, 8-0.8 = 7.2
1	(9, 7)	9-1=8, 7-0.8 = 6.2
2	(8, 6)	8-1=7, 6-0.8 = 5.2
3	(7, 5)	7-1=6, 5-0.8 = 4.2
4	(6, 4)	6-1=5, 4-0.8 = 3.2
5	(5, 3)	5-1=4, 3-0.8 = 2.2
6	(4, 2)	4-1=3, 2-0.8 = 1.2
7	(3, 1)	3-1=2, 1-0.8 = 0.2
8	(2, 0)	2-1=1, 0
9	(1, 0)	1-1=0, 0
10	(0, 0)	0-1=-1, 0



Bresenham's line drawing algorithm-

→ The user of an interactive computer graphics program specifies what classes of data items or objects are to be generated and represented pictorically.

→ Most task of the programmer concern creating and editing the model and handling user interaction.

DDA (Differential digital analyser –

Step 1: Accept start and end point coordinates (x_1, y_1) & (x_2, y_2)

Step 2: Calculate $dx = x_2 - x_1$
$dy = y_2 - y_1$

Step 3: If abs(dx) > abs(dy)
then k = abs(dx)
else
k = abs(dy)

Step 4: Calculate $\Delta x = \frac{dx}{k}$, $\Delta y = \frac{dy}{k}$

Step 5: Initialize $x = x_1$, $y = y_1$

Step 6: Display pixel (x, y)

Step 7: $x = x + \Delta x$
$y = y + \Delta y$ // next point

Step 8: Display pixel round(x), round(y)

Step 9: Repeat step 7, step 8 k times

$x_1 = 10$ $x_2 = 0$
$y_1 = 8$ $y_2 = 0$

$dx = x_2 - x_1 = -10$
$dy = -8$
abs(dx) = 10
abs(dy) = 8
Here abs(dx) > abs(dy)

⇒ k = abs(dx) = 10

$\Delta x = \frac{-10}{10} = -1$ $\Delta y = \frac{-8}{10} = -0.8$

(5) $x = 10$
$y = 8$

(7) ⇒ $x = x + \Delta x = 10 - 1 = 9$
$y = y + \Delta y = 8 - 0.8 = \cancel{2.8}\ 7.2$

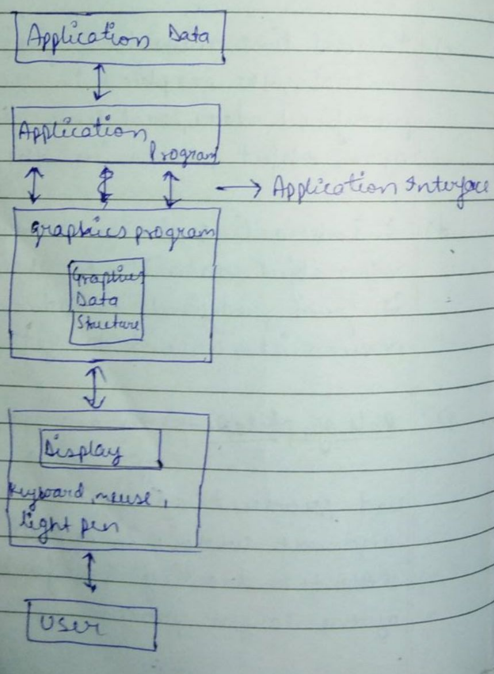
(8) $x = 9$, $y = 7$

4) Logical or temporal relationship-

The relationship between objects and their pictures may be logical or temporal.

Graphics system configuration-
(conceptual framework of interactive graphics)

1) Application system
2) Graphic system
3) User (operator, client, system server)



→ The configuration consist of three main components

- Application
- Graphics
- User

→ Hardware includes i/p and o/p devices for interaction and display respectively. The software has three components

- Application Data
- Application program ~~C97~~
- Graphic system

① It creates stores and retrieve the data toard objects to be pictured on the screen

② The application program helps in producing images by sending a series of graphics output command to the 3rd component the graphics system.

→ The graphics system interacts between the user and the application program and is responsible for actually producing the pictures from the detailed description & for passing the user i/p to the ~~data~~ Application program for processing determines the user degree of control over images. Examples are offline floating, interactive floating, real time animation graphics and interactive design.

a) Offline floating is done with a predefined DB produced by other application programs or digitized from physical model.

b) In interactive floating the user controls graphics by changing the parameters of the objects.

c) In real time animation graphics, the user controls the graphics in real time with predefined data, real time data or perception of the object status in the next movement.

d) In interactive design, the user defines a new object on a blank sheet by assembling it from predefined components and then moves it around to get a desired view

3) **Role of picture** - In one application, the graphics end is the end product as in case of animation and art work. In another application like CADD the drawing is just a small part of the larger process.

Types of computer graphics-

Interactive graphics (Active) [Two-way communication]

Non-interactive graphics (Passive) [one way communication (observer has ~~as~~ no control on Image)

Basic components of Interactive graphics system -

1. Input device
2. Processing
3. Display / output

Classification of applications-

~~for the~~ All the CG applications may be categorized into the following 4 groups according to the manner of representation-

1) <u>Dimensional</u> :- Images of objects can be represented in either 3D or 2D form.

2) <u>Types of interaction</u>:- The type of interaction

20/01/2020

Date ____
Page ____

Unit : 01

Computer graphics is the art of drawing lines, circles, chart etc. with the help of any programming language.

Application-

- Computer aided design
- Presentation graphics
- Computer art
- Entertainment
- Education & training
- visualization
- Image processing
- Cathode ray tube