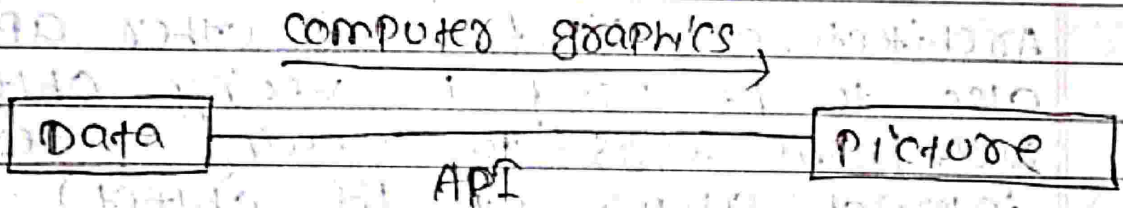
Computer Graphics :- Unit-1

On traditional base, the computer system perform only textual manipulation as the every end user can't be understand all technique term of computer, so that in this case some graphical notation to be initiated for end user convinient data manipulation.

Definition :-

It is a Method, technique or mathematics through which every types of data to be converted any pictorial representation. That means when this concept to be embeded with any application programming interface language then it provides for graphical data manipulation

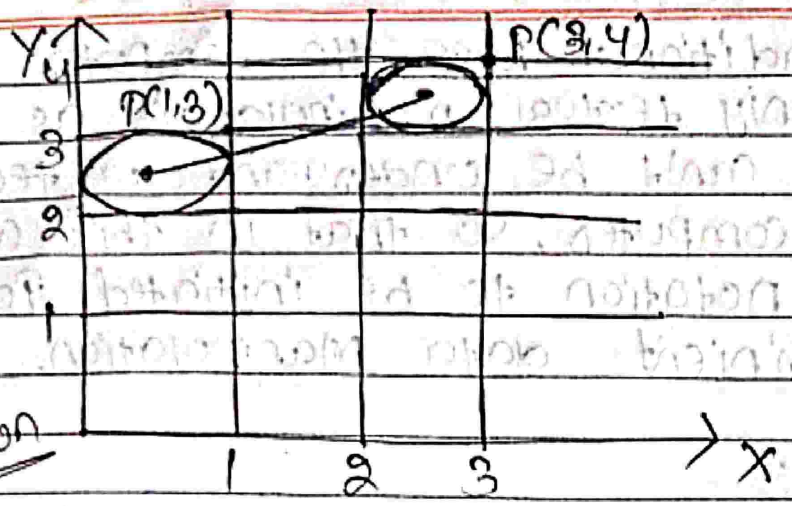
* Here every picture to be build up by consisting discrete no of pixel points (countable) or (dot) point



- Each pixel of the graphics to be generally stored in rectangular or square format box is called as frame buffer.

Ex

Let us take to draw a line by addressing pixel point from P(1,3) to P(3,4)



long question

Application of compues Graphics :-

. The computer Graphics, now a days to be used in different Areas, Here some Areas are given below.

① CAD (computer Aided de ign) :-

- In general way, it is use for design processes, It specialy used in Engineering & Architectural field, some other application also to be used for viewing object in wireframe ways (for viewing External & internal schema of the object)
- It is use for Live Animation Creation (vehicle motion behaviour)
- It provides for object rendering mechanism in 2D & 3D format
- In case of computer system, it specially used for logical circuit maping from one device to another

Ex:

Autocard software

② PRESENTATION:-

- for representing any illustration report on format of slide presentation through projection.
- It used Bargraph, Piegraph, dotted graph etc. for report representation.
- It provides one counter through which the presentation to be momentum from one slide to another.
- It view of windows appearance in 2D or 3D format.

③ Computer art:-

- This interface to be used in case of commercial art and as well as in fine-art.
- It used one pen plotter (input device), whose automatically drawn an object using some mathematical function or asthetic values

④ Entertainment:-

Now a days, every types of Entertainment application (like Audio, video Gaming etc) to be used graphical concepts for build of inter active object
- It widely used for movie application because one Animation slod to be created using 24 frame (who se concatinate per second)

⑤ Visualization :-

The art of calculation mathematical and scientifical data in visual format (with & also specifying its internal behaviour that called visualisation.

⑥ Education & Training :-

- The computer graphics to be used as an educational aids for convinent lesson interpretation within very certain time interval
- Different types of scientifical training educational training, Technical training etc to be used some graphical user interface.
- This type of interface to be specially used for DUMP people

⑦ Image Processing :-

for Modify the object appearance, size, color in different views format of any existing interpreting object

(8) GUI (Graphical user Interface) :-

Now a days, some software packages to be provide graphical user interface through which the end user to be easily interact with this software.

- Initially the windows OS to be embedded GUI concept for providing multiple windows with different graphical icons.

Graphic I/P & O/P devices :-

Graphic O/P Device :-

Those devices view of all the internal logical interpreting data i.e, called output device & also this devices to be used for computer write purpose. that means by using this devices, the end user to be read the data & the system to be write the data.

- The given devices to be available now a days in two ways

(1) soft copy output device (soft disk)
(2) Hard copy output device. (hard disk)

(1) Soft copy output device :-

Those devices to be viewing all the logical interpreting data in visual formed i.e, called soft copy output device.

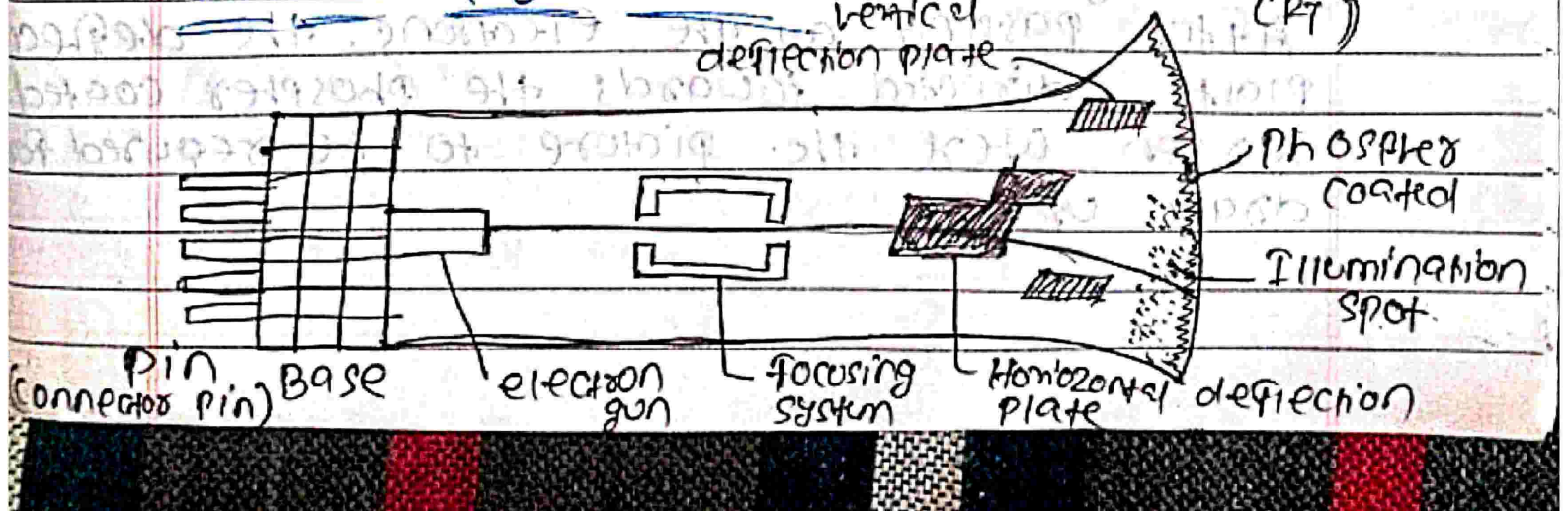
Ex :- Video display unit (VDU)

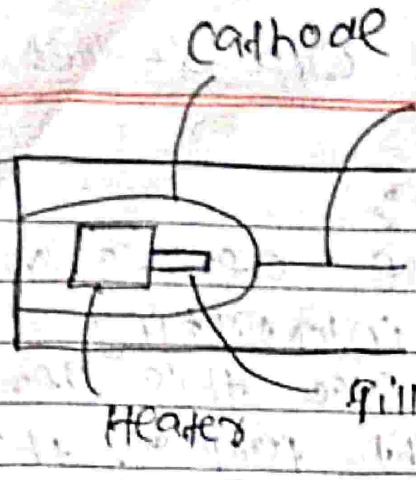
Visual display unit :-

It is the primary O/P graphics devices whose view of all the interpreting data above monitor screen

- The monitor data processing operation to be performed by using one technique i.e, CRT (Cathode Rays Tube)

Cathode Rays Tube :- (Fig. of Monochromatic CRT)





- The Cathode Rays Tube to be use for converting an electrical energy into the form of visual
- It looks like vacuum glass Tube with one display screen (which is coated with Phosphor material) & the other end it connect with no of connector pins. The Phospher coated screen to be emits some light for few time period, when the electron beam to be scattered with screen

Basic Operation of CRT :-

The internal layout of CRT to be contain no of parameters through which is processing different visual data.

Electron gun :-

(Cathode)

A beam of electrons to be emitted by electron gun & passing through focusing system or deflection plate. After passing all the electrons, the deflection plate directed towards the phospher coated screen where the picture to be required for drawn up.

The electron gun contains 2 parameter (component) i.e. heated metal cathode & control grid. The heater to be heating by electricity and after that to be embedding with metal. Once all the metals to be consume electricity upto 45V, then it emitted no of electrons.

Here, the control grid to be used for controlling the intensity of electron beams A high voltage negative to be shutdown the electron beams

Focusing System:-

The focusing system to be used for force to converge all the electrons towards the phosphor coated screen (where the object to be redrawn).

The focusing system is accomplished by either electric or magnetic field.

- High precission focusing system to be also used for directing all the electrons towards phosphor coated screen in fully

Deflection plate:-

The Deflection plate to be used for providing margin boundaries of the object above phosphor coated screen

- Here it used 2 deflection plate for providing pixer margin location, where it to be required plot up.

- The 1st Horizontal deflection plate provi marginal value using left & right attribu
- The 2nd deflection plate i.e. vertical plate which providing marginal value using top & bottom approach.

Phospeo coated screen:-

The phospeo coated screen, then embedo with low peosistance then it wewing son animation type of objeat.
- When the phospeo coated screen provides high peosistance property then it wewing diffeend types of images.

PICTURE RESOLUTION:-

The pi'cture Resolution i's high, then it calculate all the electrons have satisfy ore properties i.e. non over lapping.

The by defaud pixel Resolution of ary graphi'cs i.e. 1280 x1024

ASPECT RATIO:-

The Ratio bet vertical deflection plate value with respect horizontal deflection plate value in equal quantity

Conclusion:-

When the phospeo coated scren to be scattered by electron beam, then it instantly emitted sone light through which sone illumination small spot to be created above monitor screen.

After that by, concatinating all the illumination small spot point for build of ore object.

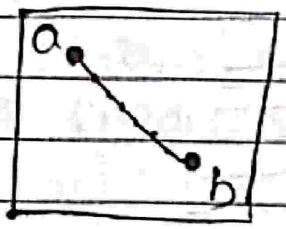
Technique of CRT:-

- for concatinating all the illumination spot on the format of real object. Here it provides 2 technique
  (1) vector scan or Random scan
  (2) Raster scan

(1) Vector scan or Random scan:

- Here all the no of electrons to be already accelerated towards the specified object position above the screen
- It always draw a line at a time in unidirectional format, so that it is also known as vector scan.
- As, here all the electrons to be only accelerated towards object position so here picture Resolution is very high
- It generally used for drawing any line, curve or any geomatrical figure.

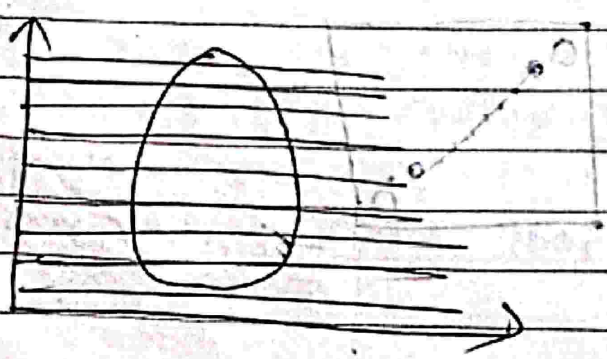
Ex
(1) Line drawing application i.e. Pen plotter.



(2) Raster Scan System :-

- Here, the electron beams are swipt across the whole screen of Monitor at a time
- on by default, it scanning all the data value Row by row from left to right
- It used one frame buffer or Refresh Buffer for storing pixel value of the previous line.
- Here, the intensity of the electron beam is turned on & turned off (Due to row by row data scanning)
- once here one row to be scanned up then the next row to be mapping from left to right using some illumination shade spot of the previous row line. (when the intensity of electron beams are turned off).
- one all over the row to be scanned up then the given object to be view of or drawn up from top to bottom
- It specially used in case of television view of, any realistic live application view of or any geomatrical area application view of.

Difference Betn Random Scan & Raster Scan :-



	Random scan	Raster scan
①	Here, the electron beams are only reflected towards the specified object pos^n	① But here the electron beams are swipt across the whole screen of the viewer
②	Here the line is not jiggered & the curves is very smooth	② But here the line is jiggred and the curves is not smooth.
③	As here, the electron beam only reflected towards object pos^n so the picture Resolution is very high	③ But here the Resolution picture value is less as compared to random scan
④	It specially used for geomatrical function data manipulation on format of graphical, for ex:- Pen plotter	④ It specially used for Real animation using any Realistic live application. for ex- Television.

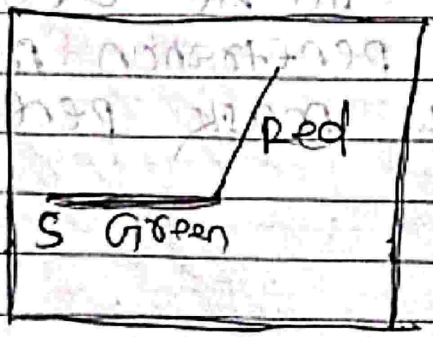
COLOR CRT :-

By default, initially all CRT to be viewing every object in black & white color values. But, now a days some color CRT to be evolved using 3 color values i.e. RGB (Red, Green, Blue).

- This color CRT to be design in 2 ways i.e. (1) Beam penetration method
  (2) Shadow mask peretration method

① Beam Penitration Method :-

This Technique to be used in case of Random Scan CRT. In this case the monitor screen coated with Phospher element through 2 layer color values i.e Red & Green
- The outer layer of the Phospher coated screen always initialised one color value i.e green (3). But the inner layer of the Phospher coated screen always initialised one color value i.e Red (4)
- Here the Beam Penitration Method to be used for Providing color object. When the electron Beam to be accelerated towards Phospher coated screen in very Ist, Then it viewing the given object in green color But when the intensity of electron Beam voltage is low then it instantly viewing the object in Red color format
- When the electron Beam to be Accelerate towards Phospher coated screen in very slowly then the object to be viewing in Red color format.
- By using the given 2 color, it also create 2 other color i.e, orange & yellow
- It specially used for viewing for any Geomatrical object or educational aids in colorful format



LASER - Light Amplification Stimulation Emition of Radiation

(2) Shadow Mask Method :-

- This Technique to be applied, when any object to be view of in color format using Raster Scan mechanism.
- Here the Phosper coated screen to be coating by using 3 color values i.e. RGB (Red, Green, Blue)
- In this case, when the electron beam to be accelerate towards Phosper coated screen then the Phosper emits light through some illumination color spot to be create. (In triangle format)
- The each dotted point to be contain 3 color values in 3 direction of the triangle.
- It provides for creating seven Rainbow color for object view UP.
- It generaly used for view of television Object or any Real live object.

Hard copy O/P Devices :-

Computer Graphics S/W :-

- The computer graphics software to be generally used for creating one interactive interacting user friendly object through the end user to perform task manipulation very conviently
- The given graphical s/w provides to perform different Anomalies above the obj
- The graphics software to be initially develop by Ivon Sutherland / on 1960, The electrical scientist [S.] evo one graphical s/w i.e. SKETCH s/w (which looks like pencil) This pencil to be built up using 2 h/w i.e, Photo cell & light
- After that the Khronous group from to develop some graphical library whose embedding with high level language for built up other graphical s/w.
- Initially, the graphics s/w to be available in 2 ways (i) Microsoft paint
  (ii) Adobe Photoshop s/w
- By using Microsoft paint the user performing any graphical object manipulation in free hand momentum ways. This application provides some IDE tools through which the object easily forming. It also provides object Anomalies operation in confined purpose
- The Adobe Photoshop s/w also provides some graphical manipulation above existing objects in high precision formed

Animation Graphic S/W :-

This type of S/W collecting several types of object in single suite layout format, which always use motion properties

CAD S/W :-

Computer Aided design

This type of S/W to be always used for specific purposed, in case of Engineering & Architect field.

Presentation S/W :-

The presentation S/W provides for viewing all the documents in Abstract format through power point presentation.

It also provides for creating different Animation S/W video using existing graphical IDE tools like bar graph, piechart dotted, curve etc.

Highlevel language Graphic S/W :-

In general way As we know, All types of data to be interpret on the format of graphical, so every high level language to be embedded with one graphics library i.e. open GL (Khronous group). This open GL software provides different methods & data elements, through which the high level language interpret whole types of data in graphical format.

1.5 INPUT DEVICES

Following are the commonly used input devices

1. Keyboard
2. Mouse
3. Scanner
4. Trackball/Space ball
5. Joystick
6. Digitizer /Graphical tablet

1. **KEYBOARD:** Keyboard is the primary input device for any graphics system. It is used for entering text and numbers. Keyboards are available in various sizes, shapes, styles. The standard keyboard consists of
   - Alphanumeric keys
   - Function keys
   - Modifier keys
   - Cursor Movement keys
   - Numeric Keypad

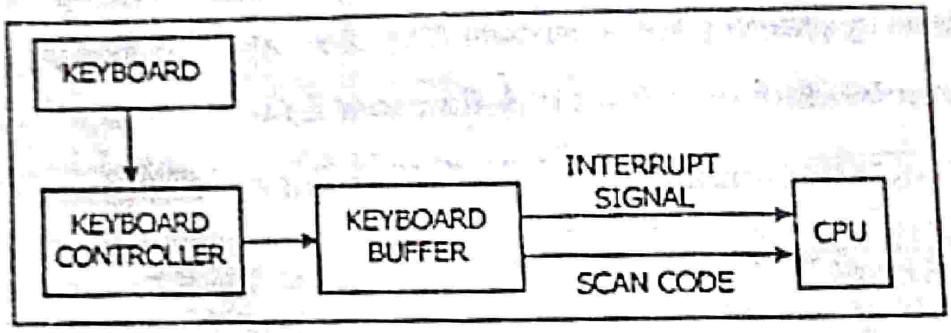


Fig. 1.8 KEYBOARDROCESS

When we press a key on the keyboard, the keyboard controller places a code corresponding to the key pressed in a part of its memory called keyboard buffer. This code is called the scan code. The keyboard controller informs the CPU of the computer about the key pressed with the help of interrupt signals. The CPU then reads the scan code from the Keyboard Buffer.

2. *MOUSE* : A mouse is a palm sized box used to position the screen curs or. It consists of a ball on the bottom connected to wheels to provide the amount and direction of movement. One two or three buttons are usually included on the top of the mouse for signaling the execution of same operation. Now-a-days mouse consists of one more wheel on top to scroll the screen pages.

TRACKBALL&SPACEBALL

A track ball is a ball that can be rotated with the finger or palm of the hand to produce the screen curs or movement. Potentiometers attached to the ball measure the amount and direction of rotation. While the track ball is used for 2D positioning, the space ball is used for 3D positioning & selection of operation in virtual reality system unlike the track ball the space ball does not actually move. It consists of strain gauges which measures the amount of pressure applied to the space ball to provide input for spatial positioning & orientation the ball is pushed pulled in various directions.

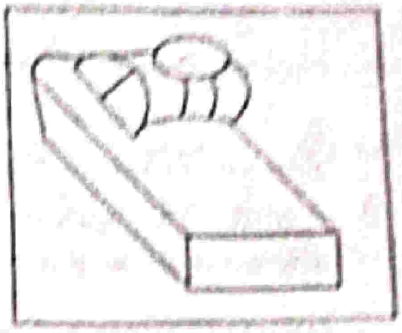


Fig.1.9 Trackball

SCANNER

Scanner have become an important part of the home, office over the last few years. It is a device which can be used to store drawings, graphics, photos as text available in the printed form for computer processing.

Asshown in the fig the photograph is mounted on a rotating drum. A finely collimated light beam is directed at the photo, and the amount of light reflected is measured by a photocell. As the drum rotates the light source slowly moves from one end to the other, thus doing a raster scan of the inter photograph.

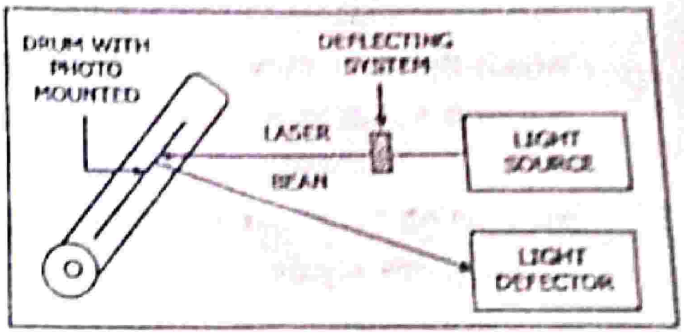


Fig.1.10 PHOTO SCANNER

JOYSTICK

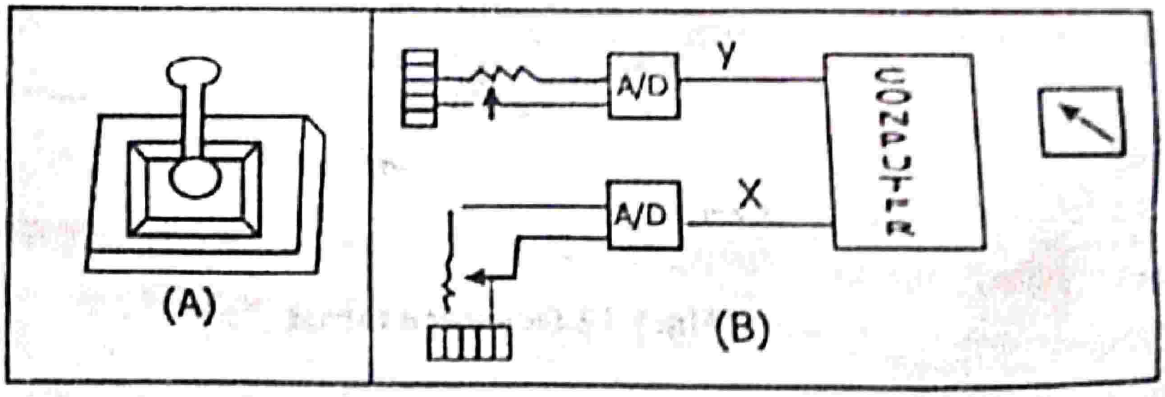


Fig. 1.11 (A) JOYSTICK (B) CIRCUIT DETAILS

A joystick has a small vertical lever mounted on the base and used to steer the screen cursor a round. It consists of two potentiometers attached to a single liver. Moving the liver changes the settings on the potentiometer. The left or right movement is indicated by other one potentiometer & the forward or backward movement is indicated by other potentiometer. Thus with a joy stick hox x & y co-ordinate positions can be simultaneously altered by the motion of a single lever.

Some joysticks may return to this zero (centre) positions when released. Joysticks are inexpressive and quiet commonly used where only rough positioning is needed.

6. DIGITIZER/ GRAPHICAL TABLET

A graphics tablet (or digitizer, digitizing tablet, graphics pad, drawing tablet) is a computer input device that enables a user to hand-draw images and graphics, similar to the way a person draws images with a pencil and paper. These tablets may also be used to capture data or handwritten signatures. It can also be used to trace an image from a piece of paper which is taped or otherwise secured to the surface. Capturing data in this way, either by tracing or entering the corners of linear poly-lines or shapes is called digitizing.

A graphics tablet (also called pen pad or digitizer) consists of a flat surface upon which the user may "draw" or trace an image using an attached stylus, a pen-like drawing apparatus. The image generally does not appear on the tablet itself but, rather, is displayed on the computer monitor. Some tablets, however, come as a functioning secondary computer screen that you can interact with images directly by using the stylus.

Fig.1.12 Graphics tablet

Hard copy Devices :-

All the O/P devices can be categorised into 2 categories

* Hard copy Devices
* Soft copy Devices

- Hard copy devices are those that give the O/P in the tangible form. Printers and plotters are a common hard copy devices
- Soft copy devices give output in the intangible form or the virtual form, e.g. something displayed on a screen. All the computer monitors are covered under this category.

Printers :-

All the Printers irrespective of the technology used can be categorised as

* impact printers
* Non impact printers.

- Impact printers are those printers in which there is a direct contact b/w the printing head & the paper on which the print is produced.
- They work by striking a head or a needle against an inked ribbon which leaves a mark on the paper.
- These printers produce a lot of noise when printing, because of the head striking the paper.
- Examples are Dot Matrix, Daisy wheel and Line printers.

In the case of non-impact printers the printing head never comes in direct contact with papers.

- These printers work by spraying ink on the papers.

- Electrostatic or electromagnetic charge is used in these printers.
- Examples are Ink-jet and Laser printers.

Dot-Matrix Printers :-

- Dot matrix is an impact printer
- These printer forms characters from individual dots
- These printers have a print head which runs back and forth on a paper
- The print head has a two-dimensional array of pins called dot matrix. There may be 9 to 24 pins in the dot matrix.
- From this array of pins some pins are drawn out (or driven forward) to form the shape of a character.
- The drawn out pins strike an ink soaked cloth ribbon against a paper. This forms that particular character on the paper.
- Thus dot matrix printers can be used to print different fonts of characters.
- since mechanical force is used, carbon copies of documents can be taken
- 40 to 250 characters can be printed per second.

Daisy wheel Printers :-

- This is an impact printer
- only preformed fonts of characters can be printed.
- This printer contains a daisy wheel. Daisy wheel is made of plastic or metal. This holds an entire character set as raised characters moulded on each "petal".

- A motor rotates the daisy wheel to posn required character betn the hammer & the ribbon.
- A small hammer then strikes the petal, which in turn strikes the inked ribbon to leave the character mark on the paper.
- The daisy wheel and hammer are mounted on a sliding carriage similar to that used by dot matrix printers.
- Different fonts can't be printed using this technology

Line printer :-

The line printer is a high speed impact printer in which one line is printed at a time.
- 600 - 1200 lines can be printed per minute
- Drum printer is an example of line printer
- These printers are very expensive
- These kind of printers were popular in the early days of computers, but the technology is still in use.

Drum printers :-

On a drum printer, a fixed character set is engraved onto a number of print wheels
- There are as many print wheels as the number of columns (letter in a line) the printer could print
- The print wheels are joined to form a large drum (cylinder)
- This drum spins at high speed and paper and an inked ribbon is moved past the print posn

- As the desired character for each column passes the print posn, a hammer strikes the paper from the rear and presses the paper against the ribbon and the drum, causing the desired character to be printed on the paper.

Ink-Jet Printers :-

- Inkjet printer is a non impact printer, core of an inkjet printer is the print head.
- The print head contains an ink cartridge which has a series of nozzles that are used to spray tiny drops of ink on to the paper.

Plotter :-

A plotter is a computer hw device much like a printer that is used for printing vector graphics. Instead of toner, plotters use a pen, pencil, marker or another writing tool to draw multiple continuous lines on paper rather than multiple dots, like a traditional printer.

Plotter produces a hard copy of schematics and other similar applications. Though once widely used for computer aided design these devices were more or less phased out by wide-format printers.

Advantages of Plotter :-

Plotters can work on very large sheets of paper while maintaining high resolution. They can print on a wide variety of flat material including plywood

aluminium, sheet steel, cardboard & plastic. Plotters allow the same pattern to be drawn thousands of times without any image degradation.

Disadvantages of Plotters:-

- Plotters are quite large compared to a traditional printer.
- Plotters are also much more expensive than a traditional printer.

When was the 1st plotter invented?

The 1st plotter was invented in 1953 by Remington-Rand. It was used in conjuction with the UNIVAC computer to created technical drawings.

Types of Plotter.

Three types of plotters are most popular for their ability to allow you to create different designs.

1. Drum Plotters:-

The Drum Plotter is a special output device whose name indicates its function. This device works by moving a pen on a single axis track while the paper moves on a cylindrical drum. Typically, the drum moves the paper to the right and left, while the pen or pens draw up and down.

2. Flat Bed Plotters:-

Flatbed plotters are o/p device whose name also hints at their function. They work by fixing paper on a flat surface while pens move to draw the image. The pen itself may be attached to an arm, allow it

easily move over the paper. They do not use traditional printing heads, nozzles and ink cartridges. The bed itself is a flat vacuum bed or table meant to keep the paper still.

Flatbed plotters are available in larger sizes than drum plotters. That makes them an ideal choice to create even larger documents, particularly if you are consistently printing larger architectural or CAD drawings

Ink Jet Plotters:-

The 3rd most popular kind of plotter is an inkjet plotter. This device pushes beads of ink directly onto the surface of whatever you are printing on. These inkjet plotters typically print in 3 or 4 color palettes. The 3-color inkjets focus on cyan, magneta and yellow and they mix colors to create darker shades, such as black.

Four-color inkjet plotter are available that have dedicated blank ink. These are recommend when you want the clearest shades of black.

Unit-1

1. Application of computer graphics (desgn)
2. VDU
3. I/O device

Hard copy device
Graphics s/w

Unit-2

1. All Algorithm with example
2. Graphics primitive O/P elemend
3. Filled Area primitive

Unit-3

1. 2D Transformation
2. 3D Transformation
3. Composite Transformation (2D)
4. Homogeneous co-ordinate System

Unit-4

1. 2D Viewing with co-ordinate Transformation
2. Clipping
   Cohen Sutherland line clipping
   Sutherland Hogeman polygon clippi

UNIT-2

→ Graphics output primitive Elements:

In general way, the computer graphics provides some primary o/p graphical functions or object through which some other complex o/p graphical elements to be build up

- Some primitive o/p attributes are given below.

① dot/pixel
② line
③ circle
④ Ellipse
⑤ polygon
⑥ curve

① Dot/pixel :-

It is the 1st graphic o/p primitive elements

- It has no dimension
- The length or size of pixel to be mapping by default 0 to 100

② Line :-

It is the 2nd primitive graphics elements through which some other complex graphical object to be build up

- By connecting more than one pixel point in straight line formed using 45° angle that means in geometrically "for interconnection of 2 end co-ordinate point"
- The line object looks like one dimensional but it always may roll in 2D plane (xy plane)

- It provides one property through which the line size is compressed & also enlarge i.e. length.
- But in general way as the line to be built up using multiple pixel point through 2 end co-ordinate points, so it require computer graphics for calculating intermediate pixel point.
- It used a technique for calculating intermediate pixel point of the line or draw the line. i.e.
  (1) DDA (Digital differential Analyzer) Algorithms
  (2) Bresenham's Algorithm.

① **DDA :-** (Digital differential Analyzer Algorithm)

- It is very simple Algorithm for end user job execution
- This Algorithm consume very less memory space & also time quantum that means here the CPU utilization throughput is much more
- This Algorithm always calculate the next approximation pixel point by assuming one default approximation value using Δy & Δx with the previous pixel point value
- Here the graphical object is not very smooth
- As the line concept to be initiate using straight line so it uses the straight line eqn for calculating all other intermediate pixel point.

The eqn of straight line is
$y = mx + c$
Here x & y are 2 co-ordinate elements of the 2D plane.
C is the y intercept
Here m is the slope of the line i.e.

$$m(\text{slope}) = \frac{y_2 - y_1}{x_2 - x_1} = \frac{dy}{dx} = \frac{\Delta y}{\Delta x}$$

$$\Delta y = m \Delta x$$
$$\Delta x = \frac{\Delta y}{m}$$

The DDA Algorithm provides 3 constraints through which the pixel value position in 2D plane to be easily evaluate.

(I) When $|m| < 1$, the pixel points to be present below 45° angle & Here, $x = x+1$, $y = y$

(II) When $|m| > 1$, then the pixel points to be present above the 45° angle. So Here $y = y+1$, $x = x$

(III) When $|m| = 1$ then the pixel point to be lies on the same observed 45° angle, so Here $x = x+1$, $y = y+1$

Algorithm:-

Let us take 2 end co-ordinate pixel point $P_1(x_1, y_1)$ & $P_2(x_2, y_2)$ & Here to check the 2 end pixel point are not same,

② Now to calculate Δx & Δy

$\Delta x = x_2 - x_1$

$\Delta y = y_2 - y_1$

③ To calculate length of the line using the given formula If ($\Delta x > \Delta y$)

then length = Δx

otherwise

length = Δy

④ Now to calculate the assuming approximation pixel value i.e.

$\Delta x = \frac{\Delta x}{length}$, $\Delta y = \frac{\Delta y}{length}$

(Here Δx or Δy value to be comes one at least any)

⑤ Now to calculate the next pixel point i.e.

$x = x + sign(\Delta x)$

$y = y + sign(\Delta y)$

⑥

```
i = 1
while(i<=length)
{
plot (integer(x), integer(y))
   x = x + sign(Δx)
   y = y + sign(Δy)
   i = i + 1
}
```

Q To draw a line using the pixel points (0,0) to (4,5)

① Here $x_1=0$, $y_1=0$, $x_2=4$, $y_2=5$

② $\Delta x = x_2 - x_1 = 4-0 = 4$
$\Delta y = y_2 - y_1 = 5-0 = 5$

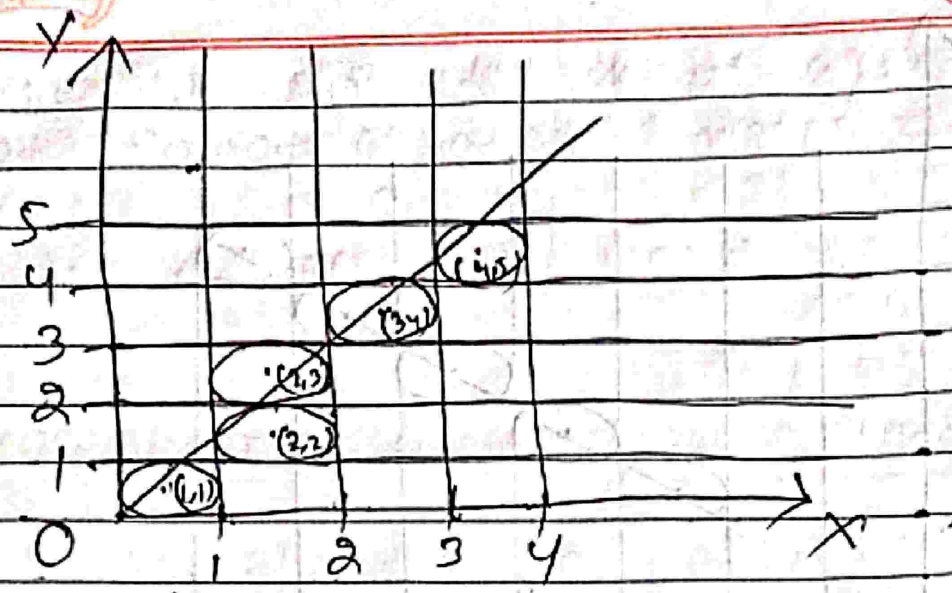
③ if $(\Delta x > \Delta y) \;||\; (4>5)$ ✗
length $= \Delta y = 5$

④ $\Delta x = \frac{4}{5} = 0.8$
$\Delta y = \frac{5}{5} = 1$

⑤ $x = 0 + 0.8 = 0.8$
$y = 0 + 1 = 1$

i	Plot	x	y
		0.8	1
1	(1,1)	1.6	2
2	(2,2)	2.4	3
3	(2,3)	3.2	4
4	(3,4)	4.0	5
5	(4,5)	4.8	6

⑥ while $(i \le 5)$
{ Plot (1,1)
$x = 0.8 + 0.8 = 1.6$
$y = 1 + 1 = 2$
$i = i + 1$
}



Q. (2,3) , (7,8)

① Here $x_1 = 2$, $x_2 = 7$
$y_1 = 3$ $y_2 = 8$

② $\Delta x = x_2 - x_1 = 7 - 2 = 5$
$\Delta y = y_2 - y_1 = 8 - 3 = 5$

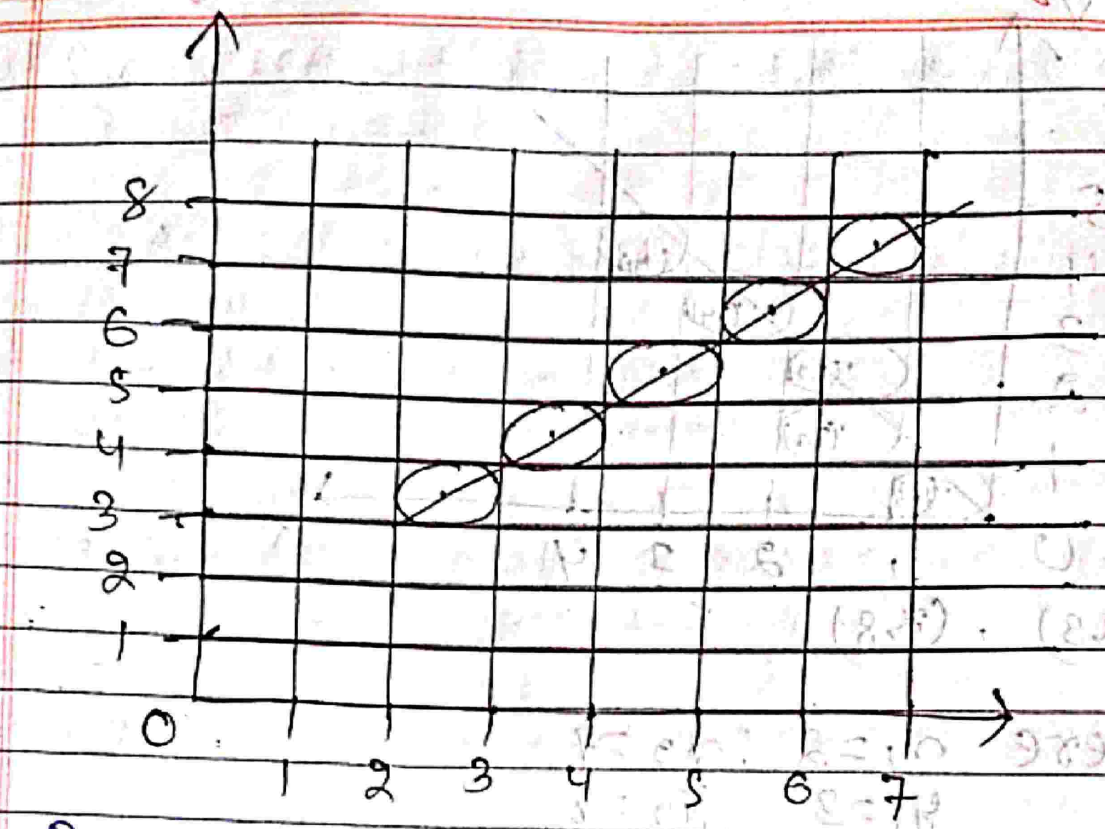
③ if $(\Delta x > \Delta y)$ / $(5 > 5)$ X
length $= \Delta y = 5$

④ $\Delta x = \frac{5}{5} = 1$
$\Delta y = \frac{5}{5} = 1$

⑤ $x = 2 + 1 = 3$
$y = 3 + 1 = 4$

⑥ while $(1 \leq 5)$
{
plot (3, 4)
$x = 3 + 1 = 4$
$y = 4 + 1 = 5$
$i = i + 1$
}

i	plot	x	y
		3	4
1	(3, 4)	4	5
2	(4, 5)	5	6
3	(5, 6)	6	7
4	(6, 7)	7	8
5	(7, 8)	8	9



Bresenham's line drawing Algorithm

DDA	Bresenham's line drawing
(I) This Algorithm calculate all intermediate pixel point using nearest approximation pixel value in precesion format	(I) But here, all the pixel point to be directly evaluate in integer format
(II) Its Architecture design is very costly, because it uses multiplication & division operator	(II) But, its Architecture design is chipper as compared to DDA, because here it uses Addition & substraction operator
(III) It is very simple application for line draw	(III) But it is very complex application for line draw.

(iv) Here the line is not smooth, that means it is slightly jiggered format	(iv) But here the line is very smooth

Bresenham's line drawing Algorithm :-

This Algorithm to be used for drawn up one accuracy line object using one decesion vamable / error term. The decesion vamable to be defined as " The distance betn the nearest pixel value to actual pixel location above given object".

As this method provides for new up the given object in Ranjom way so it uses linear eqn concept

Let us, take the 2 end co-ordinate point i.e. (x_1, y_1) & (x_2, y_2).

Now Assuming the linear eqn using the given 2 end co-ordinate value i.e $ax + by + c = 0$ — (1)

It also used 3 constraints for viewing the object pixel posn above 2D plane

(1). When $f(x, y) < 0$, then the given pixel point to be locate below part of the object (45°)

(2) When $f(x, y) > 0$ then the given pixel point to be locate top part of the object (45°)

(3) Given $f(x,y) = 0$, then the given pixel point to be locate above same object

As we know the straight line eqn i.e

$y = mx + c$

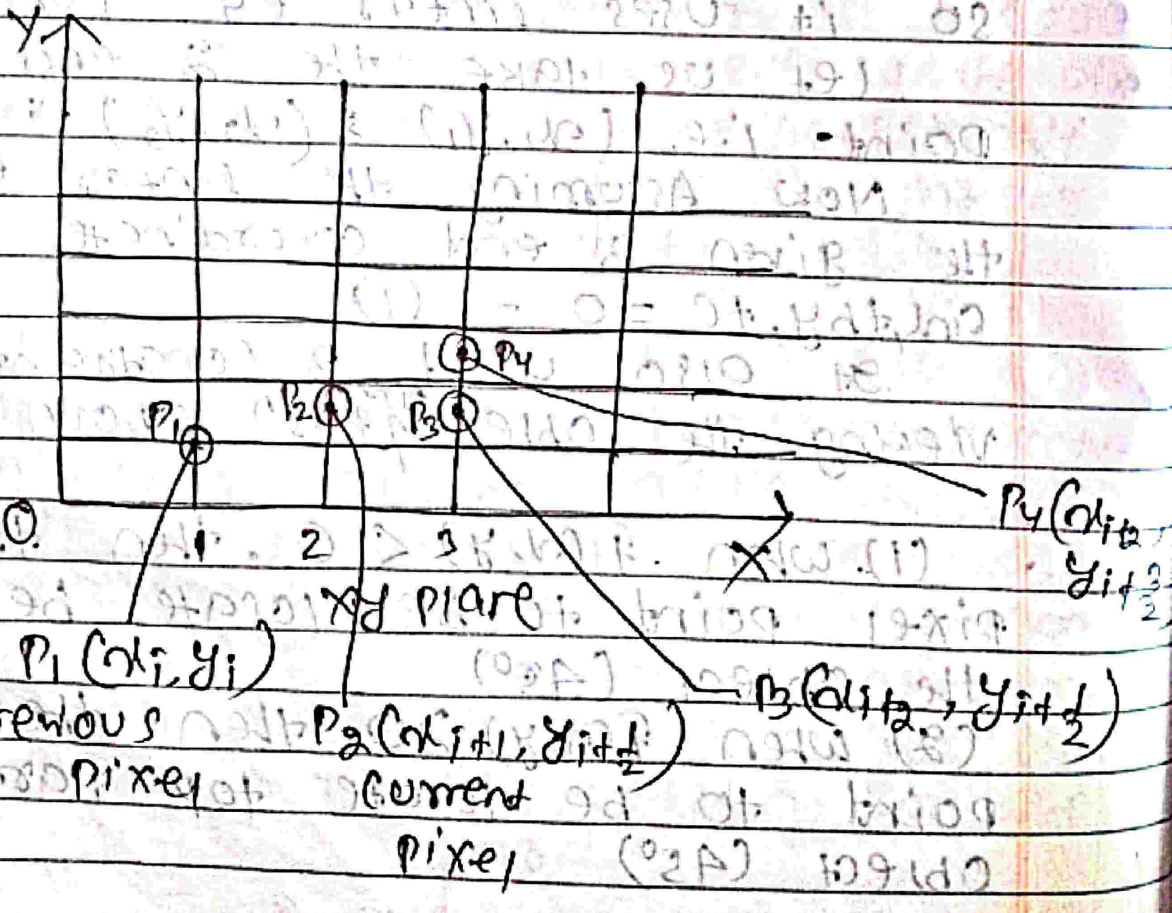
$x = \frac{\Delta y}{\Delta x} x + c$

$\Rightarrow y = \frac{\Delta y x + c\Delta x}{\Delta x}$

$\Rightarrow y\Delta x = \Delta y x + c\Delta x$

$\Rightarrow \Delta y x + (-\Delta x y) + c\Delta x = 0$

Here $a = \Delta y$, $b = -\Delta x$ & $c = c\Delta x$



Now to find the initial decision parameter value i.e. d_0 at pixel point $P_2(x_{i+1}, y_{i+\frac{1}{2}})$ so now eq^n (1) becomes

$$d_0 = ax_{i+1} + by_{i+\frac{1}{2}} + c$$
$$= ax_i + a + by_i + b/2 + c$$
$$= ax_i + by_i + c + a + b/2$$
$$= 0 + a + b/2$$
$$d_0 = \frac{2a+b}{2}$$
$$\Rightarrow 2d_0 = 2a+b$$

As here d_0 is a small value so cancel the constant element

$$\Rightarrow d_0 = 2a+b$$
$$\Rightarrow \boxed{d_0 = 2\Delta y - \Delta x} \quad \text{——} (3)$$

Now to find the next pixel location above 2D plane through the current decision parameter, so, assume the next pixel is $P_3(x_{i+2}, y_{i+\frac{1}{2}})$

Now the eq^n (1) becomes

$$d_{next} = ax_{i+2} + by_{i+\frac{1}{2}} + c$$
$$= ax_{i+1} + 1 + by_{i+\frac{1}{2}} + c$$
$$= ax_{i+1} + a + by_{i+\frac{1}{2}} + c$$
$$= ax_{i+1} + by_i + 1/2 + c + a$$
$$d_{next} = d_0 + a$$
$$= d_0 + 2a$$
$$\boxed{d_{next} = d_0 + 2\Delta y} \quad \text{——} (4)$$

If assume the next pixel point P_4 i.e $P_4(x_i + 2, y_i + 3/2)$

$$d_{next} = ax_{i+2} + by_{i+3/2} + c$$
$$= ax_{i+1+1} + by_{i+\frac{1}{2}} + 1 + c$$
$$= ax_{i+1} + a + by_{i+\frac{1}{2}} + b + c$$

$= a x_{i+1} + b y_{i+1/2} + c + a + b$

$= d_0 + a + b$

$= d_0 + 2a + 2b$

$$\boxed{d_{next} = d_0 + 2(\Delta y - \Delta x)} \quad \text{— (5)}$$

Algorithm:-

(1)- consider the two end co-ordinate pixel point are store the left end co-ordinate point into buffer

(2) Now to plot the left end co-ordinate point plot (x_1, y_1) (initial pixel point)

(3) Now to calculate all constant value i.e. Δx, $2\Delta y$ and $2\Delta x$ for initial decesion parameter $d_0 = 2\Delta y - \Delta x$

(4) It calculates d_k at $k=0$ when (if $(d_k < 0)$) then it calculate next pixel point i.e.

$d_{k+1} = d_k + 2\Delta y$, $x = x+1$, $y = y$

else

$d_{k+1} = d_k + 2(\Delta y - \Delta x)$

(5) Repeat step-4 Δx times ($x = x+1$, $y = y+1$)

Q To draw a line using the 2 end co-ordin point (2,4) to (5,7)

Ans:-

1- $x_1 = 2$, $y_1 = 4$, $x_2 = 5$, $y_2 = 7$

(1) Buffer = (2,4)

2- Plot (2,4)

3- $\Delta x = x_2 - x_1 = 5 - 2 = 3$

$\Delta y = y_2 - y_1 = 7 - 4 = 3$

$2\Delta x = 2 \times 3 = 6$

$2\Delta y = 2 \times 3 = 6$

k	Plot	x	y	d_k
	(2,4)	2	4	
0	(2,4)	3	5	3
1	(3,5)	4	6	3
2	(4,6)	5	7	3
3	(5,7)	6	8	3

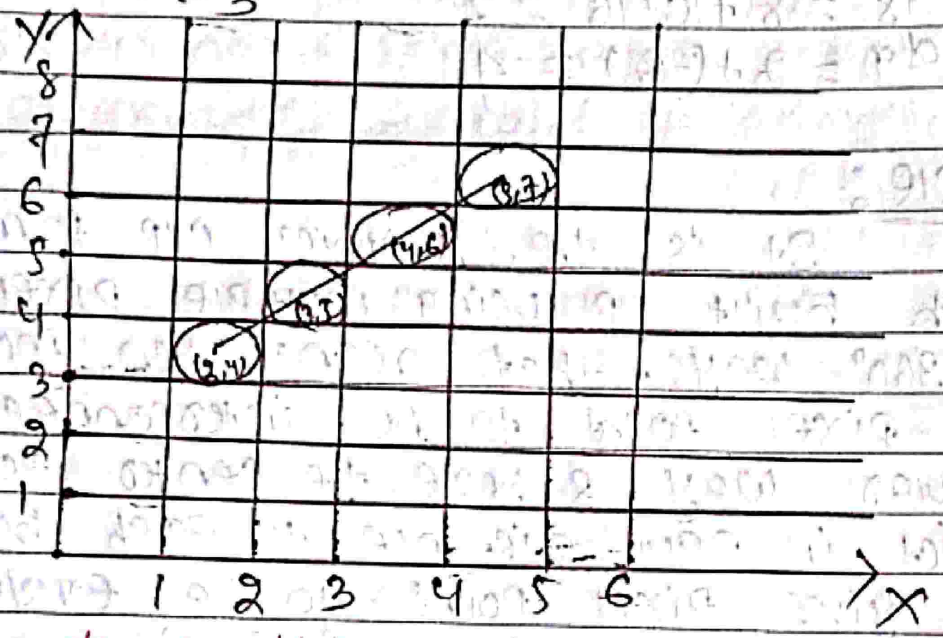
$d_0 = 6-3 = 3$

4 - if $(3<0)x$

$d_1 = d_0 + 2\Delta y - 2\Delta x$

$= 3 + 6 - 6$

$= 3$



Q. To draw a line using the pixel point (2,5) to (10,10)

Ans- 1- $x_1=2$, $y_1=5$, $x_2=10$, $y_2=10$

Buffer (2,5)

2- Plot (2,5)

3- $\Delta x = 10-2 = 8$

$\Delta y = 10-5 = 5$

$2\Delta x = 2 \times 8 = 16$

$2\Delta y = 2 \times 5 = 10$

$d_0 = 10-8 = 2$

4- if $(2<0)x$

$d_1 = d_0 + 2\Delta y - 2\Delta x$

$= 2 + (10-16)$

$= -4$

$d_2 = -4 + 10$

$= 6$

$d_3 = 6 + 10 - 16 = 0$

K	Plot	x	y	d_k
	(2,5)	2	5	2
0	(2,5)	3	6	2
1	(3,6)	4	6	-4
2	(4,6)	5	7	6
3	(5,7)	6	8	0
4	(6,8)	7	8	-6
5	(7,8)	8	9	4
6	(8,9)	9	9	-2
7	(9,9)	10	10	8
8	(10,10)	11	11	2
9				-4

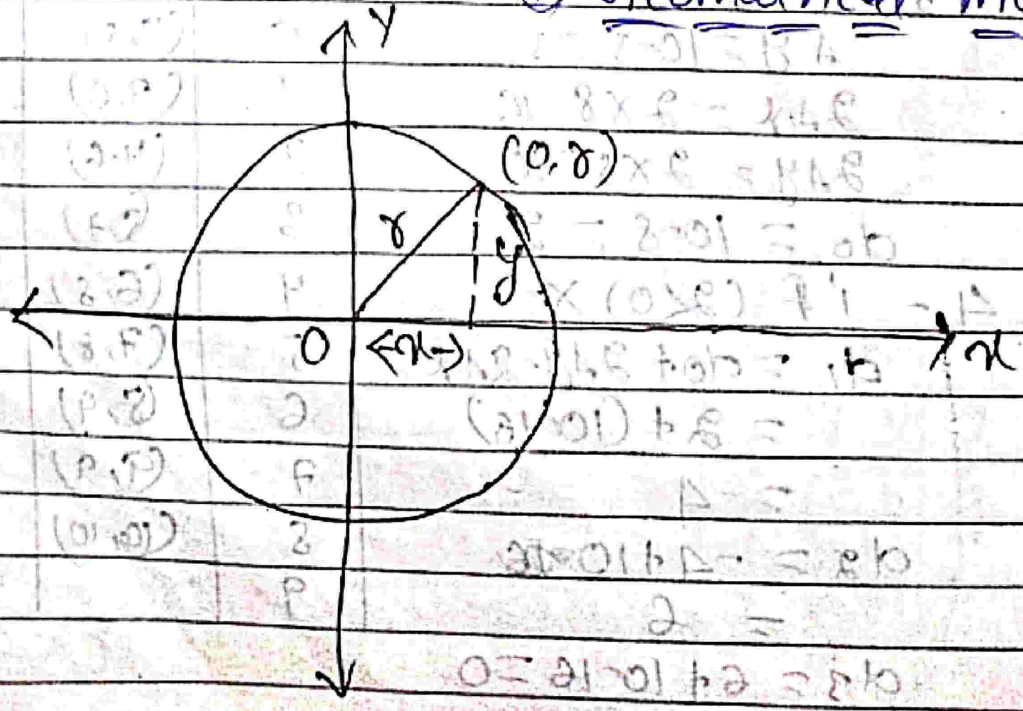
$d_4 = 3 + 10 - 16 = -6$
$d_5 = -6 + 10 = 4$
$d_6 = 4 + 10 - 16 = -2$
$d_7 = -2 + 10 = 8$
$d_8 = 8 + 10 - 16 = 2$
$d_9 = 2 + (-6) = -4$

Circle :-

It is the graphics O/P primitive elements which built up using multiple pixel elements in 360° ways. that means, when more than one pixel point to be interconnected in circular way & here the center posn of the object in equi distance to each boundary
- The given pixel point to be evaluate using circle equation i.e. $x^2+y^2=r^2$

Now here uses 2 technique for evaluating each pixel points through eqn of circle i.e.
(1) Geomatric Method
(2) Trigonmetric Method

(1) Geomatrical Method



initially, if we incrementing x co-ordinate value, then the y co-ordinate value to be affected above my plane at $(0, r)$.

As it affected y co-ordinate value so it is very inefficient mechanism.

So using the geometrical method, the x co-ordinate to be calculate i.e. $x^2 = r^2 - y^2$

$$\Rightarrow x = \sqrt{r^2 - y^2}$$

N.B:-

This geometrical method to be used when the circle to be build up using 4 pixel point.

(2) Trigometric Method :-

When applied trigonmetric method here each quadrant to be divided into 2 sectors through angle θ i.e. 45°

By using right angle triangle here

$x = r\cos\theta$ & $y = r\sin\theta$

So now the geometrical eqn becomes i.e.

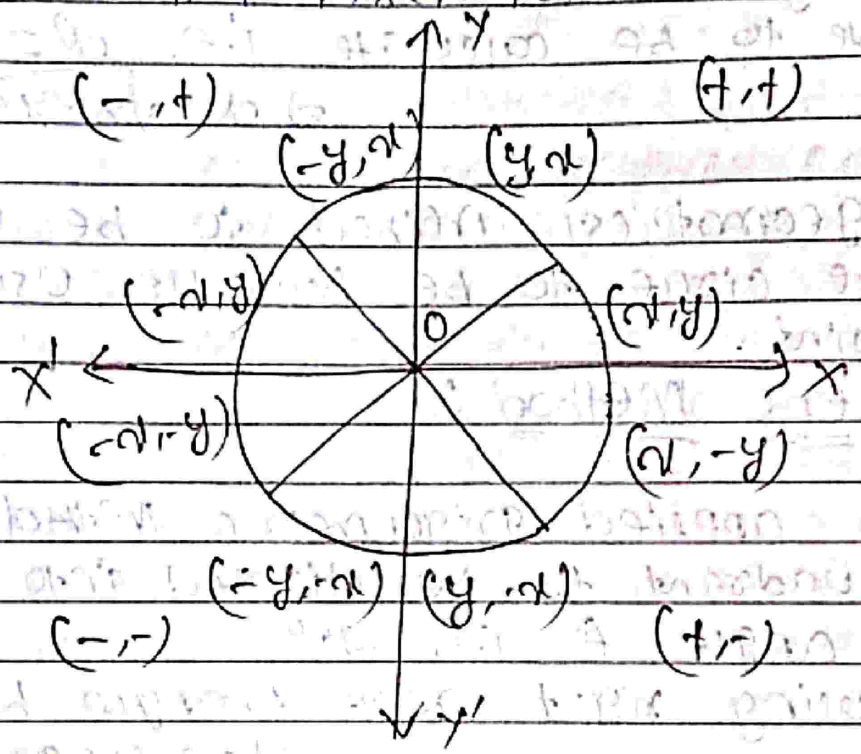
$x = r\cos 45^\circ = \frac{r}{\sqrt{2}}$ $\quad$ $y = r\sin 45^\circ = \frac{r}{\sqrt{2}}$

Now $x = \sqrt{r^2 - \frac{r^2}{2}} = \sqrt{\frac{r^2}{2}} = \frac{r}{\sqrt{2}}$

& $y = r$

8 Symmetric method :-

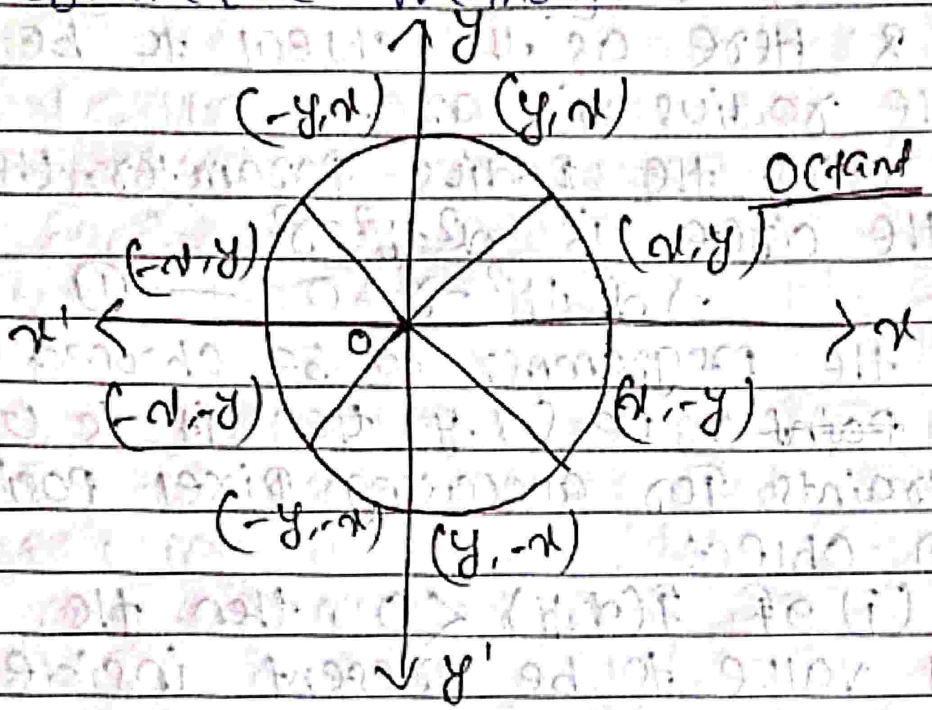
This methods to be used for calculating 7 pixel point of circle through one pixel point where the radius is hidden up
It applys symmetric rules using (x,y) plane & perpendicular xy plane through the initial pixel point



So here initial pixel value is (x,y)
Now the other 7 pixel value i.e.

$= \{(x,y), (y,x), (x,-y), (y,-x), (-x,-y), (-y,x), (-x,y), (-y,x)\}$

Ex To calculate 7 octant points using one pixel value of circle i.e. (-4,5) through 8 symmetric method.



The given pixel point P(-4,5)

Here $x = 4$

$y = 5$

So the 8 pixel points are

$S = [(-4,5), (-5,4), (5,4), (4,5), (4,-5), (5,-4), (-5,-4), (-4,-5)]$

Midpoint Circle drawing Algorithm :-

- It is an easiest method for calculating circle & pixel value through squaring and symmetric method.
- This Method calculating 8 pixel value and Rasterized the given object i.e. the circle.
- Here the programmer to be required for calculating one octant pixel value & through this pixel value the others 7 symmetric

Octant value to be evaluate

As we know, in 2D plane every object to be view of using 2 co-ordinate value i.e. (x,y) & Here as the object to be circle so the radius is arc.

By using the Radius Parameter, the equation of the circle is $x^2+y^2=r^2$

$$\Rightarrow x^2+y^2-r^2=0 \quad \text{---} (1)$$

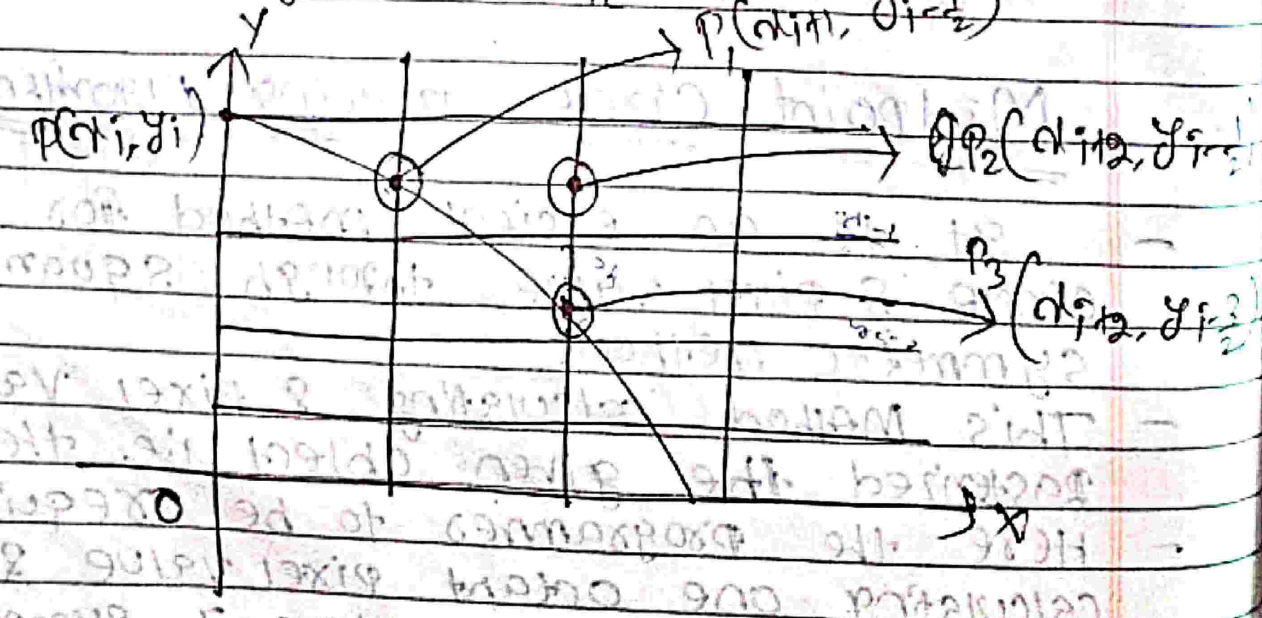
If the programmer to be choored the pixel value (x,y) then it is used 3 constraints for allocating pixel position above given object

(i) If $f(x,y)<0$, then the given pixel value to be present inside the circle

(ii) If $f(x,y)>0$, then the given pixel value to be present outside of the circle

(iii) If $f(x,y)=0$, then the given pixel value to be present on the circle

Now to calculate all pixel value of the circle through radius r



Let us assume 3 pixel value i.e. current pixel value $P_1(x_{i+1}, y_{i-\frac{1}{2}})$, next pixel value i.e. $P_2(x_{i+2}, y_{i-\frac{1}{2}})$ or $P_3(x_{i+2}, y_{i-\frac{3}{2}})$

Now to calculate the initial decision parameter value using the current pixel value i.e $P_1(x_{i+1}, y_{i-\frac{1}{2}})$

Now eqn (1) becomes

$$d_0 = (x_{i+1})^2 + (y_{i-\frac{1}{2}})^2 - r^2 = 0 \quad \text{— (2)}$$

now to calculate the next decision parameter using the point $P_2(x_{i+2}, y_{i-\frac{1}{2}})$ so now eqn (1) becomes

$$d_{next} = (x_{i+2})^2 + (y_{i-\frac{1}{2}})^2 - r^2$$

$$= (x_{i+1}+1)^2 + (y_{i-\frac{1}{2}})^2 - r^2$$

$$= x_{i+1}^2 + 2x_{i+1} + 1 + y_{i-\frac{1}{2}}^2 - r^2$$

$$= d_0 + 2x_i + 2 + 1$$

$$= d_0 + 2x_i + 3 \quad \text{— (3)}$$

So for $P_3(x_{i+2}, y_{i-\frac{3}{2}})$ now eqn (1) becomes

$$d_{next} = (x_{i+2})^2 + (y_{i-\frac{3}{2}})^2 - r^2$$

$$= (x_{i+1}+1)^2 + (y_{(i-\frac{1}{2})-1})^2 - r^2$$

$$= x_{i+1}^2 + 2x_{i+1} + 1 + y_{i-\frac{1}{2}}^2 - 2y_{i-\frac{1}{2}} + 1 - r^2$$

$$= d_0 + 2x_i + 3 - 2y_i + 1 + 1$$

$$= d_0 + 2x_i - 2y_i + 5$$

$$= d_0 + 2(x_i - y_i) + 5 \quad \text{— (4)}$$

Current

In case of circle the initial pixel value i.e. $(0, r)$ at radius r. By using given current pixel value, the initial decision parameter i.e.

$$x_{i+1} = x_i + 1 = 0 + 1 = 1$$

$$y_i - \frac{1}{2} = y_i - \frac{1}{2} = r - \frac{1}{2}$$

$$d_0 = (x_{i+1})^2 + (y_i - \tfrac{1}{2})^2 - r^2$$

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

$$d_0 = 1 + r^2 - 2r \cdot \frac{1}{2} + \frac{1}{4} - r^2$$

$$= 1 - r + \frac{1}{4}$$

$$= \frac{5}{4} - r = 1 - r$$

Algorithm's

1. To Assign the initial current pixel value using radius r i.e. $x = 0$, $y = r$
2. Now to calculate the initial decision parameter $d_1 = 1 - r$
3. Repeting the steps from step 4 to 7 when $(x < y)$
4. setpixel (x, y)
5. If $(d_i < 0)$

$$d_{i+1} = d_i + 2x_i + 3$$

6. else

$$d_{i+1} = d_i + 2(x_i - y_i) + 5$$

$$y = y - 1$$

[If - else end]

7. $x = x + 1$
8. End

Ex To draw a circle by using the Radius 5 through mid point circle Algorithm.

Algorithm

① $x=0$, $y=r=5$

② $d_i = 1-r = 1-5 = -4$

③ while $(x<y)$

$(0<5)$ ✓

④ setPixel $(0,5)$

i	Plot	x	y	d_i
0	(0,5)	0	5	-4
1	(1,5)	1	5	-1
2	(2,5)	2	5	4
3	(3,4)	3	4	3
4		4	3	6

⑤ if $(d_i<0)$

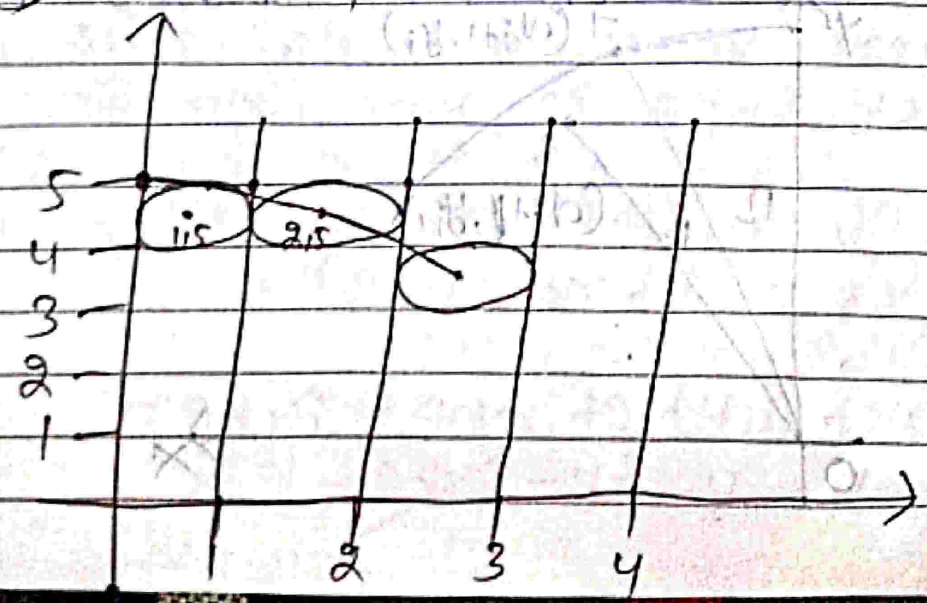
$(-4<0)$ ✓

$d_i = d_0 + 2x_i + 3$

$= -4 + 0 + 3 = -1$

⑦ $x = 0+1 = 1$

O/P



Bresenham's Circle drawing Algorithm:

This circle drawing Algorithm is similar as Mid-point circle drawing Algorithm, but there is some difference attributes to be added i.e.

(1) It provides one optimal solution, which consumes very less time & very less space for its execution.

(2) It avoids Trigonometric & Squaring Method

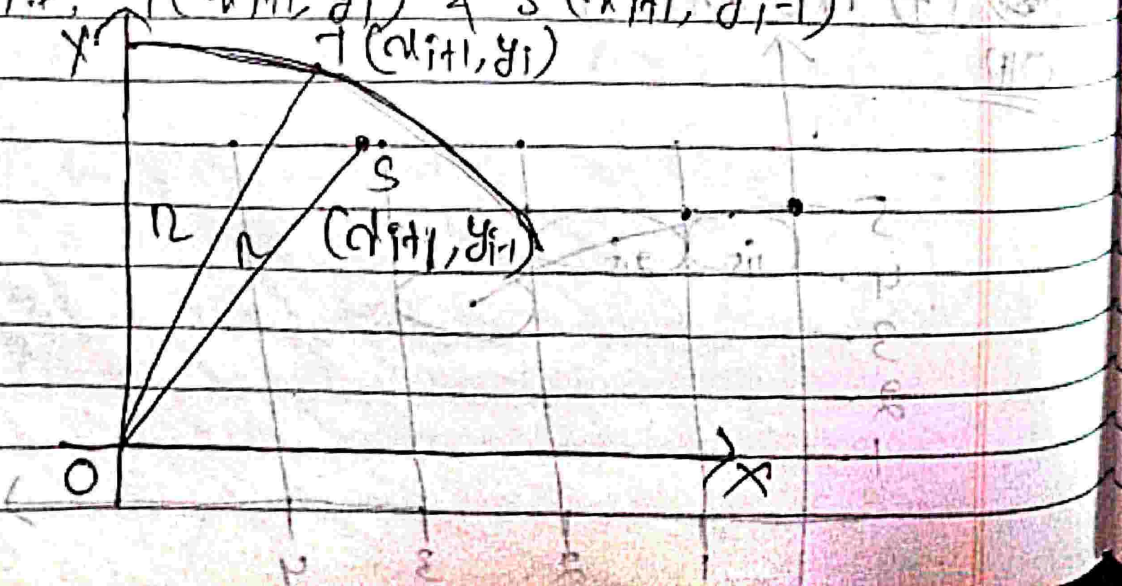
(3) It used two constraints i.e.

(a) Here x co-ordinate value always increments 1unit value;

(b) But the y-th co-ordinate value some times to be constant as previous pixel value or decrements unit value.

As it built up using 8 pixel value, so it also use 8 symmetric Method. Here also one octant to be evaluate from 90° to 45°

Let us assume the current pixel value of x,y plane i.e. (x_i, y_i). The next pixel value to be evaluate using the true Radius value i.e. R. Let Here T & S 2 variable pixel i.e. $T(x_{i+1}, y_i)$ & $S(x_{i+1}, y_{i-1})$.



Let D(T) = distance from origin to pixel T squared minus from actual radius of the circle.

Let D(S) = distance from origin to pixel S squared from actual radius of the circle

for choosing the given 2 pixel value it used 2 condition

① if $d_i < 0$ then

D(T) < DS, Here pixel T choosen up

② if $d_i \geq 0$

D(T) ≥ DS, Here pixel S choosen up.

Now to find $D(T) = x_{i+1}^2 + y_i^2 - r^2$ — ①

Similarly $D(S) = x_{i+1}^2 + y_{i-1}^2 - r^2$ — ②

Now to find the 1st decision parameter value i.e, $d_i = D(T) + D(S)$

$$d_i = x_{i+1}^2 + y_i^2 - r^2 + x_{i+1}^2 + y_{i-1}^2 - r^2$$

$$d_i = 2x_{i+1}^2 + y_i^2 + y_{i-1}^2 - 2r^2 \quad \text{— ③}$$

Now to find the next decision parameter value put i = i+1

$$d_{i+1} = 2(x_{i+1}+1)^2 + y_{i+1}^2 + (y_{i+1}-1)^2 - 2r^2 \quad \text{— ④}$$

By using initial decision parameter, the actual pixel value of octant to be evaluate i.e.

$$d_{i+1} - d_i = 2(x_{i+1})^2 + y_{i+1}^2 + (y_{i+1}-1)^2 - 2r^2 - \{2x_{i+1}^2 + y_i^2 + y_{i-1}^2 - 2r^2\}$$

$$= 2x_{i+1}^2 + 4x_{i+1} + 2 + y_{i+1}^2 + y_{i+1}^2 - 2y_{i+1} + 1 - 2r^2 - 2x_{i+1}^2 - y_i^2 - y_{i-1}^2 + 2r^2$$

$= 4x_{i+1} + 2y_{i+1}^2 - 2y_{i+1} - y_i^2 - y_{i-1}^2 + 3$

$= 4x_i + 4 + 2y_i + 1^2 - 2y_{i+1} - y_i^2 - y_i^2 + 2y_i - 1 + 3$

$= 4x_i + 2(y_{i+1}^2 - y_i^2) - 2(y_{i+1} - y_i) + 6$ ——— (5)

When choose the pixel value 7 Hex

$y_{i+1} = y_i$

$d_{i+1} = d_i + 4x_i + 2(y_i^2 - y_i^2 - 2(y_i - y_i) + 6$

$= d_i + 4x_i + 6$ ——— (6)

When choose the pixel value 5 Hex put $y_{i+1} = y_{i-1}$

$d_{i+1} = d_i + 4x_i + 2(y_{i-1}^2 - y_i^2) - 2(y_{i-1} - y_i) + 6$

$= d_i + 4x_i + 2(y_i^2 - 2y_i + 1 - y_i^2) - 2(y_{i-1} - y_i) + 6$

$= d_i + 4x_i - 4y_i + 2 + 2 + 6$

$= d_i + 4(x_i - y_i) + 10$

Let the initial pixel value of the circle i.e. $(0, r)$, so now the eq^n (3) becomes

$d_i = 2(x_i + 1)^2 + y_i^2 + (y_i - 1)^2 - 2r^2$

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

$= 2(0 + 0 + 1) + r^2 + r^2 - 2r + 1 - 2r^2$

$= 3 - 2r$

R=-7

Algorithm:-

initial

(1) To set 1 (x,y) coordinate value of the circ
i.e. (0,r) through radius r
Here x=0, y=r

(2) To calculate initial decision parameter
$d_i = 3-2r$

(3) Repeating the step 4 to 7 while($x \le y$)

(4) set pixel (x,y)

(5) if ($d_i < 0$)
$d_{i+1} = d_i + 4x_i + 6$

(6) else
$d_{i+1} = d_i + 4(x_i - y_i) + 10$
[y=y-1]
(if-else close]

(7) x=x+1

(8) End.

Ex To draw a circle by using radius 10 through Bresenham's circle drawing method.

(1) x=0, y=10=r

(2) $d_i = 3-2(10)$
$= 3-20 = -17$

(3) while ($0 \le 10$)

(4) set pixel (0,10)

i	d_{i+1}	plot	x	y	d_i
1	-11	(0,10)	1	10	-17
2	-1	(1,10)	2	10	-11
3	13	(2,10)	3	10	-1
4	-5	(3,10)	4	9	13
5	17	(4,9)	5	9	-5
6	11	(5,9)	6	8	17

next

7	13	(6,8)	7	7	11
8	23	(7,7)	8	6	13

⑤ if (di<0)
(-17<0) ~
di+1 = -17 + 4(0) + 6
= -17 + 6
= -11

⑦ x = x+1
= 0+1 = 1

* R = -7

① x=0, y=R=-7

② di = 3-2R
= 3-2(-7) = 3+14 = 17

③ while(x y)
(0)

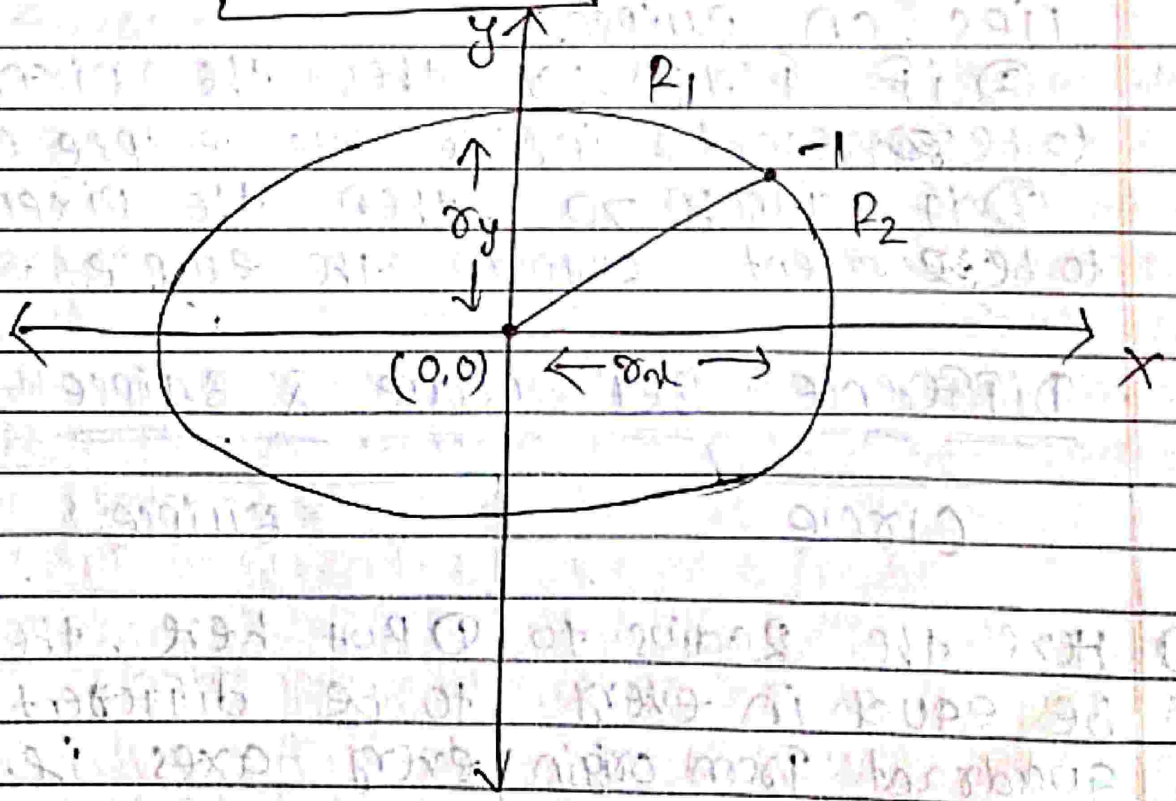
④ setpixel(,)

Ellipse :-

It is an Efficient prominent o/p graphics primitive element. It is an elongated version of circle, where the Radius to be different from centre of Ellipse It elongated the given circle in uni-direction ways. This two are denoted by major axis and minor axis

Suppose the centre of the Ellipse is (0,0) and here the Radii of 2 axis i.e. r_x & r_y. Now to calculate the initial pixel value of Ellipse using the given below formula

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad \text{---------- (1)}$$



Here $a = r_x$ & $b = r_y$ now eqn (1) becomes

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

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

$$\Rightarrow x^2 r_y^2 + y^2 r_x^2 = r_x^2 r_y^2$$

$$\Rightarrow x^2 r_y^2 + y^2 r_x^2 - r_x^2 r_y^2 = 0 \quad \text{---} \quad (2)$$

As it is derived from circle, it also used the 3 given constraints for pixel position. evaluation above graphics.

① if $f(x,y) = 0$ then the pixel value lies on ellipse.
② if $f(x,y) < 0$ then the pixel value to be present inside the ellipse
③ if $f(x,y) > 0$, then the pixel value to be present outside the ellipse.

Difference Bet^n Circle & Ellipse

	Circle	Ellipse
①	Here the Radius to be equal in every quadrant from origin Pos^n of the circle	① But here, the radius to be different in every axes i.e. r_x, r_y

② It always follows 8 symmetric method	② But it always follows 4 symmetric method
③ Here only one octant value to be evaluate then automatically the other 7 to be interpret.	③ But here only one quadrant to be evaluate then automatically the other 3 to be interpret.
④ It evaluate the actual pixel value to assuming x coordinate in increment ways & the y co-ordinate to be constant	④ But as here, the Radii to be different so it calculate the actual pixel value using 2 region method.

Region-1:-

Here all the pixel value to be evaluate using radius R_y, where $P(R_1) < -1$. In this region, the x co-ordinate value always increment one unit but the yth coordinate value can't be determined accurately (because some times here the yth coordinate value to be constant or decrement one)

Region-2:-

Here all the pixel value to be evaluate using radius R_x, where $P(R_2) > -1$. In this region the yth co-ordinate value always decrement one unit value but the x co-ordinate value can't be determine accurately

(because some times here x coordinate value to be constant or increment or unit)

Slope calculation

Assuming the slope of the intermediate region pixel value of ellipse quadrant that is -1

Now the eqn (2) becomes

$$\frac{dy}{dx}(x^2r_y^2 + y^2r_x^2 - r_x^2r_y^2) = 0$$

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

$$\Rightarrow \frac{dy}{dx}2yr_x^2 = -2xr_y^2 + 0$$

$$\Rightarrow \frac{dy}{dx} = \frac{-2xr_y^2}{2yr_x^2} \quad (-1)$$

$$\Rightarrow 2yr_x^2 = -2xr_y^2$$

Region -1 (Eqn calculation)

let us assume the 2 pixel value of region-1 i.e. $P_1(x_{i+1}, y_i)$ & $P_2(x_{i+1}$ now to find the intermediate pixel value for decision parameter calculation i.e. $P_3(x_{i+1}, y_{i-\frac{1}{2}})$

Now to put the intermediate pixel value in eqn (2)

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

$$d_i = (x_{i+1})^2 r_y^2 + \left(y_i - \frac{1}{2}\right)^2 r_x^2 - r_x^2 r_y^2 \quad \text{— (3)}$$

Now to find the next decision parameter value

$$d_{i+1} = (x_{i+1}+1)^2 r_y^2 + \left(y_{i+1} - \frac{1}{2}\right)^2 r_x^2 - r_x^2 r_y^2 \quad \text{— (4)}$$

Now to calculate the actual next decision parameter value i.e.

$$d_{i+1} - d_i = \{(x_{i+1}+1)^2 r_y^2 + (y_{i+1} - \tfrac{1}{2})^2 r_x^2 - r_x^2 r_y^2$$

$$- \{(x_{i+1})^2 r_y^2 + (y_i - \tfrac{1}{2})^2 r_x^2 - r_x^2 r_y^2$$

$$= [(x_{i+1}^2 + 2x_{i+1} + 1) r_y^2 + (y_{i+1}^2 - 2y_{i+1} + \tfrac{1}{4}) r_x^2 - r_x^2 r_y^2]$$

$$- (x_{i+1}^2 r_y^2 - (y_i^2 - y_i + \tfrac{1}{4}) r_x^2 - r_x^2 r_y^2)$$

$$= r_y^2 (x_{i+1}^2 + 2x_{i+1} - x_{i+1}^2) + r_x^2 (y_{i+1}^2 - y_{i+1} + \tfrac{1}{4} - y_{i+1} + \tfrac{1}{4} - y_i^2 + y_i - \tfrac{1}{4})$$

$$= r_y^2 (2(x_{i+1}+1)) + r_x^2 (y_{i+1}^2 - y_i^2 - y_{i+1} + y_i) \quad \text{— (5)}$$

When the previous decision parameter value is less than 0 then $y_{i+1} = y_i$ now eqn (5) becomes.

$$d_{i+1} = d_i + r^2_y (2(x_{i+1}+1) + r^2_x (y_i^2 - y_i^2 - y_i + y_i)$$

$$\boxed{d_{i+1} = d_i + r^2_y (2(x_{i+1}+1)} \quad \text{——— (6)}$$

But when the previous decision parameter value greater than or equal to zero then $y_{i+1} = y_i - 1$

$$d_{i+1} = d_i + r^2_y 2(x_{i+1}+1) + r^2_x (y_{i-1}^2 - y_i^2 - y_{i-1} + 1$$

$$= d_i + r^2_y 2(x_{i+1}+1) + r^2_x (y_i^2 - 2y_i + 1 - y_i^2 + 1 + y_i)$$

$$= d_i + r^2_y 2(x_{i+1}+1) + r^2_x (-2y_i + 2)$$

$$\boxed{d_{i+1} = d_i + r^2_y 2(x_{i+1}+1) + 2r^2_x (-y_i + 1)}$$

initial ——— (7)

Now to find the actual decision parameter value using the pixel $(0, r_y)$

$$d_i = (x_{i+1})^2 r^2_y + (y_i - \tfrac{1}{2})^2 r^2_x - r^2_x r^2_y$$

$$= (x_i^2 + 2x_i + 1) r^2_y + (y_i^2 - y_i + \tfrac{1}{4}) r^2_x - r^2_x r^2_y$$

$$= (0 + 0 + 1) r^2_y + (r^2_y - r_y + \tfrac{1}{4}) r^2_x - r^2_x r^2_y$$

$$= r^2_y + r^2_y r^2_x - r_y r^2_x + \tfrac{1}{4} r^2_x - r^2_x r^2_y$$

$$\boxed{d_i = r^2_y - r^2_x r_y + \tfrac{1}{4} r^2_x} \quad \text{——— (8)}$$

Region-2

Let us take, the two pixel value of Region-2 i.e. $P_1(x_i, y_{i-1})$, $P_2(x_{i+1}, y_{i-1})$ Here y_{th} co-ordinate value to be always decrease 1 unit but x co-ordinate value to be required for calculation.

Now to take the middle value of 2 pixel for calculating initial deresion parameter.

Now the eqn (2) becomes.

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

$$= (x_i + \tfrac{1}{2})^2 r_y^2 + (y_i - 1)^2 r_x^2 - r_x^2 r_y^2 \quad \text{---} (9)$$

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

$$= (x_{i+1} + \tfrac{1}{2})^2 r_y^2 + (y_{i-1} - 1)^2 r_x^2 - r_x^2 r_y^2 \quad \text{---} (10)$$

So $d_{i+1} - d_i = \{(x_{i+1} + \tfrac{1}{2})^2 r_y^2 + (y_{i-1} - 1)^2 r_x^2 - r_x^2 r_y^2\} - \{(x_i + \tfrac{1}{2})^2 r_y^2 + (y_{i-1})^2 r_x^2 - r_x^2 r_y^2\}$

$$= \{(x_{i+1}^2 + x_{i+1} + \tfrac{1}{4}) r_y^2 + (y_{i-1}^2 - 2y_{i-1} + 1) r_x^2\} - \{(x_i^2 + x_i + \tfrac{1}{4}) r_y^2 + (y_{i-1})^2 r_x^2\}$$

$$d_{i+1} = d_i + (x_{i+1}^2 + x_{i+1} - x_i^2 - x_i) r_y^2 + (-2y_{i-1} + 1) r_x^2 \quad \text{---} (11)$$

Suppose the decesion parameter value is > 0

now to put

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

$$\Rightarrow d_{i+1} = d_i + (x_i^2 + x_i - x_i^2 - x_i)\gamma_y^2 + (-2y_i)\gamma_x^2$$

$$\Rightarrow \boxed{d_{i+1} = d_i + (-2y_{i-1} + 1)\gamma_x^2} \quad \text{---------- (12)}$$

When the decesion parameter value is ≤ 0 $x_{i+1} = x_{i+1}$

$$d_{i+1} = d_i + (x_{i+1}^2 + x_{i+1} - x_i^2 - x_i)\gamma_y^2 + (-2y_{i-1})$$

$$= d_i + (x_i^2 + 2x_i + 1 + x_i + 1 - x_i^2 - x_i)\gamma_y^2 + (-2y_{i-1} + 1)\gamma_x^2$$

$$= d_i + (2x_i + 2)\gamma_y^2 + (-2y_{i-1} + 1)\gamma_x^2$$

$$= d_i + 2(x_i + 1)\gamma_y^2 + (-2y_{i-1} + 1)\gamma_x^2$$

$$\boxed{d_{i+1} = d_i + 2(x_i + 1)\gamma_y^2 + (-2y_{i-1} + 1)\gamma_x^2} \quad \text{---(13)}$$

Now to find the actuall decesion parameter value at point $(\gamma_x, 0)$

now eq^n (9) becomes.

$$d_i = \left(x_i + \frac{1}{2}\right)^2 \gamma_y^2 + y_{i-1}^2 \gamma_x^2 - \gamma_x^2 \gamma_y^2$$

$$= \left(x_i^2 + x_i + \frac{1}{4}\right)\gamma_y^2 + (y_i^2 - 2y_i + 1)\gamma_x^2 - \gamma_x^2 \gamma_y^2$$

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

$= r_x^2 r_y^2 + r_x r_y^2 + \frac{1}{4} r_y^2 + r_x^2 - r_x^2 r_y^2$

$= r_x^2 + \frac{1}{4} r_y^2 + r_x r_y^2$

Ellipse Algorithm :-

1. Take input radius along x axis & y axis & obtain centre of Ellipse
2. Initially, we assume ellipse to be centred at origin & the 1st point as: $(x_0, y_0) = (0, r_y)$
3. Obtain the initial decision parameter for region 1 as $P_{10} = r_y^2 + \frac{1}{4} r_x^2 - r_x^2 r_y$
4. For every x_k position in region 1:
   If $(P_{1k} < 0)$ then the next point along the is (x_{k+1}, y_k) and $P_{1k+1} = P_{1k} + 2r_y^2 x_{k+1} + r_y^2$
   else, the next point is (x_{k+1}, y_{k-1})
   And $P_{1k+1} = P_{1k} + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_y^2$
5. Obtain the initial value in region 2 using the last point (x_0, y_0) of region 1 as
   $P_{20} = r_y^2 (x_0 + 1/2)^2 + r_x^2 (y_0 - 1)^2 - r_x^2 r_y^2$
6. At each y_k in region 2 starting at $k=0$ perform the following task.
   If $P_{2k} > 0$ the next point is (x_k, y_{k-1}) and $P_{2k+1} = P_{2k} - 2r_x^2 y_{k+1} + r_x^2$
7. Else, the next point is (x_{k+1}, y_{k-1}) and $P_{2k+1} = P_{2k} + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_x^2$
8. Now obtain the symmetric points in the 3 quadrants and plot the coordinate values:
   $x = x + x_c$, $y = y + y_c$
9. Repeat the steps for reg 1 until $2r_y^2 x \geq 2r_x^2 y$

Ex

To draw an Ellipse using 7 radii (8,6)

① $r_x = 8$, $r_y = 6$

② $(x, y) = (0, r_y) = (0, 6)$ Reg-1

$2r_y^2 x = 2 \times 36 \times 0 = 0$

$2r_x^2 y = 2 \times 64 \times 6 = 768$

$0 \geq 768$ ✗

③ $d = (6)^2 + \frac{1}{4} \times 64 - 64 \times 6$ (16)

$= 36 + 16 - 384$

$= 52 - 384 = -332$

④ $P1_0 \; (-332 < 0)$

$P_{next} = (-332) + 2 * 36 * 1 + 36$

$= -224$

K	Plot	d_i	x	y	$2r_y^2 y$
0	(1,6)	-332	1	6	768
1	(2,6)	-224	2	6	768
2	(3,6)	-44	3	6	768
3	(4,5)	208	4	5	640
4	(5,5)	108	5	5	640
5	(6,4)	288	6	4	512
6	(7,3)	244	7	3	384

N.B.

The last pixel value of Region 1 which is discarded due to condition check, so it jumps to 2nd Region by assuming initial pixel value.

Region 2

K	Plot	d_i	x	y	x^2	y^2
6	(7,3)	244	7	3	54	34
7	(8,2)	-23	8	2	576	256
8	(8,1)	617	8	1	576	128
9	(8,0)	533	8	0	576	0

Rough

$$\frac{225 \times 36}{4} + 64 \times 4 - 36 \times 8$$

$$= 2025 + 256 - 288$$

$$= 2281 - 288$$

$$= 1993$$

$$= -23 + 64$$

$$617 =$$

$\therefore \bar{x} = 12, \bar{y} = 10$

* To Draw an Ellipse using the Radii (12,10)

① $r_x = 12$; $r_y = 10$

② $(x,y) = (0, r_y) = (0,10)$

Reg-1

$2r_y^2 x = 2 \times 100 \times 0$
$= 0$

$2r_x^2 y = 2 \times 144 \times 10$
$= 2880$

$d_i = r_y^2 + \frac{1}{4} r_x^2 - r_x^2 r_y$
$= 100 + \frac{1}{4} \times 144 - 1440$
$= -1304$

If $(d_i < 0)$

$x = 1$, $y = 10$

$d_{i+1} = d_i + 2r_y^2 x + r_y^2$
$= -1304 + 200 + 100$
$= -1004$

K	Plot	d_i	x	y	$2r_y^2 x$	$2r_x^2 y$
1	(1,10)	-1304	1	10	200	2880
2	(2,10)	-1004	2	10	400	2880
3	(3,10)	-504	3	10	600	2880
4	(4,9)	196	4	9	800	2592
5	(5,9)	-1496	5	9	1000	2592
6	(6,9)	-396	6	9	1200	2592
7	(7,8)	904	7	8	1400	2304
8	(8,7)	100	8	7	1600	2016
9	(9,7)	-216	9	7	1800	2016
10	(10,6)	1684	10	6	2000	1728

Reg-2

K	Plot	d_i	x	y	$2r_y^2 x$	$2r_x^2 y$
10	(10,6)	1684	10	6	2000	1728
11	(10,5)	225	10	5	2050	10440
12	(11,4)	-1071	11	4	2200	1152
13	(11,3)	121	11	3	2200	864
14	(12,2)	-599	12	2	2400	576
15	(12,1)	1369	12	1	2400	288
16	(12,0)	1225	12	0	2400	0

Unit 3

2D Transformation

When any object to be performed different operation above new screen in 2D plane. that means "for changing object orientation, size & shape through co-ordinate Transformation".

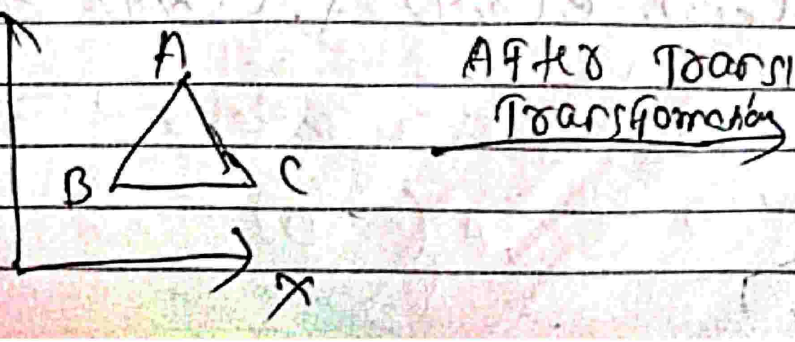
- In general way it used 3 method for forming 2D Transformation i.e.

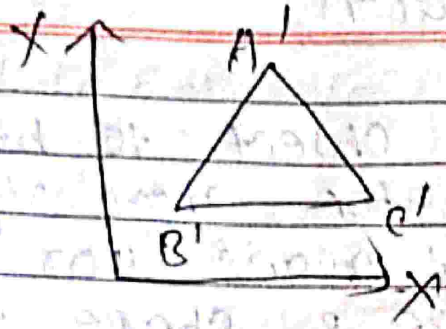
  (1) Translation
  (2) Scaling
  (3) Rotation.

- As the above 3 method performing only one action at a time so, another one new transformation method to be evolved i.e. composite Transformation
- Now a days, for speed up computation some other transformation method to be evolved i.e. Steaming & Reflection

(1) Translation :

This Transformation Translate all coordinate value from 1 location to another location in straight line format. that means for movable any object above same plane, it uses Translation Transformation





Here Every object to be initialised using 2 co-ordinate value i.e. (x, y). When the given object to be needs Translation Transformation then it used Translation vector value i.e. t_x & t_y.

Now to find the new co-ordinate value through which the transformation object to be build up i.e.

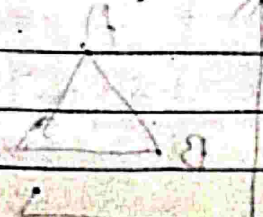
$$x' = x + t_x$$
$$y' = y + t_y$$

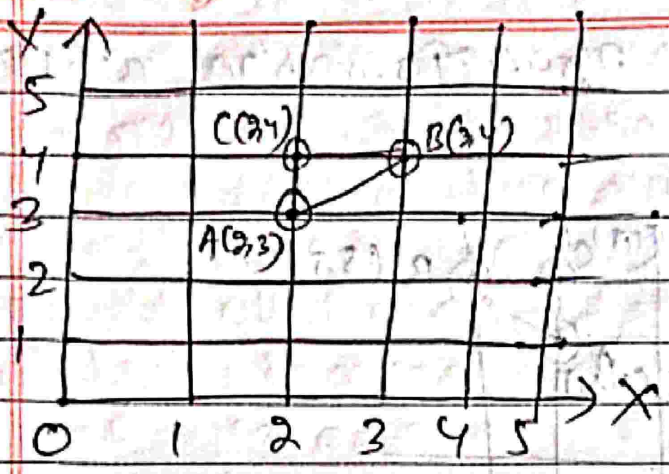
For optimisation representation, it use matrix concept.

$$P'\begin{pmatrix} x' \\ y' \end{pmatrix} = P\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

$$P'\begin{pmatrix} x' \\ y' \end{pmatrix} = P\begin{pmatrix} x+t_x \\ y+t_y \end{pmatrix}$$ Translation Matrix

Ex Let us perform Translation Transformation above the given co-ordinate values i.e. A(2,3), B(3,4) & C(2,4) with $t_x = 2$ & $t_y = 1$





Here the object contain 3 pixel value A(2,3), B(3,4), C(2,4) & $t_x = 2$, $t_y = 1$

A(2,3)

$x = 2$, $y = 3$, $t_x = 2$, $t_y = 1$

$x' = x + t_x = 2 + 2 = 4$

$y' = y + t_y = 3 + 1 = 4$

$A'(x', y') = A'(4,4)$

B(3,4)

$x' = x + t_x = 3 + 2 = 5$

$y' = y + t_y = 4 + 1 = 5$

$B'(x', y') = B'(5,5)$

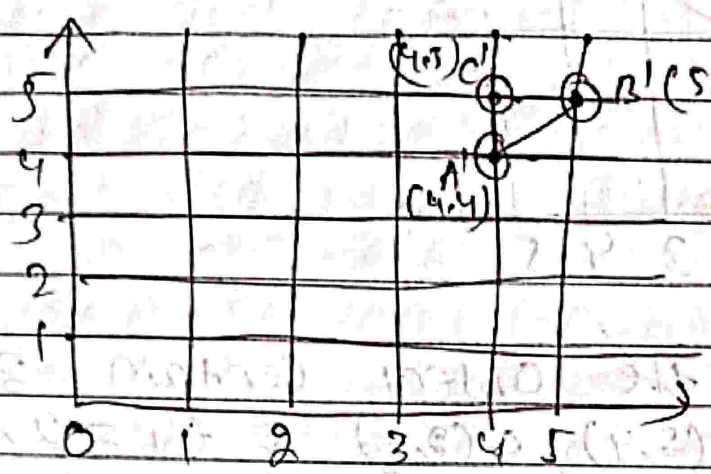
C(2,4)

$x' = 2 + 2 = 4$

$y' = 4 + 1 = 5$

$C'(x', y') = C'(4,5)$

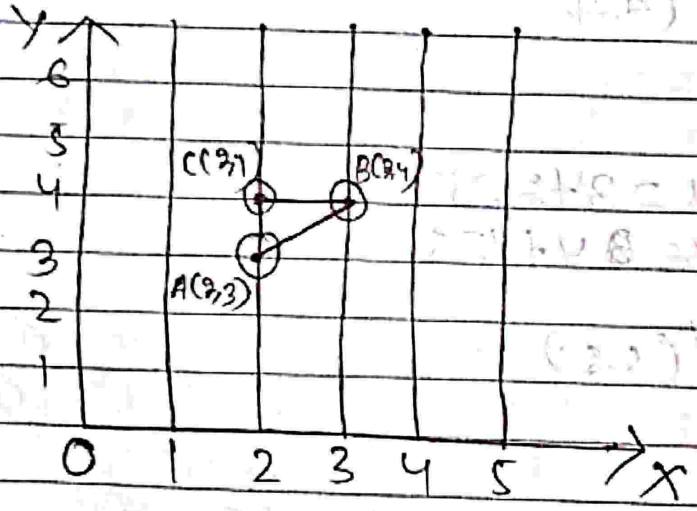
Now the new Transformation object is given below.



$t_x = -3$

$t_y = -3$

* Here the object contain 3 pixel value A(2,3), B(3,4), C(2,4) & $t_x = -3$ & $t_y = -3$



A(2,3)

$x = 2, \quad y = 3, \quad t_x = -3, \quad t_y = -3$

$x' = x + t_x = 2 - 3 = -1$

$y' = y + t_y = 3 - 3 = 0$

$A'(x', y') = A'(-1, 0)$

B(3,4)

$x=3$, $y=4$, $t_x=-3$, $t_y=-3$

$x' = x + t_x = 3-3 = 0$

$y' = y + t_y = 4-3 = 1$

$B'(x', y') = B'(0, 1)$

C(2,4)

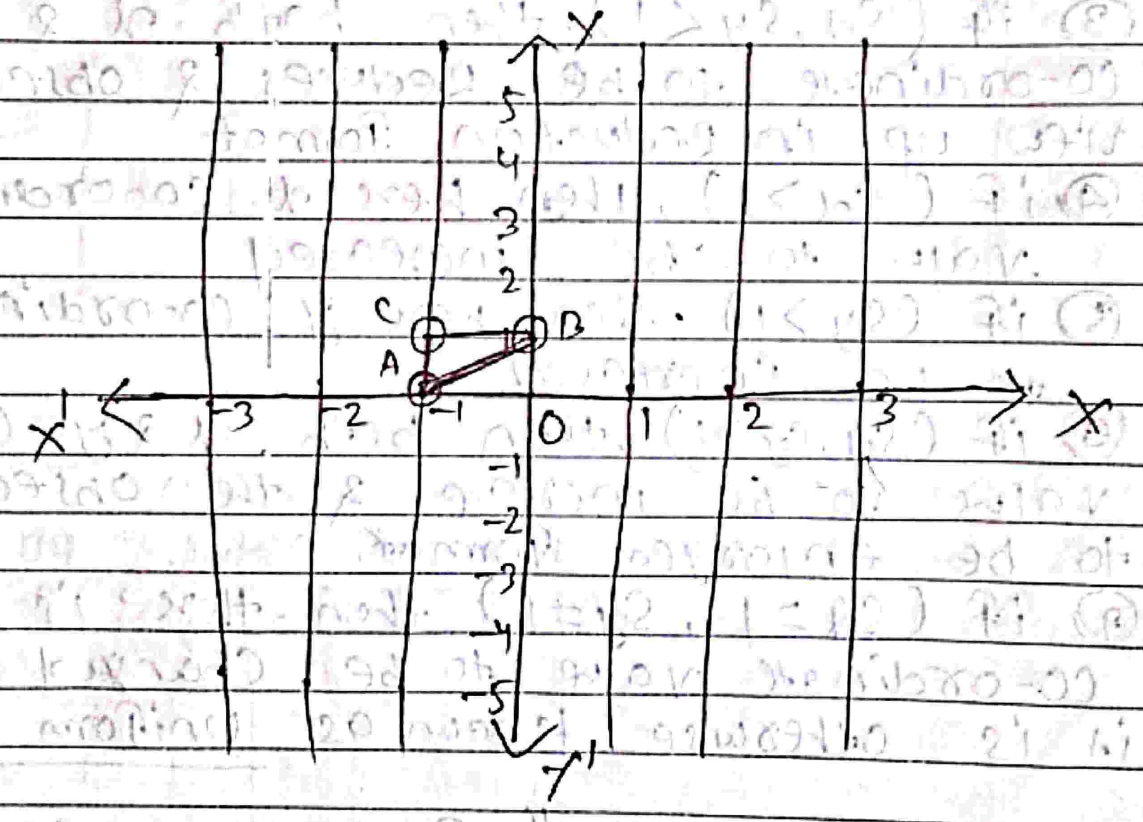
$x=2$, $y=4$, $t_x=-3$, $t_y=-3$

$x' = x + t_x = 2-3 = -1$

$y' = y + t_y = 4-3 = 1$.

$C'(x', y') = C'(-1, 1)$

Now the new Transformation object is given below.



Scaling :-

This transformation to be use for Altering object size in form of Reduction and Enlarged
- It uses scaling vector value through which all old co-ordinate value to be Repositioned in new co-ordinate format i.e. (S_x, S_y)
- It used different constraints for assigning scaling factor value.

① If $(S_x < 1)$, Here x co-ordinate value to be Reduces
② if $(S_y < 1)$, Here y coordinate value to be Reduces.
③ if $(S_x, S_y < 1)$, then both x & y co-ordinate to be Reduces & object to be view up in Reduction format
④ if $(S_x > 1)$, then here x co-ordinate value to be increased
⑤ if $(S_y > 1)$, then here y co-ordinate value to be increased.
⑥ if $(S_x, S_y > 1)$, then both x & y co ordinate value to be increase & the object size to be enlarged format.
⑦ if $(S_x = 1, S_y = 1)$ then there is no co-ordinate value to be. Changed so that it is otherwise known as Uniform scaling

In other words, "If both the scaling vector value to be equal i.e. called uniform scaling, otherwise non-uniform scaling"

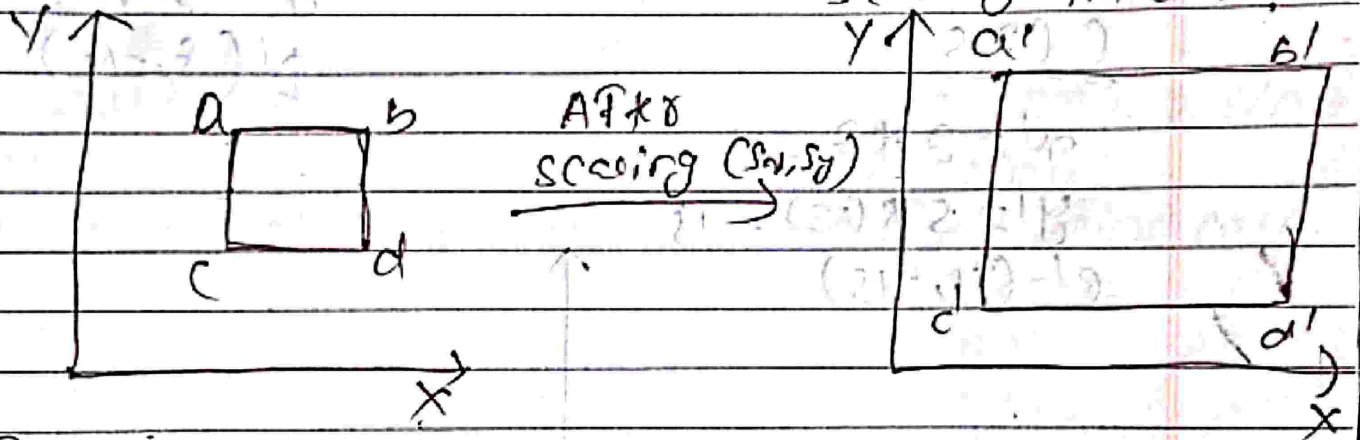
Let us take the old co-ordinate object value i.e. P(x,y). Here the scaling transformation vector value (S_x, S_y). Now to find the new transformation co-ordinate value

$$P\begin{pmatrix} x' \\ y' \end{pmatrix}, \quad x' = x \cdot S_x$$
$$y' = y \cdot S_y$$

The Matrix Representation of scaling Transformation is

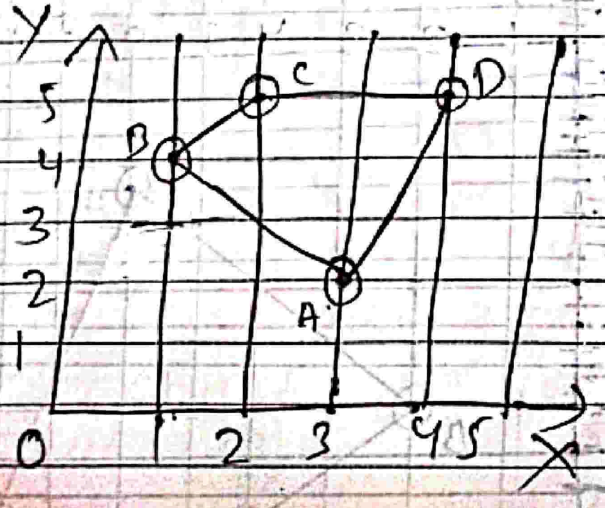
$$P\begin{pmatrix} x' \\ y' \end{pmatrix} = P\begin{pmatrix} x \\ y \end{pmatrix}\begin{pmatrix} S_x & 0 \\ 0 & S_y \end{pmatrix}$$

Scaling Matrix.



Ex

To perform scaling transformation of object co-ordinate point A(3,2) B(1,4), C(2,5), D(4,5) with $(S_x, S_y) = (2,3)$



A(3,2)

$x=3, y=2, s_x=2, s_y=-3$

$x' = x \cdot s_x = 3*2 = 6$

$y' = y \cdot s_x = 2*(-3) = -6$

$A'(x', y') = A'(6, -6)$

B(1,4)

$x' = 1*2 = 2$

$y' = 4 \times (-3) = -12$

$B'(x', y') = B'(2, -12)$

C(2,5)

$x' = 2*2 = 4$

$y' = 5*(-3) = -15$

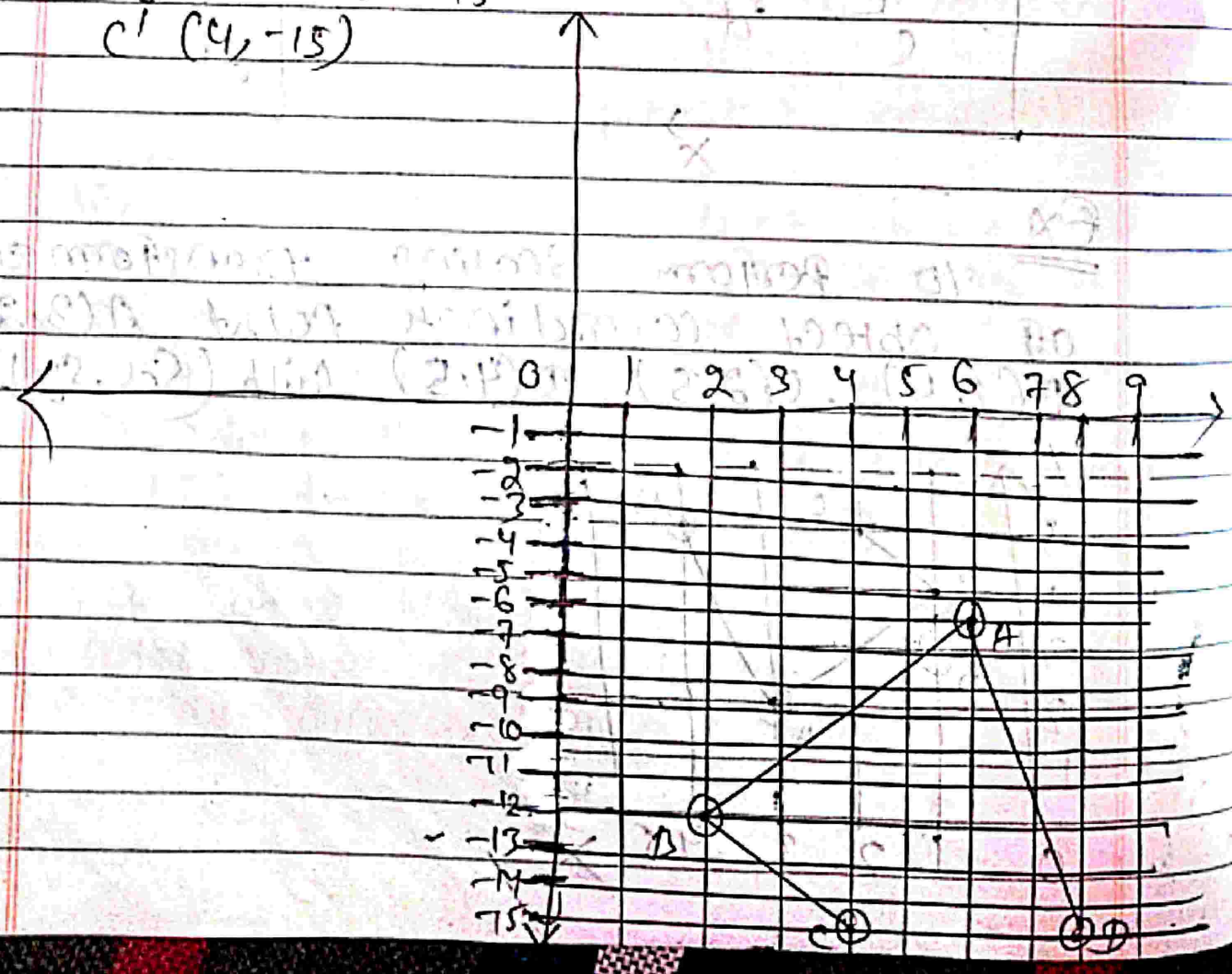
$C'(4, -15)$

D(4,5)

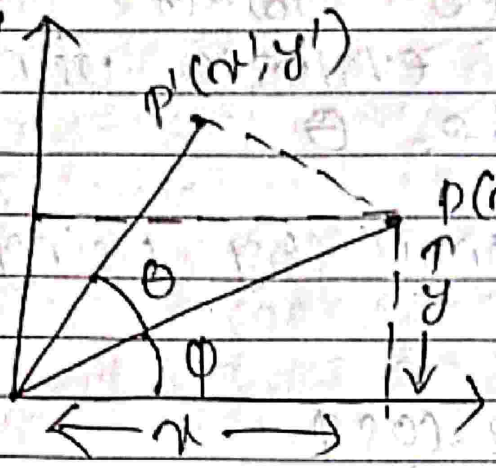
$x' = 4*2 = 8$

$y' = 5*(-3) = -15$

$D'(8, -15)$



Rotation :-



$$\frac{x}{r} = \cos\phi \Rightarrow x = r\cos\phi$$

$$\frac{y}{r} = \sin\phi \Rightarrow y = r\sin\phi$$

$$x' = r\cos(\phi+\theta) = r\cos\phi\cos\theta - r\sin\phi.\sin\theta$$
$$= x\cos\theta - y\sin\theta$$

$$y' = r\sin(\phi+\theta) = r\cos\phi\sin\theta + r\sin\phi\cos\theta$$
$$= x\sin\theta + y\cos\theta$$

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

Rotation Matrix

This transformation to be used for repositioning all object co-ordinate value from centre position to other location using some angle value

- That means, here the object do be rotate in clockwise & anti clock wise using some angle value
- Let us take the initial co-ordinate point of object i.e. (x,y) and the pivot angle value. (ϕ)

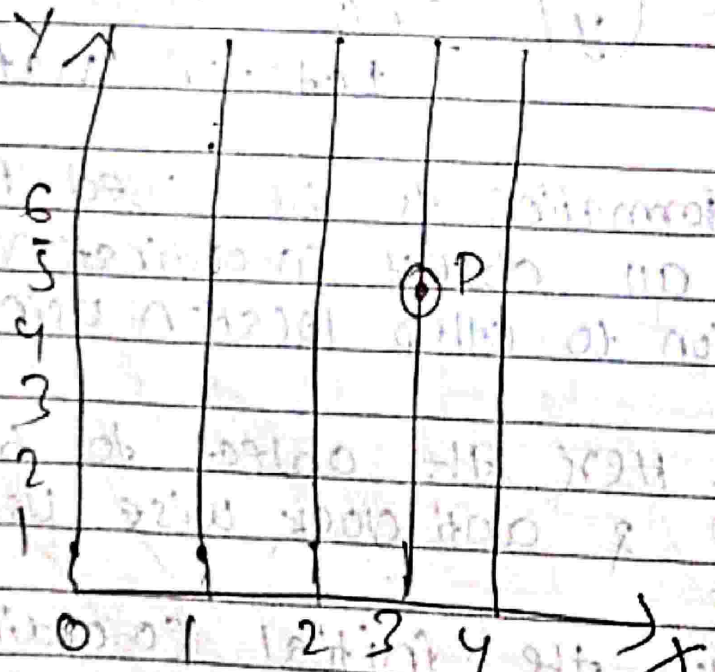
Now the new object co-ordinate value (x', y') to be evaluate using another angle value i.e. θ.

As we know by using Trigonometric func here

$$x = r\cos\phi$$
$$y = r\sin\phi$$

Ex
To perform Rotational Transformation of given object co-ordinate value (3,5) with $\theta = 90°$

Ans
Here, $P(x, y) = (3, 5)$
Angle $(\theta) = 90°$



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

$= 3\cos 90 - 5\sin 90$

$= 3 \times 0 - 5 \times 1$

$= 0 - 5 = -5$

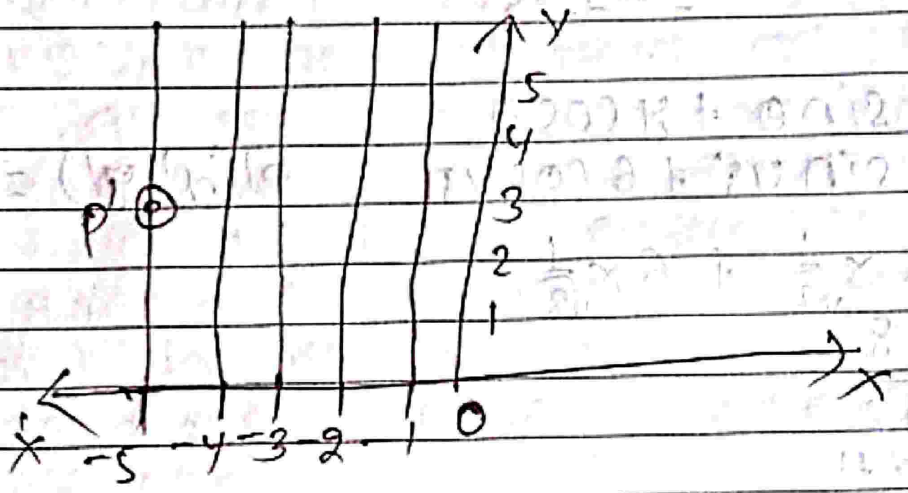
$y' = x\sin\theta + y\cos\theta$

$= 3\sin 90^\circ + 5\cos 90^\circ$

$= 3 \times 1 + 5 \times 0$

$= 3$

$P'(x', y') = (-5, 3)$



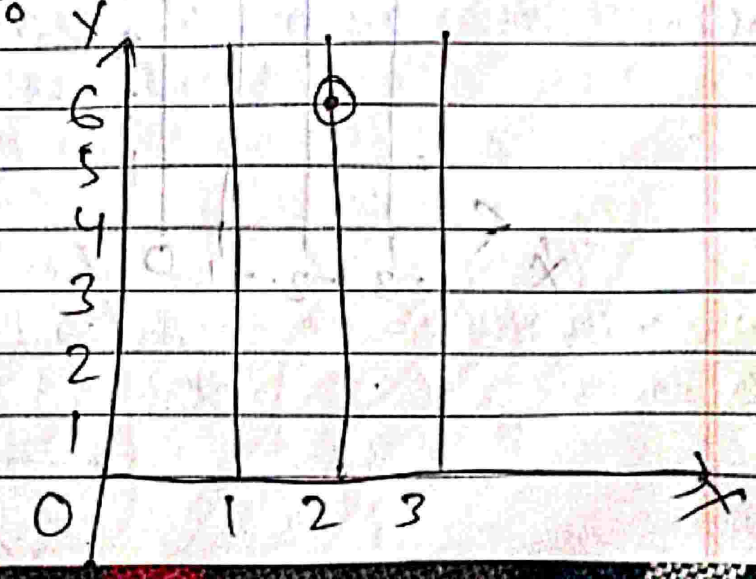
For clockwise Rotation

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

Ex

$P(x, y) = (2, 6)$

Angle $(\theta) = 45^\circ$



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

$= 2\cos 45 - 6\sin 45°$

$= 2 \cdot \frac{1}{\sqrt{2}} - 6 \cdot \frac{1}{\sqrt{2}}$

$= \frac{2 \times \sqrt{2}}{\sqrt{2} \times \sqrt{2}} - \frac{6 \times \sqrt{2}}{\sqrt{2} \times \sqrt{2}}$

$= \frac{2\sqrt{2}}{2} - \frac{6\sqrt{2}}{2}$

$= \sqrt{2} - 3\sqrt{2} \quad = 1.4 - 4.2$

$= -2.85$

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

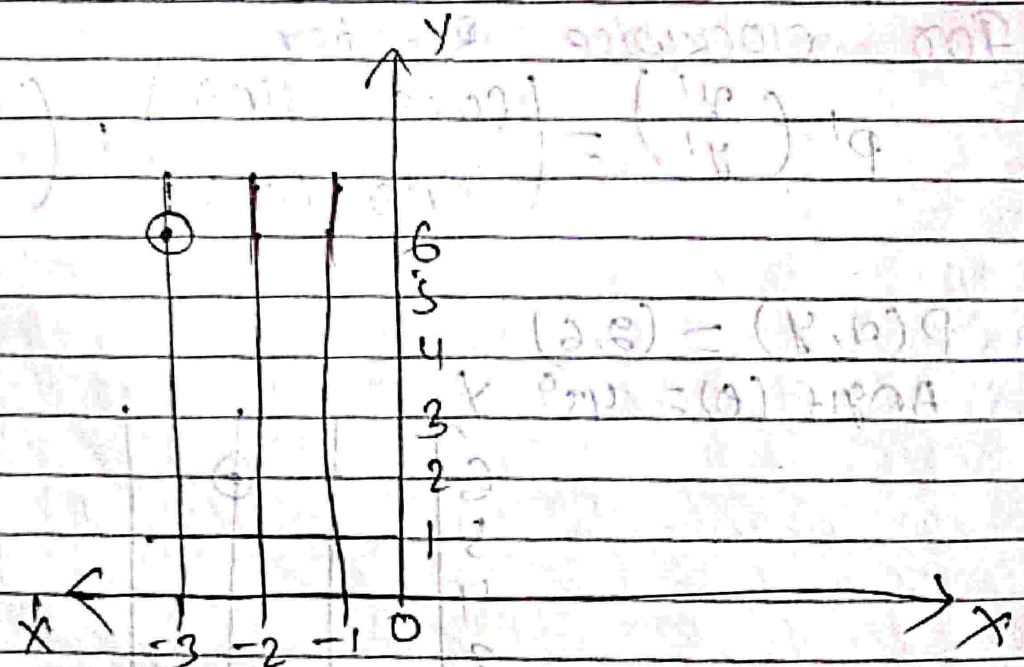
$= 2\sin 45° + 6\cos 45$

$= 2 \times \frac{1}{\sqrt{2}} + 6 \times \frac{1}{\sqrt{2}}$

$= \frac{8}{\sqrt{2}}$

$= 5.71$

$P'(x', y') = (-3, 6)$



Reflection

This is the most prominent Transformation technique, which is normally used for Plane conversion (i.e. 2D to 3D) & vice versa.

- When any object to be viewing its mirror format after the Transformation i.e. called Reflection Transmission.
- When any object to be viewing its opposite format by rotating 180° angle i.e. called reflection. Transformation

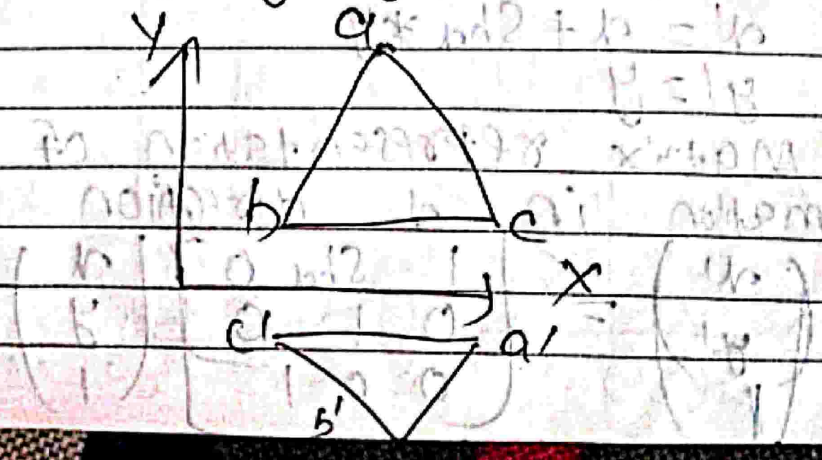
Points about Reflection :-

It use 3 protocols for its Transformation from old co-ordinate system to new co-ordinate system.

1. The old object to be view up xy plane or ⊥r to xy plane.
2. If the old object to be lies above xy plane, then the Reflection object to be lies above ⊥r xy plane
3. If the old object to be lies above ⊥r xy plane, then the Reflection object lies above xy plane

When

Ex.) the old object lies above xy plane, then now calculate its reflection matrix (only y rotate)



Old Matrix $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

Reflection matrix $\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

Shearing:-

This transformation to be specially used for 3D object view up. It distorts the object co-ordinate value for altering all position of given object to change its apperance, that means, the shape of the object in different apperance to be performed by using this transformation. It used 2 shearing vector value for changing the object apperance.

(1) With respect x co-ordinate:-

let us take the object co-ordinate point i.e. P(x,y), here the shearing vector value is Sh_x. Now to perform shearing transformation for calculating the new co-ordinate object value i.e. P(x',y'),

here $x' = x + Sh_x * y$

$y' = y$

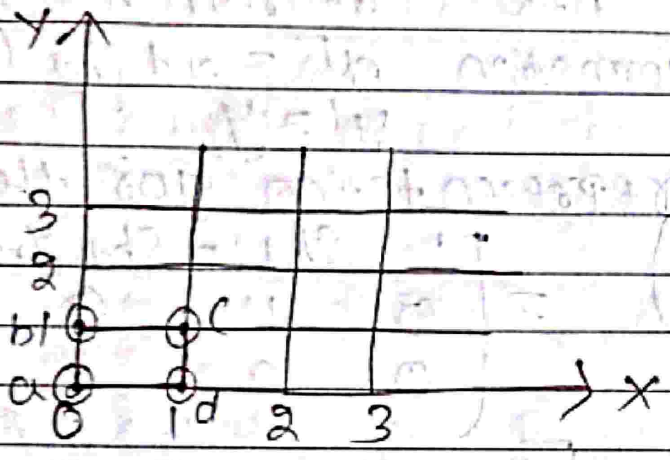
The Matrix representation of shearing transformation in x direction

$$P' \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & Sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

where $m = \begin{bmatrix} 1 & Sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Ex

To perform shearing transformation of given object where the pixel value are a(0,0), b(0,1), c(1,1), d(1,0) & here $Sh_x = 2$



a(0,0)

$x' = x + Sh_x * y$

$= 0 + 2 * 0 = 0$

$y' = 0$

$a'(x', y') = (0,0)$

b(0,1)

$x' = 0 + 2 * 1$

$= 2$

$y' = 1$

$b'(x', y') = (2,1)$

c(1,1)

$x' = 1 * 2 * 1$

$= 3$

$y' = 1$

$c'(x', y') = (3,1)$

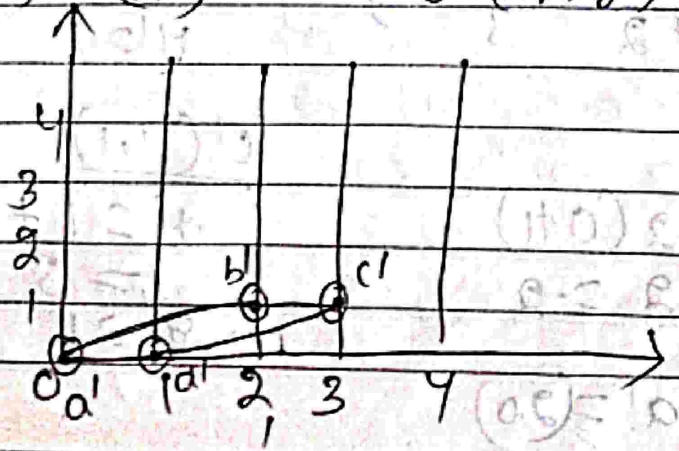
d(1,0)

$x' = 1 + 2 * 0$

$= 1$

$y' = 0$

$d'(x', y') = (1,0)$



As the shearing Transformation to be used for 3D object we, so here some of the co-ordinate value or object portion to be partially reflected. Now at the time of 2D shearing Transformation it assume the y reference value ($y_{ref} = -1$)

Now to find new co-ordinate value after shearing Transformation $x' = x + sh_x (y - y_{ref})$

$y' = y$

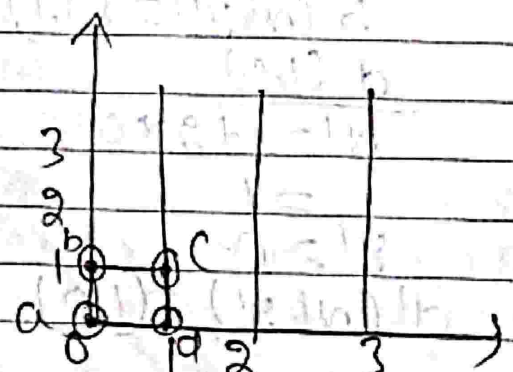
The Matrix representation for the matrix is

$$P' \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & sh_x & -sh_x \cdot y_{ref} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

where $m = \begin{bmatrix} 1 & sh_x & -sh_x \cdot y_{ref} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Ex a(0,0), b(0,1), c(1,1), d(1,0), $y_{ref} = -1$

$sh_x = 2$



a'(0,0)

$x' = 0 + 2(0+1)$

$= 0 - 2 = -2$

$y' = 0$

$a' = (2,0)$

b'(0,1)

$x' = 0 + 2(1+1)$

$= 4$

$y' = 1$ b'(4,1)

c'(1,1)

$x' = 1 + 2(1+1)$

$= 5$

$y' = 1$ c'(5,1)

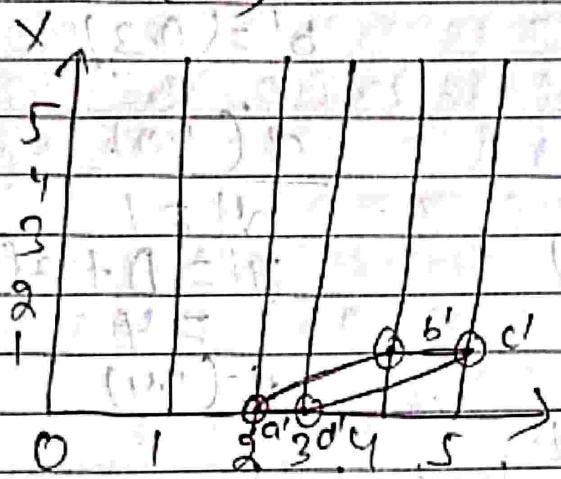
d'(1,0)

$x' = 1 + 2(0+1)$

$= 3$

$y' = 0$

d'(3,0)



(2) With respect to Y co-ordinate

Let us take, the object pixel value i.e P(x,y) & hear the shearing vector value shy. Now to perform shearing Transformation for calculating new Co-ordinate Object value i.e. P(x',y')

here $x' = x$ $x_{ref} = -1$

$y' = y + shy(x - x_{ref})$

$$P'\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ shy & 1 & -shy.x_{ref} \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

where $m = \begin{bmatrix} 1 & 0 & 0 \\ shy & 1 & -shy.x_{ref} \\ 0 & 0 & 1 \end{bmatrix}$

Ex $x_{ref} = -1$ $sh_y = 2$

a(0,0)

$x' = 0$

$y' = 0 + 2(0+1)$

$= 2$

$a'(0,2)$

b(0,1)

$x' = 0$

$y' = 1 + 2(0+1)$

$= 3$

$b' = (0,3)$

c(1,1)

$x' = 1$

$y' = 1 + 2(1+1)$

$= 5$

$c' = (1,5)$

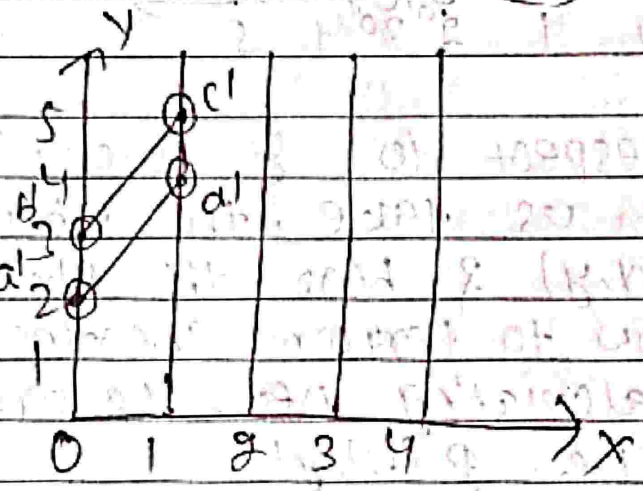
d(1,0)

$x' = 1$

$y' = 0 + 2(1+1)$

$= 4$

$d' = (1,4)$



Homogeneous co-ordinate Transformation:

- It is used to convert all 2D data value in 3D vector format
- The objective of this transformation is to convert 1 plane to other plane co-ordinate system
- It specially used for viewing all 2D object in an efficient way above 3D plane
- In general way, 2D shearing & reflection transformation to be used homogeneous coor

in direct ways.

Now to apply Homogeneous co-ordinate concept above Translation, scaling & Rotation Transformation

Translation :-

The 2D Translation Transformation matrix is

$$M = \begin{bmatrix} x+t_x \\ y+t_y \end{bmatrix}$$

After Homogeneous co-ordinate Transformation the matrix is

$$P \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Scaling :-

The 2D scaling Transformation matrix is

$$M = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix}$$

After Homogeneous co-ordinate Transformation

$$P \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Rotation :-

The 2D Rotation Transformation matrix is

$$M = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

After Homogeneous co-ordinate Transformation

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

Composite Transformation :-

This transformation to be used for providing multiple operation simultaneously above given object.

- For solve any complex object manipulation in different form, it used composite transformation

"The combination of Translation, Scaling & Rotational transformation is known as composite transformation"

It require 2 successive vector value & Angle value through which transformation to be perform.

Translation :-

2 successive translation vector of pixel value $P(x,y)$ i.e. (tx_1, ty_1) & (tx_2, ty_2)

now the new co-ordinate value

$$x' = x + (tx_1 + tx_2)$$
$$y' = y + (ty_1 + ty_2)$$

$$M = \begin{bmatrix} 1 & 0 & tx_1 + tx_2 \\ 0 & 1 & ty_1 + ty_2 \\ 0 & 0 & 1 \end{bmatrix}$$

Scaling :-

2 successive scaling vector of pixel value $P(x,y)$ i.e. (Sx_1, Sy_1) & (Sx_2, Sy_2)

now the new co-ordinate value

$$x' = x * (Sx_1 * Sx_2)$$
$$y' = y * (Sy_1 * Sy_2)$$

$$M = \begin{bmatrix} Sx_1 * Sx_2 & 0 & 0 \\ 0 & Sy_1 * Sy_2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation

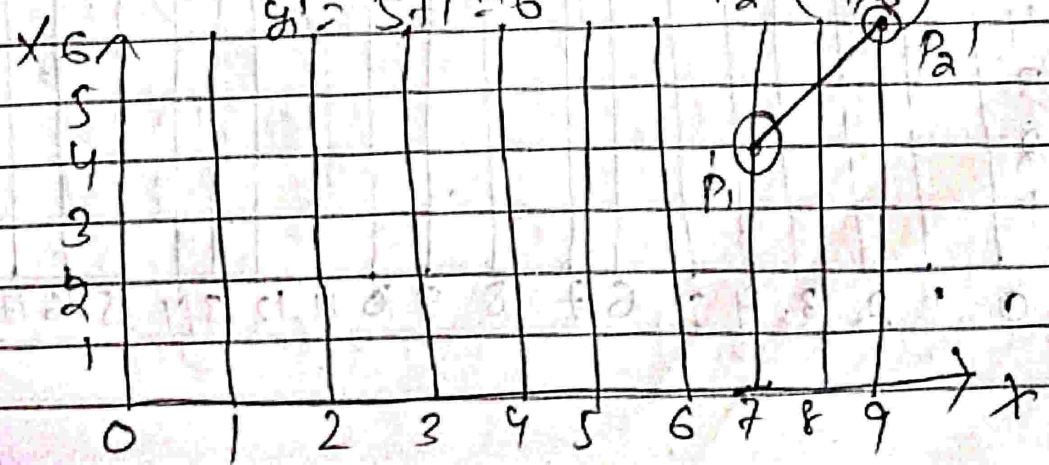
2 successive pivot rotation angle value of pixel $P(x,y)$ i.e. (θ_1, θ_2)
now calculate total pivot angle value $\theta_1 + \theta_2$
so the new co-ordinate transformation value

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

$$M_1 = \begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) & 0 \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Ex To perform composite transformation of given object where the pixel value $P_1(4,3)$ $P_2(6,5)$, where $(tx_1, ty_1) = (2,1)$
$(tx_2, ty_2) = (1,0)$
$(sx_1, sy_1) = (2,2)$
$(sx_2, sy_2) = (2,2)$
$\theta_1 = 90^\circ$ & $\theta_2 = 90^\circ$

Translation:-
$(tx_1, ty_1) = (2,1)$ & $(tx_2, ty_2) = (1,0)$
new co-ordinate value
$x_1' = 4 + (2+1) = 7$
$y_1' = 3 + (1+0) = 4$ $P_1'(7,4)$
for P_2 $x_2' = 6+3 = 9$
$y_2' = 5+1 = 6$



scaling

$P_1(x, y) = (4, 3)$

$P_2(x, y) = (6, 5)$

$(S_{x_1}, S_{y_1}) = (2, 2)$ $(S_{x_2}, S_{y_2}) = (2, 2)$

new co-ordinate value P_1

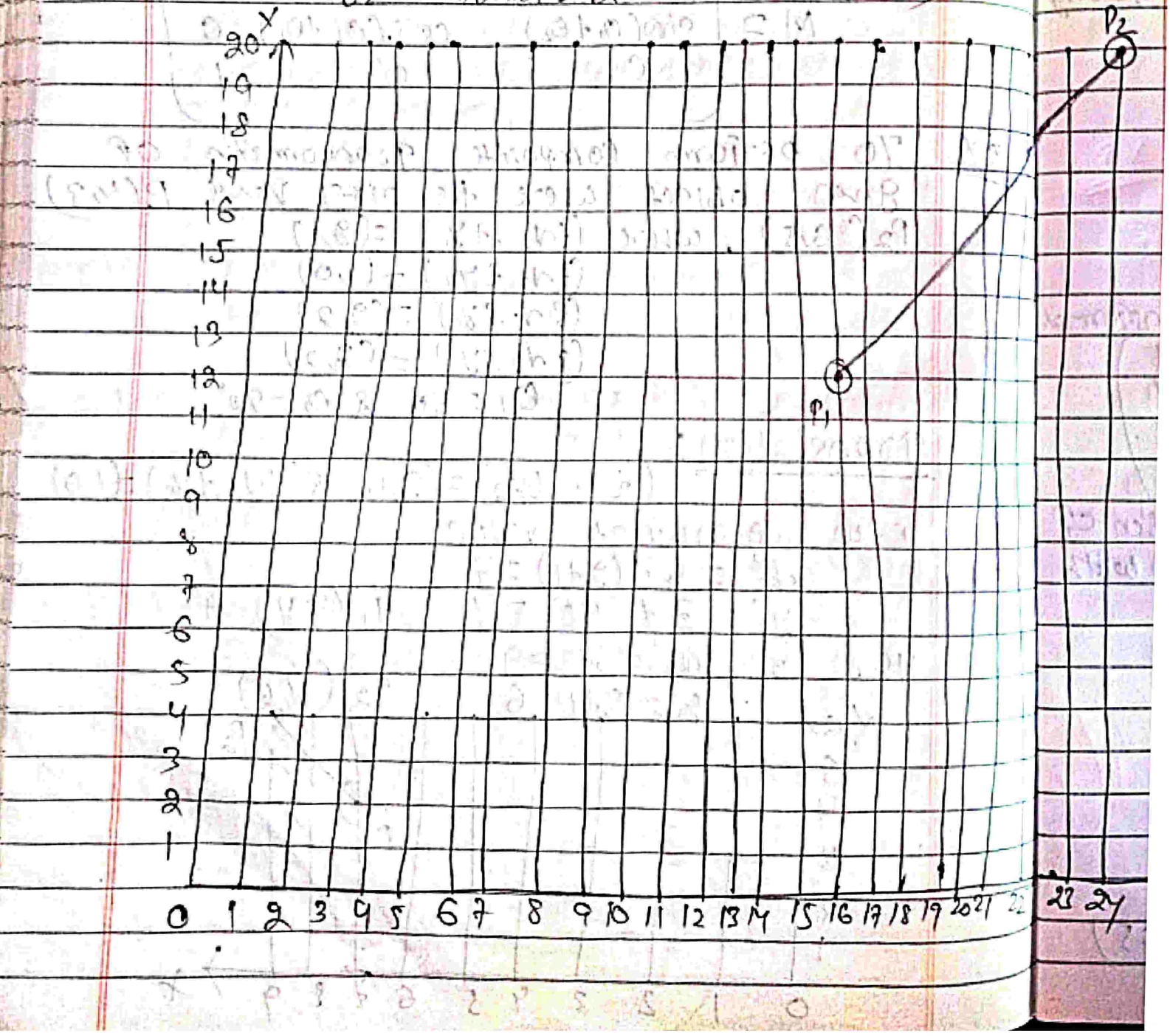
$x_1' = 4 * 2 * 2 = 16$

$y_1' = 3 * 2 * 2 = 12$ $\quad P_1'(16, 12)$

for P_2

$x_2' = 6 * 2 * 2 = 24$

$y_2' = * 2 * 2 = 20$ $\quad \therefore P_2'(24, 20)$



Rotation

$P_1(x,y) = (4,3)$ $\theta_1 = \theta_2 = 90°$

$P_2(x,y) = (6,5)$

Pivot angle $\theta = \theta_1 + \theta_2 = 180°$

new co-ordinate value for P_1

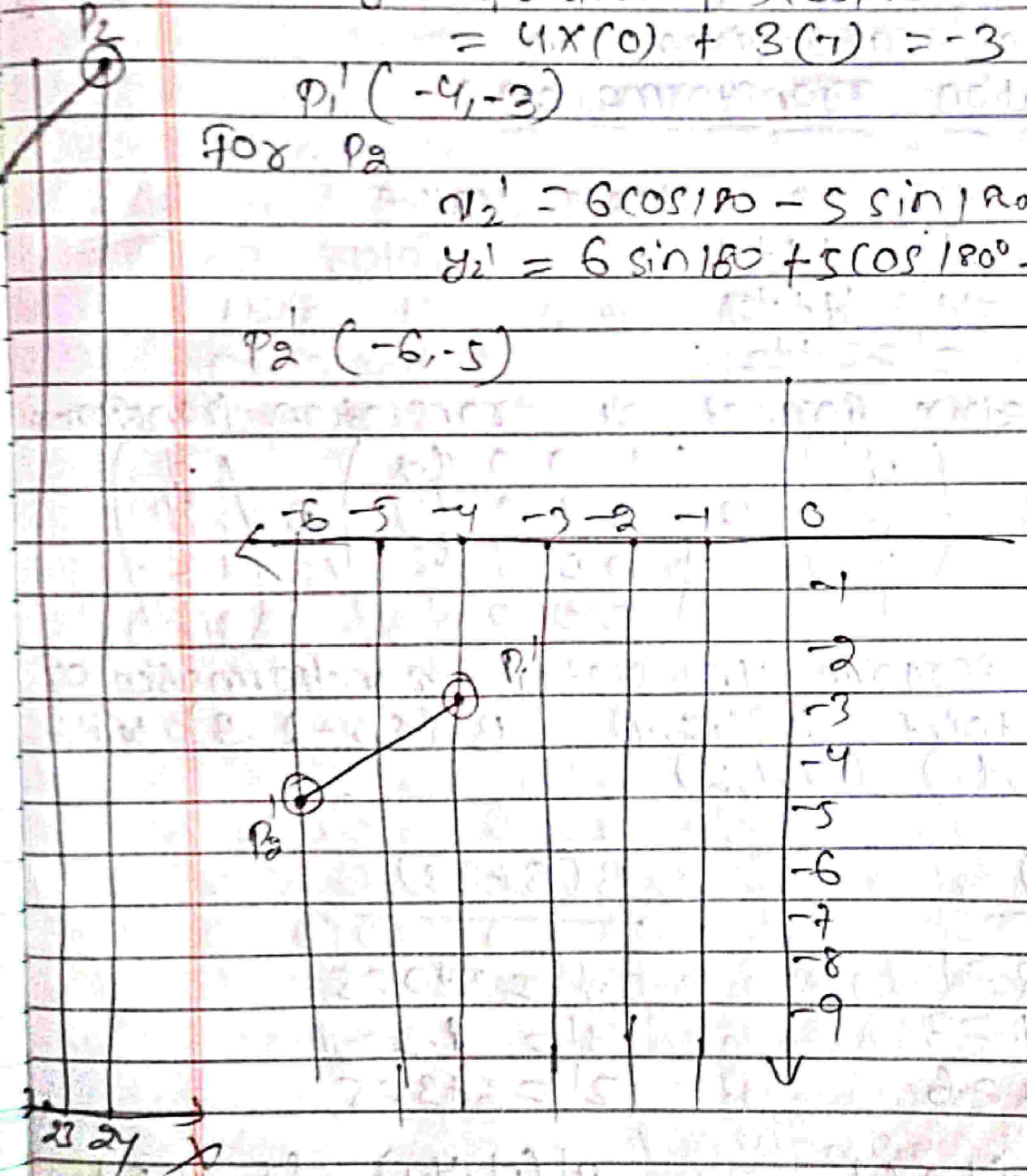
$$x' = x\cos\theta - y\sin\theta$$
$$= 4\cos 180 - 3\sin 180°$$
$$= 4(-1) - 3\times 0 = -4$$
$$y' = 4\sin 180 + 3\cos 180$$
$$= 4\times(0) + 3(-1) = -3$$

$P_1'(-4,-3)$

for P_2

$$x_2' = 6\cos 180 - 5\sin 180 = 6(-1) - 5(0) = -6$$
$$y_2' = 6\sin 180 + 5\cos 180° = 6(0) + 5(-1) = -5$$

$P_2'(-6,-5)$



3D Transformation :

Like 2D Transformation, here the object to be altering its co-ordinate value in different format like size, shape, location etc form.

It uses 3-coordinate value i.e P(x, y, z) and another one homogeneous co-ordinate value for plane transformation

It also performed composite transformation using the combination of translation, scaling, Rotational transformation.

Translation Transformation :

The new co-ordinate value

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

The matrix format of translation transformation is

$$P' = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} P \quad \text{where } P = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}, \; P' = \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix}$$

Ex To perform translation transformation of pixel point A(1, 2, 4) B(5, -2, 3) with (t_x, t_y, t_z) (0, 1, 2)

A(1, 2, 4)

$x' = 1 + 0 = 1$
$y' = 2 + 1 = 3$
$z' = 4 + 2 = 6$

$A'(x, y, z) = (1, 3, 6)$

B(5, -2, 3)

$x' = 5 + 0 = 5$
$y' = 1 - 2 = -1$
$z' = 2 + 3 = 5$

$B'(x, y, z) = (5, -1, 5)$

Scaling

The new co-ordinate value.

$x' = x * Sx$

$y' = y * Sy$

$z' = z * Sz$

The matrix formed of scaling transformation.

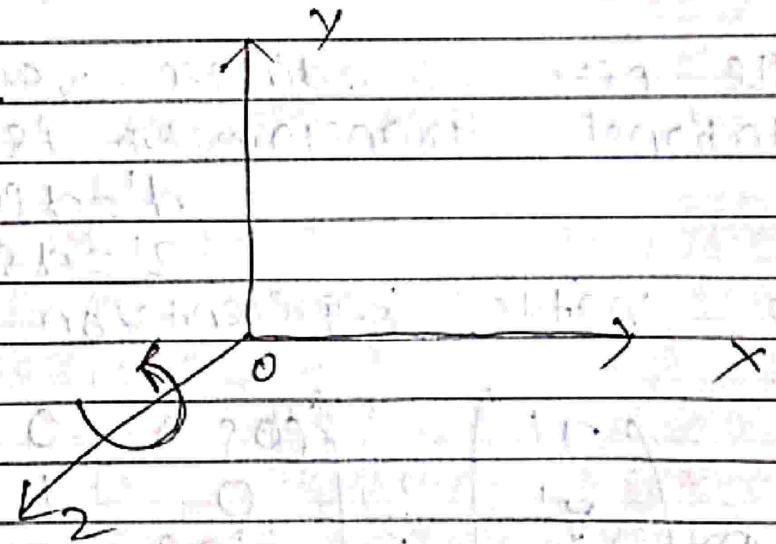
$$P\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} P\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

M.

Rotational :

Here, any one co-ordinate to be rotate using some pivot angle value, then the given co-ordinate point to be constant & the other to be Altered.

Rotation about z axis :-



The new co-ordinate elements after the rotational transformation $z' = z$

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

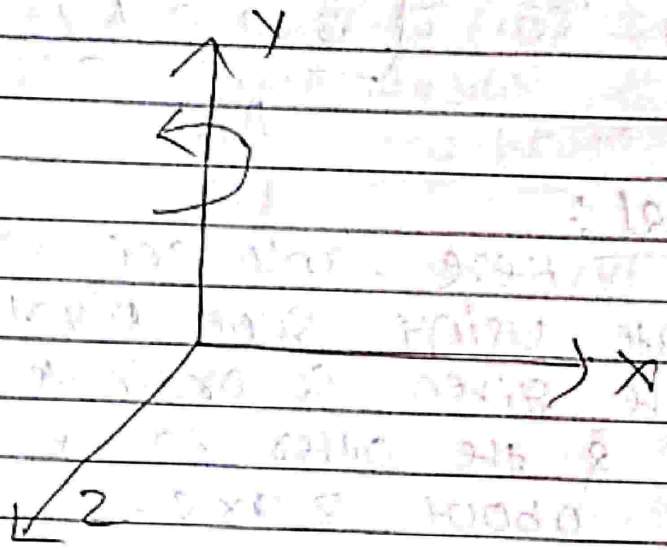
$y' = x\sin\theta + y\cos\theta$

The matrix representation about z axis

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

M

About y axis Rotation



The new co-ordinate value after the rotational transformation is $y' = y$

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

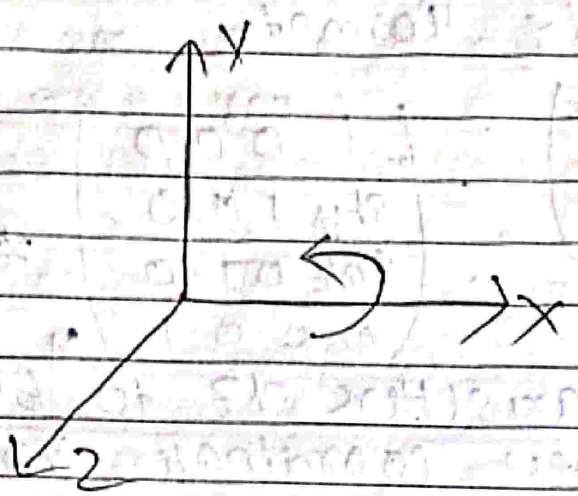
$$z' = x\sin\theta + z\cos\theta$$

The matrix representation about y axis

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

M

About x axis's Rotation



The new co-ordinate value after the rotational transformation $x' = x$

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

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

The matrix representation about x axis

$$P\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{M} P\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Shearing Transformation :-

Here the shearing transformation also possible by using uni-direction / axis's coordinate format.

Along x axis's (Here yz axis's to be preserved)

The new co-ordinate value to be calculate using a vector value w.r.t to x axis's i.e Sh_y & Sh_z

$\Rightarrow x' = x$

$y' = x * Sh_y + y$

$z' = x * Sh_y + z$

The matrix formed

$$P' \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ sh_y & 1 & 0 & 0 \\ sh_z & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{M} * P \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Along y axis (Here xz to be preserved)
The new coordinate value to be calculate by using a vector value w.r.t to y i.e. sh_1, sh_2

$$x' = x + y * sh_1$$
$$y' = y$$
$$z' = z + y.sh_2$$

$$P' \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & sh_1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & sh_2 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{M} * P \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Along z axis's (Here xy to be preserved)

$$x' = x + z * sh_1$$
$$y' = y + z * sh_y$$
$$z' = z$$

$$P' \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & sh_1 & 0 \\ 0 & 1 & sh_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{M} * P \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Reflection

Like 2D Transformation, here the given object also to be reflected in axis formed. But, as here the object to be viewing above 3D plane (some portions to be eliminated). So that the reflection portion to be measure using plane co-ordinate formed

Reflection about xy-plane

The Reflection matrix is

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\Rightarrow x' = x$

$\Rightarrow y' = y$

$\Rightarrow z' = -z$

yz-plane

The reflection matrix is

$$M = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\Rightarrow x' = -x$

$\Rightarrow y' = y$

$\Rightarrow z' = z$

xz-plane

The reflection matrix is

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\Rightarrow x' = x$

$\Rightarrow y' = -y$

$\Rightarrow z' = z$

Unit-4 :-

2D viewing :-

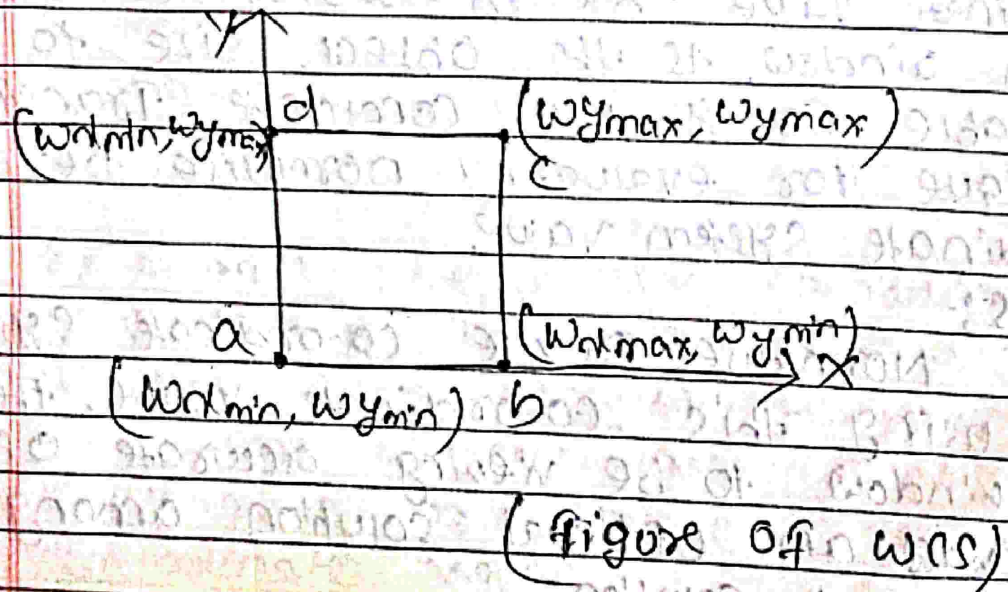
In traditionally, multiple systems had different monitor size. But when any object to be needs processing above the given monitor screen, here sometimes some object portion to be lost so that one initiative concept to be introduce for avoiding object lost above monitor screen when it view up.

The 2D viewing concept use 4 parameter for transforming world co-ordinate system value to view port co-ordinate system value.

WCS / WCM / WVS / WP :-

world co-ordinate system
world co-ordinate monitor
world view system.
world port.

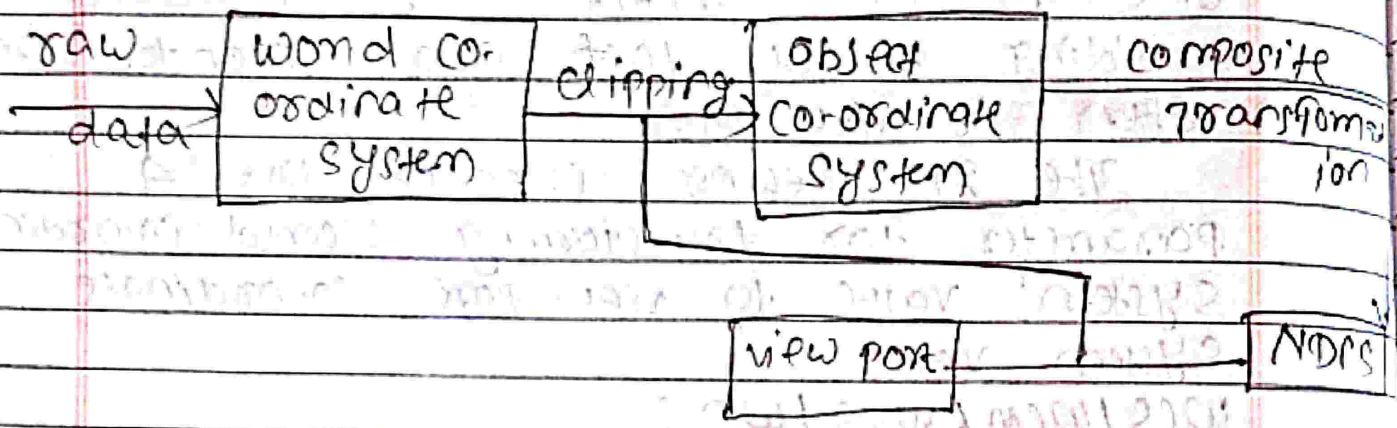
When the object constructed initially using its environmental value above any system i.e. called WCS



(figure of WCS)

The shape of wcs to be looks like rectangular or square form. It contain 4 co-ordinate value i.e. (Wxmin, Wymin, Wxmax, Wymax)

Conversion of 2D viewing:-



To perform all outer environment data, the programmer build up any object using wond co-ordinate system. As the view port screen size is very small or very large to compard as wond co-ordinate (windows co-ordinate) system so it used clipping technique

The clipping technique calculate object co-ordinate value through wond co-ordinate system window. As the object size to be changeable; so it use composite Transformation technique for evaluating normalise Device co-ordinate system value.

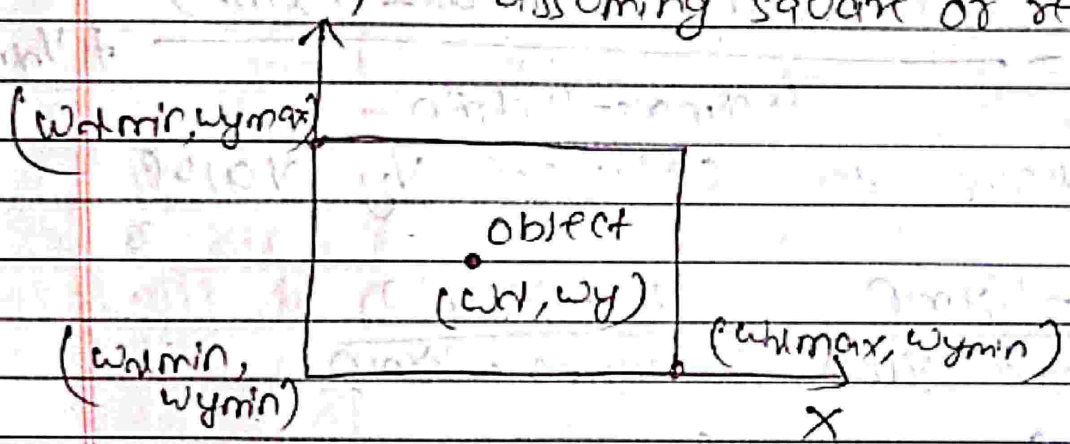
NDCS:-

Normalise Device co-ordinate system By using this co-ordinate value, the view-port window to be viewing accurate object. It provides an optimal solution among several path solution.

It also calculate all co-ordinate value as the view port appearance, (2D or 3D) By using NDCS value & clipping calculation value then the view port viewing accurate outcomes above any output screen.

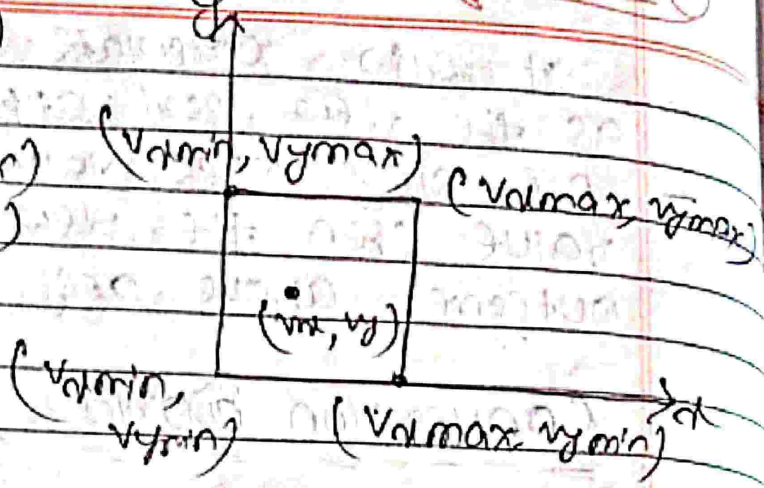
Conversion World co-ordinate system to view port:-

As the world co-ordinate system Environment contain 4 co-ordinate value. i.e. (Wxmin, Wymin)
(Wxmax, Wymin)
(Wxmax, Wymax)
(Wxmin, Wymax) so that its shape assuming square or rectangular shape



Here the object co-ordinate value is P(Wx, Wy). But the given view port window's size are different than WCS. Here view port windows also looks like rectangular or square shape with 4 co-ordinate value.

i.e. (V_{xmin}, V_{ymin})
(V_{xmax}, V_{ymin})
(V_{xmax}, V_{ymax})
(V_{xmin}, V_{ymax})



Now apply TSR (Translation, Scaling, Rotation) mechanism for viewing WCS object into view port object formed i.e.

$$\frac{W_x - W_{xmin}}{W_{xmax} - W_{xmin}} = \frac{V_x - V_{xmin}}{V_{xmax} - V_{xmin}}$$

$$\Rightarrow V_x = \frac{(W_x - W_{xmin})(V_{xmax} - V_{xmin})}{W_{xmax} - W_{xmin}} + V_{xmin}$$

similarly do calculate V_y value

$$\frac{W_y - W_{ymin}}{W_{ymax} - W_{ymin}} = \frac{V_y - V_{ymin}}{V_{ymax} - V_{ymin}}$$

$$V_y = \frac{(W_y - W_{ymin})(V_{ymax} - V_{ymin})}{W_{ymax} - W_{ymin}} + V_{ymin}$$

now the clipping mechanism to be over, if the view port object to be require any composit transformation then it apply TSR mechanism.

(T)(S)(T^-1R)

$$\begin{pmatrix} 1 & 0 & V_x \\ 0 & 1 & V_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -W_x \\ 0 & 1 & -W_y \\ 0 & 0 & 1 \end{pmatrix}$$

where

$$S_x = \frac{(W_x - W_{x\min})(V_{x\max} - V_{x\min})}{(W_{x\max} - W_{x\min})}$$

$$S_y = \frac{(W_y - W_{y\min})(V_{y\max} - V_{y\min})}{(W_{y\max} - W_{y\min})}$$

Ex $P(3,4) = (w_x, w_y)$

$w_x max = 7$ $\quad v_x max = 2$

$w_x min = 0$ $\quad v_x min = 0$

$w_y max = 5$ $\quad v_y max = 3$

$w_y min = 0$ $\quad v_y min = 0$

$$v_x = \frac{(3-0)\,(2-0)}{(7-0)} \to 0$$

$$= \frac{6}{7} = 0.8$$

$$v_y = \frac{(4-0)\,(3-0)}{(5-0)} \to 0$$

$$= \frac{12}{5} = 2.4$$

$(T)(S)(T|R)$

$$\begin{pmatrix} 1 & 0 & 0.8 \\ 0 & 1 & 2.4 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0.8 & 0 & 0 \\ 0 & 2.4 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -3 \\ 0 & 1 & -4 \\ 0 & 0 & 1 \end{pmatrix}$$

Clipping :-

It is a general methods, which is used for view of object above view port co-ordinate system in an appropriate way.

"It is a simply procedure, which identifies the position or portion of window spare above view port windows, where the object to be view up."

This mechanism to be execute by using 3 methods

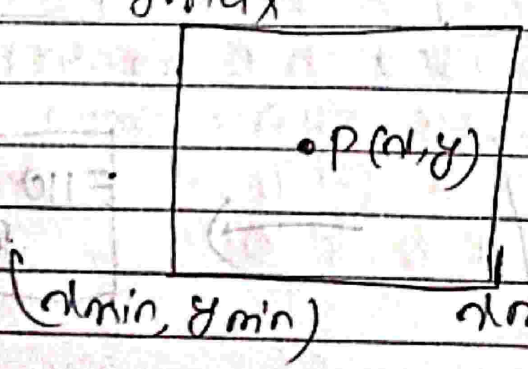
(1) visible :
If the object to be fully present inside the clipping window i.e. visible

(2) invisible :
If the object to be fully present outside the clipping window i.e. invisible

(3) clipped :
If the object to be partially intercept the given clipping window i.e. clipped.

There have several Algorithms to be used for different object to be view up above view port window through clipping mechanism

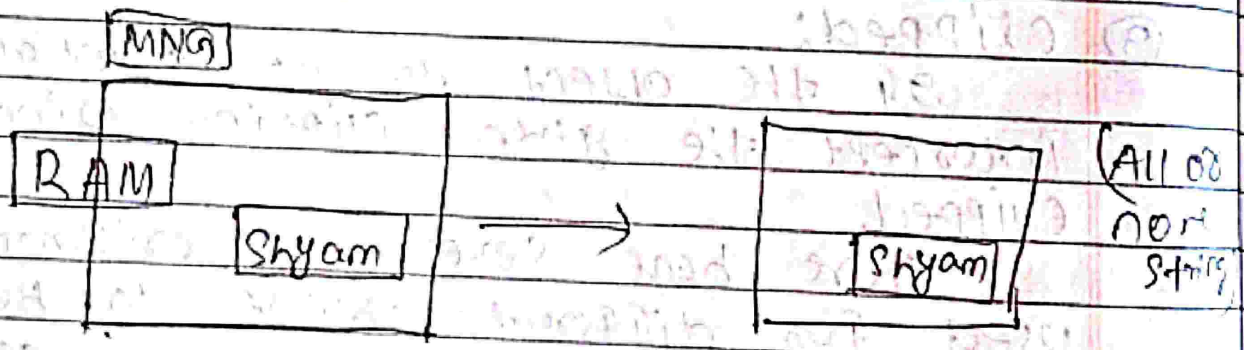
(1) Point : Clipping



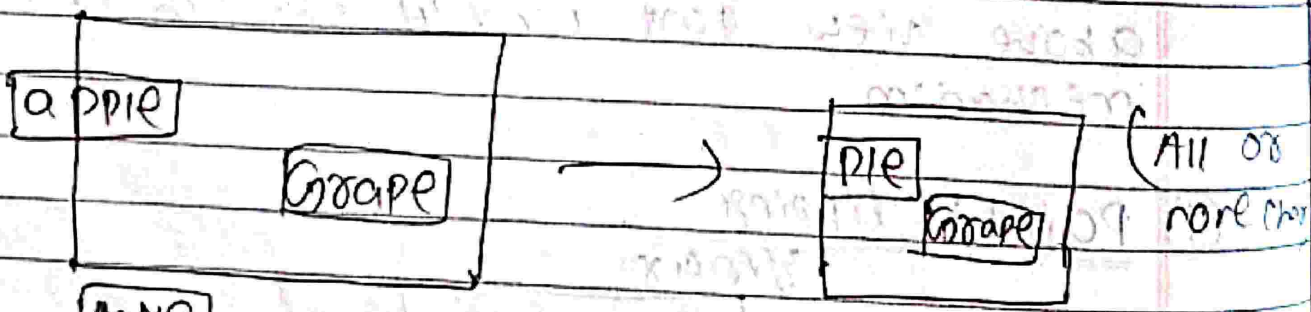
- It is a dot point or pixel, which looks like dust particle.
- It has no dimension.
- As the point to be execute by using 2 co-ordinate values i.e. P(x,y), so it required rectangular windows for performing clipping mechanism.
- The windows co-ordinates are V_xmax, V_xmin, V_ymax, V_ymin.
- The point to be clipped by using 2 constraints i.e. ① $V_xmin \leq x \leq V_xmax$
② $V_ymin \leq y \leq V_ymax$
- It satisfies visible & invisible condition of clipping window.

Text clipping :-

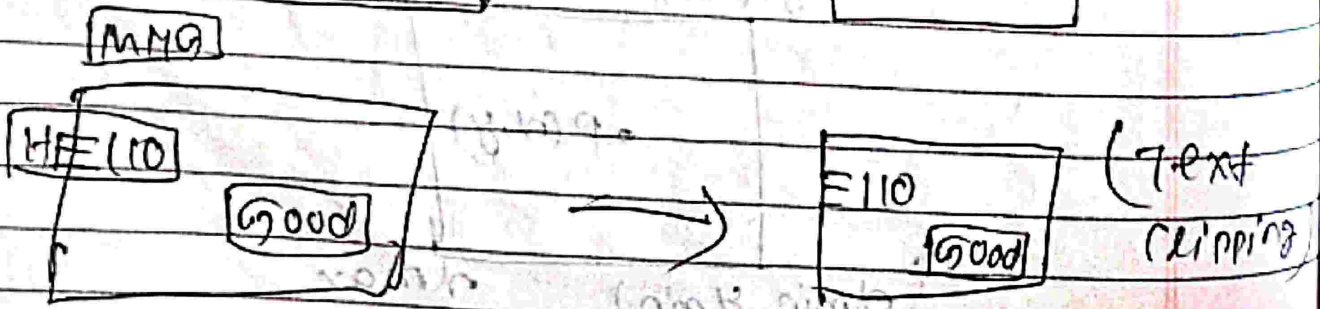
(I)



(II)



(III)



The text also one object, which build up using number of dot point. when this object to be viewing above view port screen from world co-ordinate system then it use clipping mechanism.

- The text to be created by using number of character or string, it to be clipping using 3 methods

(I) All or nore string

This method applies if any string that to be fully presented in between windows, then it accepted & viewing above view port screen otherwise discarded.

(II) All or nore char

Here the clipping mechanism use ore constraint of the given char to be fully present inside the windows then it accepted otherwise discarded.

(III) Text clipping

Here the clipping mechanism to be accept all character or string or sentence which present inside the windows and the partial part which intersect the windows boundary.

Line clipping :-

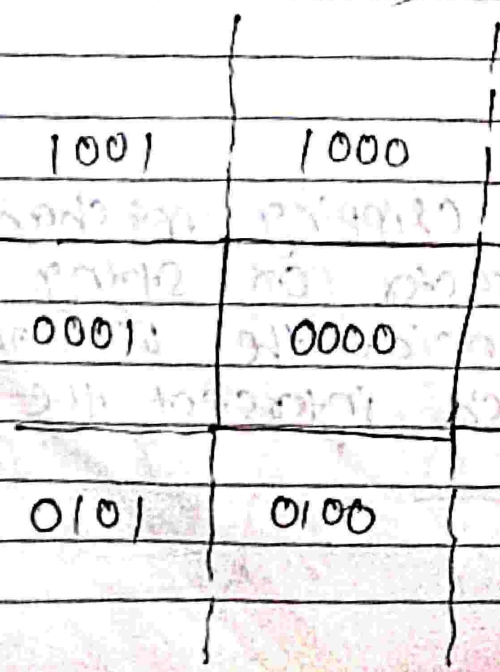
When 2 end co-ordinate points to be connected to each other i.e. called Line. The by default dimension in 1D.

- It has contain one property i.e. length
- The line clipping algorithm also used 3 parameter or method
  1. visible
  2. invisible
  3. clipped.
- In general way, the line to be clipped in 2 ways
  1. Coren, sutherland line clipping Algorithm
  2. Cyrus-Beck line clipping Algorithm

(1) Coren-Sutherland Line Clipping :-

Here the view port screen to be looks like rectangular or square shape

- In this case the scientist divided the window screen in 9 region and provides 9 obsec code value using the rule TBRL (Top Bottom Right Left)



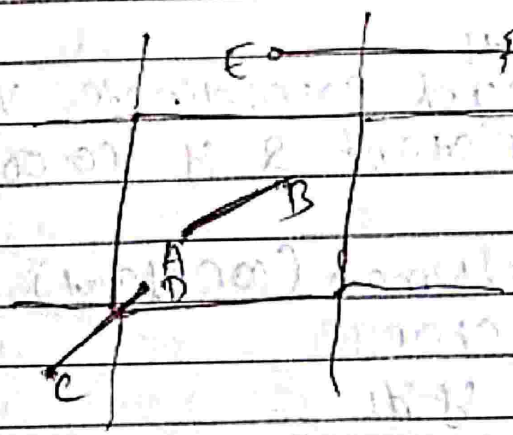
The line clipping to be possible using a code region value in 3 ways

① visible:
If both the end co-ordinate point to be placed over 0000 region

② invisible:
If both the end coordinate point to be present in same region space with 1 bit of the two end co-ordinate point to be present in different region space (then perform logical and operation, if result contain any 1 bit value then it outside of the window)

③ Clipped
If both the end co-ordinate points to be placed over different region, then it perform logical operation. if the resultant provides 0000 then it partially clipped or partially invisible otherwise fully discarded.

Ex



EF
E (1000)
F (1010)
1000
(fully outside invisible)

line CAB
region value of point A (0000)
" " " " B (0000) (visible)

line CD
C (0101)
D (0000)
0101
0000
0000
(partial clipped)

Calculation of Partial visited Point (Clipped Point)

- As though the windows shape looks like rectangular / square shape. it provides 4 partial visited method calculation like top bottom, left & right.

(i) **left ()** :-

Here the clipped point do be evaluated using slope of the line equation i.e,

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

- on this case, the x- co-ordinate value always constant & y- coordinate value to be change.

$x = x_{wmin}$ (constant)

y = change (calculation)

$$m = \frac{y - y_1}{x_{wmin} - x_1} \Rightarrow \boxed{y = m(x_{wmin} - x_1) + y_1}$$

(ii) **right ()** :-

Here the clipped point do be evaluated by using slope of eqn $m = \frac{y_2 - y_1}{x_2 - x_1}$

- on this case the x co-ordinate value always remains constant & y co-ordinate value to be change

$x = x_{wmax}$ (constant)

y = change

$$m = \frac{y - y_1}{x_{wmax} - x_1}$$

$$\Rightarrow \boxed{y = m(x_{wmax} - x_1) + y_1}$$

(iii) top():-

Here the clipped point to be evaluate by using slope of the equation $m = \frac{y_2 - y_1}{x_2 - x_1}$

- In this case the y-coordinate value always remains constant & x-coordinate value to be change.

$y = y_{max}$ (constant)

x = change

$$m = \frac{y_{max} - y_1}{x - x_1} \Rightarrow \boxed{x = \frac{1}{m}(y_{max} - y_1) + x_1}$$

(iv) bottom():-

Here the clipped point to be evaluate by using slope of the equation

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

- In this case, the y-coordinate value always remains constant & x-coordinate value change

$y = y_{min}$

x = change

$$m = \frac{y_{min} - y_1}{x - x_1} \Rightarrow \boxed{x = \frac{1}{m}(y_{min} - y_1) + x_1}$$

Ex

Here $(w_{xmin}, w_{ymin}) = (0,0)$

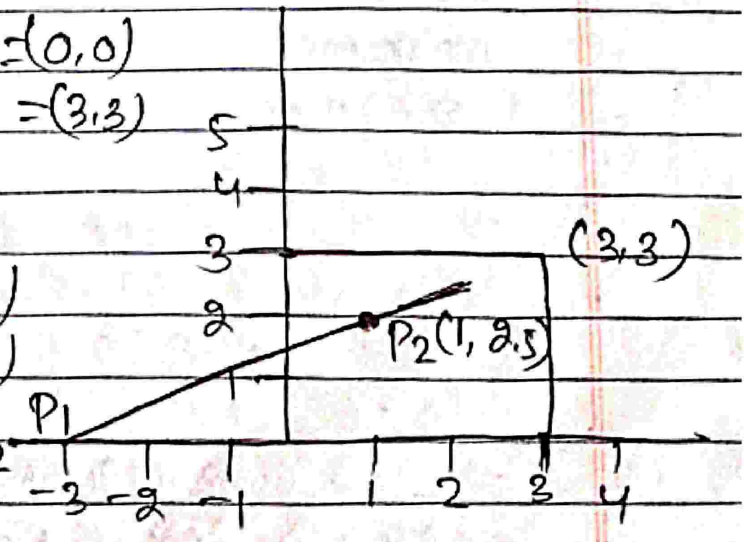
$(w_{xmax}, w_{ymax}) = (3,3)$

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

Here, $(x_1, y_1) = (-3, 0)$

$(x_2, y_2) = (1, 2.5)$

$$m = \frac{2.5 - 0}{1 - (-3)} = \frac{2.5}{4} = 0.62$$



$y = m(x_{wmin} - x_1) + y_1$

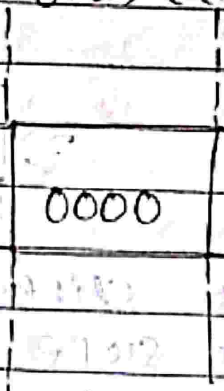
$= 0.62\,(0 - (-3)) + 0$

$= 0.62\,(3) = 1.86$

So the clipped point is $(0, 1.86)$

Algorithm :-

(I) To calculate a region object point using TOP, BOTTOM, Right, Left (TBRL) Rule.



(II) To initialize windows coordinate value i.e. (w_{xmin}, w_{ymin}) & (w_{xmax}, w_{ymax})

(III) To check object viewing portion using 3 rules i.e. • visible • invisible • clipped

(iv) To apply for calculating partial visited point using 4 function.

Cyrus beck Line clipping Algorithm :-

Cohen sutherland	Cyrus beck line
(i) This line clipping Algorithm to be only applies above rectangular or square shape view port screen	(i) But, this line clipping Algorithm to be applied any polygon shape view port screen
(ii) It is very simplest & and easiest application for line clipping	(ii) But it is very complex application for line clipping.
(iii) It is generally used in entertaining application.	(iii) But it is used Archi tech and engineering field.
(iv) It consumes very less time & less memory space for its execution	(iv) But it consumes more time & more memory space for execution

Mathematical Analysis :

Suppose a line to be draw by using 2 end co-ordinate pixel value i.e. P_0 & P_1 of given time t (1 sec). At the time of execution it interupt in intermediate pos^n (at time = 0.5) due to software & Hardware interrupt.

So now to calculate the intermediate co-ordinate point using parametric eq^n i.e.

$$\therefore P(t) = P_0 + t(P_1 - P_0) \quad \text{---(1)}$$

Now to check

at $(t=0) = P(0) = P_0 + 0(P_1 - P_0)$

$= P_0$

at $(t=1) = P(1) = P_0 + 1(P_1 - P_0)$

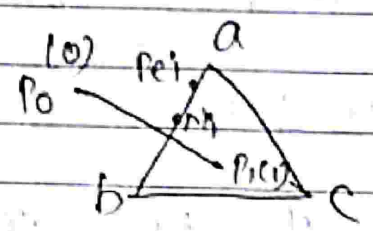
$= P_0 + P_1 - P_0$

$= P_1$

let us take this line above any polygon window & after that perform line clipping algorithm through using below rules

(I) If line > 0, then the given coordinate point to be present outside of the window.
(II) If line < 0, then the given co-ordinate point to be present inside of the window.
(III) If line = 0, then the given co-ordinate point lies above intersection point ...

As the intersection point is the clipping point, so the parametric eqn to be compared with line equal 0 value, it also used dot product method for evaluating clipping co-ordinate value.



Here the view port window to be looks like triangle polygon. one line here is above ab edge with end co-ordinate (Po Pi)
Now to perform clipping mechanism by assuming one normal vector Ni (edge) & another point Pei (point above edge)
where i = 1, 2, 3 ...
Now to apply Dot product for calculate intermediate pixel point.

$$Ni \cdot (P(t) - Pei) = 0$$
$$\Rightarrow Ni \cdot (Po + t(Pi - Po) - Pei) = 0$$
$$\Rightarrow Ni \cdot Po + Ni \cdot t \cdot Pi - Ni \cdot t \cdot Po - Ni \cdot Pei = 0$$

$\Rightarrow t(N_i \cdot P_1 - N \cdot P_0) = N_i(P_{ej} - P_0)$

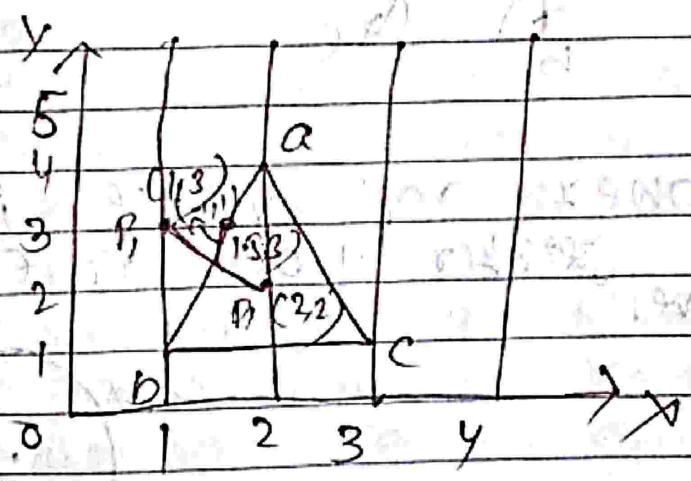
$\Rightarrow t = \frac{N_i \cdot (P_{ej} - P_0)}{N_i \cdot (P_1 - P_0)}$

Let $D = N_i(P_1 - P_0)$

As the Dot product to be possible using multivalue so it enter in bet^n windows using the below constraint.

If $D > 0$ then the single point to be leaving (if multiple, then it choose smallest value among them & applies parametric eq^n.

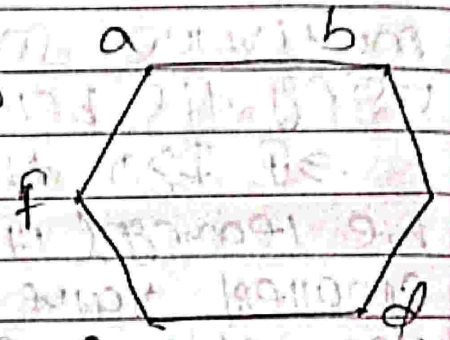
If $D < 0$, then the single point to be entering (if multiple, then it choose largest value among them & applies parametric eq^n).



Polygon

It is one graphics o/p primitive elements, whoose shape looks like flat Rigid.

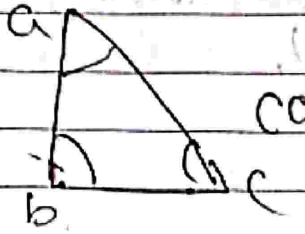
- This object to be build-up using at least 3 edges



- In general way, the polygon shapes to identify in 2 way

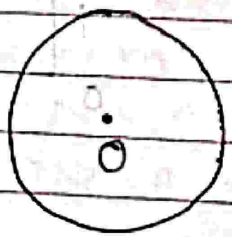
① Convex:

If all object vertices interior angle value is less than 180° i.e. convex



② Concave:

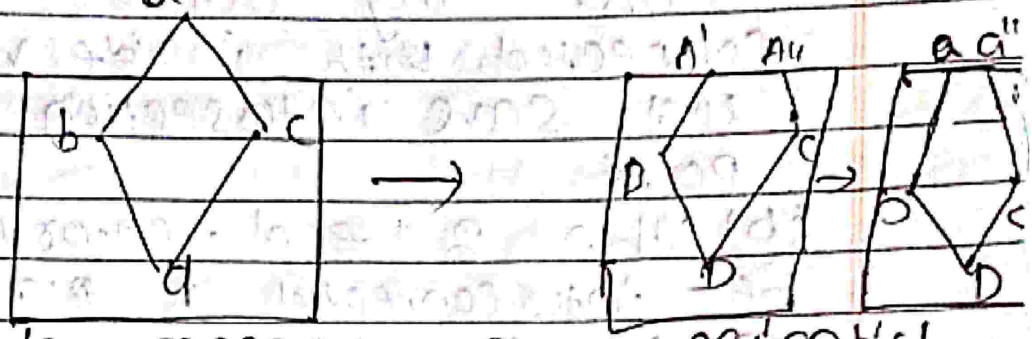
If obect any vertices interior angle value is greater than or equal to 180° i.e. concave.



Polygon clipping

The polygon object to be viewing in every new port shape screen But in general way, the screen to be looks like rectangular or square shape then any object lies on rectangular or square

view port then it apply clipping Algo. using 3 parameter (visible, invisible, clipped) After performing clipping mechanism it provides the object view screen looks like open polygon shape or other shape.



After this procedure some scientist create multiple polygon object through close look using clipping 3 parameter

Sutherland Hodgeman polygon clipping Algorithm

This polygon mechanism provides optimization Accurate polygon solution above view port screen as the view port screen size to be desired

→ It solve by using different shape view port screen but in general way it used rectangular or square shaped viewport screen

Algorithm

1. It initialize one list parameter with empty (null) values.
2. The list structure to be looks like linear array

(3) The list store 2 types of data valu
i.e. inside object vertices & intersect
vertices point

(4) At applying 4 rules for save all the
vertices in to list

(a) When any outside vertex to be
connected with inside vertex, then the
list save intersection point & inside
point

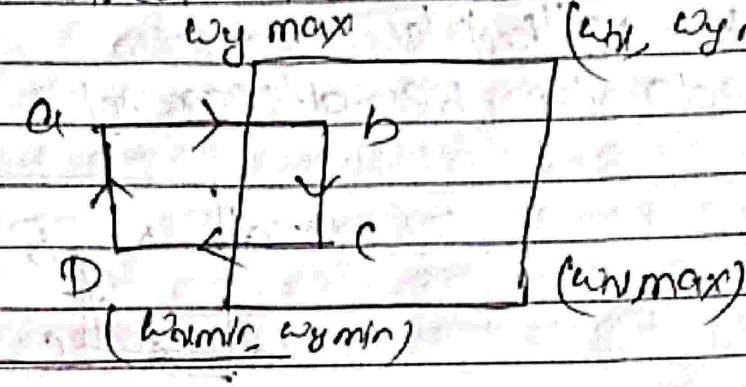
(b) When 2 end-co-ordinate point to
be interconnected & placed over inside
region then the last end point to be
save into list

(c) When any inside vertex to be inte
connected with outside vertex then the
save intersection point.

(d) When 2 end co-ordinate point
to be interconnected & placed over
outside the Region then the given
object portion to be discarded.

(5) Now to find all intersection & point.
using Cohen Sutherland line clipping Algo

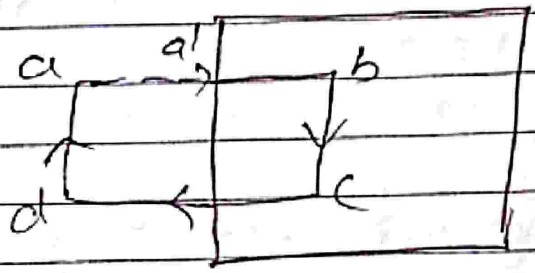
Ex To perform polygon clipping using Sutherland.



① list = [ ]

② outside to inside (a → b)

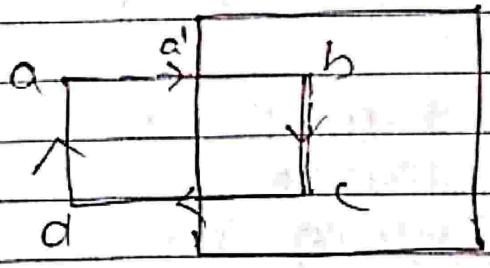
list = [a' | b]



③ (b → c)

inside → inside

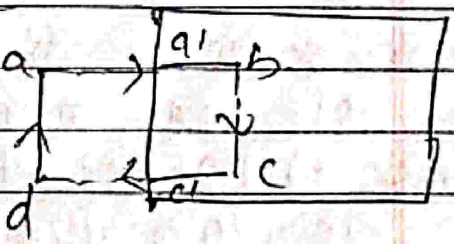
list [a' | b | c]



④ (c → a)

inside → outside

list = [a' | b | c | c']

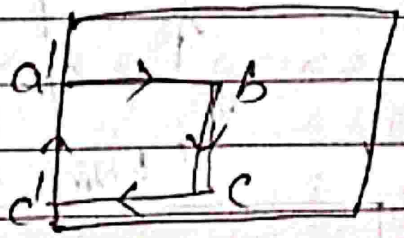


(5) d → a

Outside → outside

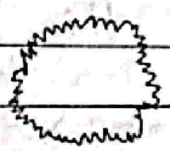
List = [a'|b|c|c']

Hex da to be discarded.



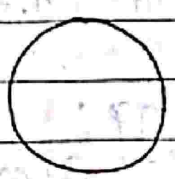
Aliasing :-

"Aliasing is the apperance of jagged distortions in curves and diagonal lines in computer Graphics because the resolution is limited or diminished".



Anti-aliasing :-

Anti-aliasing is a method for improving the realism of an image by removing the jagged edges from it.



Filling Algorithm :- / Polygon filling :-

Polygon :-

A flat shape with at least 3 and usually 5 or more.

→ A polygon is a plonal geomatric figure that has a finite number of sides. The sides of a polygon are made of straight line segments connected to each other end to end.

Types :-

eg: 

Convex polygon :-

If all the interior angles of a polygon are strictly less than 180 degree. The vertex will point outwards from the centre of the shape.

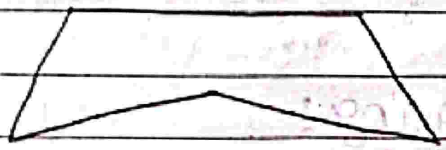
Concave Polygon:-

If one or more interior angles of a polygon are more than 180 degree, then it known as a concave polygon.

⇒ A concave polygon can have at least 4 sides

⇒ The vertex points towards the inside of the polygon.

eg.



Polygon Filling:-

Filling the polygon means highlighting all the pixels which lie inside the polygon with any colour other than background colour.

Types of Polygon Filling:

There are 2 basic approaches used to fill the polygon.

1. SEED FILL Algorithm
2. Boundary fill Algorithm

① SEED FILL Algorithm:-

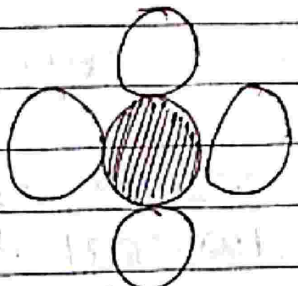
Here one way to fill a polygon is to start from given "seed" point known to be inside the polygon and highlight outward from this point, i.e. neighbouring pixels until we encounter the boundary pixels this approach is called C because colour flows from the seeds pixel until reaching the polygon boundary, like water flooding on the surface of the container

→ The seed fill algorithm is further classified as two types:-

(I) Flood fill algorithm
(II) Boundary fill algorithm or edge fill algorithm.

(II) Flood fill Algorithm :-
On this method, edges of the polygons are drawn. then starting with some seed, any point inside the polygon we examine the neighbouring pixels to check wheather the boundary pixel is reached. If boundary pixels are not reached, pixels are highlighted & the process is continued until boundary pixels are reached.
- Boundary defined regions may be either 4-connected or 8-connected.

4-connected :-
If a regeion is 4-connected, then every pixel in the regeion may be reached by a combination of moves in only 4 directions
(I) left (II) Right (III) up (IV) Down



(Fig. four connected regeion)

Algorithm :-
Procedure : boundary-fill (x, y, f-colour, b-colour)
{
if (getpixel(x,y) != b-colour && getpixel(
!= f-colour)
{

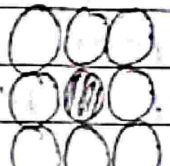
putpixel (x, y, f_colour)
boundary_fill (x+1, y, f_colour, b_colour);
boundary_fill (x, y+1, f_colour, b_colour);
boundary_fill (x-1, y, f_colour, b_colour);
boundary_fill (x, y-1, f_colour, b_colour);
}}

→ Here, getpixel fun' gives the colour of specific pixel and 'putpixel' function draws the pixel with specified colour.

8 connected region :-

⇒ For 8-connected region every pixel in the region may be reached by a comb^n of moves in the two horizontal, two vertical & 4 diagonal dir^n

→ In some cases, an 8-connected algorithm is more accurate than the 4-connected algorithm



(Fig. 8 connected region)

Algorithm :-

→ Same procedure as 4-connected, or add the test for diagonal test (4 additional) i.e. (x+1, y+1)

boundary_fill (x+1, y+1, f_colour, b_colour);
boundary_fill (x-1, y-1, f_colour, b_colour);
boundary_fill (x+1, y-1, f_colour, b_colour);
boundary_fill (x-1, y+1, f_colour, b_colour);
}

(1) Flood Filling Algorithm :-

Sometimes, it is required to fill in an area that is not defined within a single colour boundary. In such cases we can fill areas by replacing a specified interior colour instead of searching for a boundary colour. This approach is called flood fill algorithm.

→ Like boundary fill algorithm, here we start with some seed and examine the neighbouring pixels. However, here pixels are checked for a specified interior colour instead of boundary colour and they are replaced by new colour.

→ It also used 4-connected or 8-connected approach

Algorithm :- (8-connected)

```
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);
  Flood Fill (x-1, y, old_colour, new_colour);
  Flood Fill ( x, y+1, old_colour, new_colour);
  Flood Fill ( x, y-1, old_colour, new_colour);
  Flood Fill (x+1, y+1, old_colour, new_colour);
  Flood Fill (x-1, y-1, old_colour, new_colour);
  Flood Fill (x+1, y-1, old_colour, new_colour);
  Flood Fill (x-1, y+1, old_colour, new_colour);
  }}
```

scan line polygon fill Algorithm:

→ Recursive algorithm for seed fill methods have got 2 difficulties. the first difficulty is that if some inside pixels are already displayed in fill colour then recursive branch terminates, leaving further interior pixels unfilled. to avoid this difficulty, we have to 1st change the colour of any interior pixels that are initially set to the fill colour before applying the seed fill procedures.

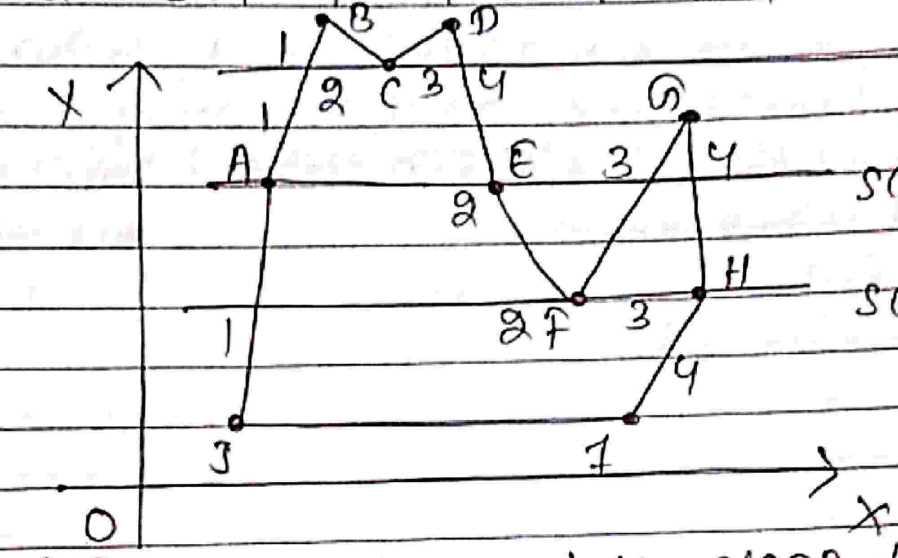
→ Another difficulty with recursive seeds fill methods is that it can't be used for large polygons. This is because recursive seed fill procedures requires stacking of neighbouring points.

→ To avoid this problem more efficient methods can be used, such method fills horizontal pixel spans across scan lines, instead of proceeding to 4 & 8 connected neighbouring points.

→ This is achieved by, identifying the rightmost and leftmost pixels of the seed pixel and then drawing a horizontal line between these two boundary pixels. this procedure is repeated with changing the seed pixel above & below the line just drawn until complete polygon is filled with this efficient method we have to stack only a beginning position for each horizontal pixel span, instead of stacking all unprocessed neighbouring positions around the current posn.

eg:- this below figure illustrates the scan line algorithm for filling of polygon. For each scan line crossing a polygon this algorithm

locates the intersection points of the scan line with the polygon edges. These intersection points are then sorted from left to right, and the corresponding positions between each intersection pair are set to the specified fill colour



(Fig. Insection points along the scan line that intersect polygon vertices).