INTRODUCTION

- Cartography → mapping (Google map)

- Moore's law:- The number of transistors in a dense Integrated circuit doubles approximately in every two years.
  Named after 'Gordon E. Moore' = co-founder of Intel.

- Super-resolution Imaging:- Technique that enhance the resolution of imaging system.
  * Optical SR — the diffraction limit of system is transcended
  * Geometrical SR — the resolution of digital imaging sensors is enhanced.

  * Picture Speaks 1000 of words.

- Digital Zoom:- accomplished by cropping an image down to a centered area with the same aspect ratio as the original.
  Interpolating the result back up to the pixel dimensions of the original.

- OpenGL = Open Graphics Library:- A cross-language, cross-platform application programming interface (API) for rendering 2D and 3D vector graphics.

Hardware Concepts:-

• Display Drivers:- Extend the function of OS's kernel → Priviliged space
space version

User Mode
Kernel Mode

write
System Call

function call → library call (eg printf)

• SAGE (Semi Automatic Ground Environment)

* A system of large computers and associated networking equipment that coordinated data from many radar sites and processes it to produce a single unified image of the airspace over a wide area. e.g: Whirlwind I

• CAD (Computer-Aided Drafting)

* Use of computer systems to aid in the creation, modification, analysis or optimization of a design.

• CAM (Computer-Aided Manufacturing)

* Use of software to control machine tools and relate in the manufacturing of work-pieces.

Topics

a) Special effect (SFX or SPFX or simply FX) — → optical effects → mechanical effects

e.g: Matrix game
trinity jump [human eye = 24 frames/sec]

b) Second Life:- It is an online virtual world, developed by Linden lab. (3D-cyber space)

- Holography:- Science of making holograms.
  * Holograms → a photographic recording of a light-field, rather than of an image formed by a lens
  → used to display 3D image of holographed subject.

- Scanner as I/p and Printer as O/p.
- Pictorial-drawing = detailed viewing of an object.

- Projector as embedded-system.
  Computer ⟶ RAM ⟶ Buffer

- bmp file = bitmap image file:- raster graphics image file format used to store bitmap digital images

- LCD (Liquid Crystal Display):-
  → Active Matrix Display = Thin Film Transistor (TFT)
  - uses a separate transistor to apply charges to each liquid crystal cell and thus displays high-quality color.
  ~~e.g: Glass e.g: gas plasma display.~~

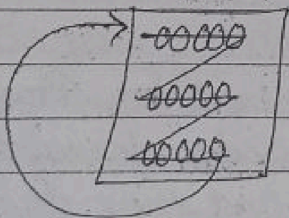
→ Passive Matrix Display:- Fewer transistors.

- Gas Plasma Display :- (Flat-Panel Display)
  - *Collection of very small neon bulbs.
  - *produce very sharp monochrome images
  - *require much more power.

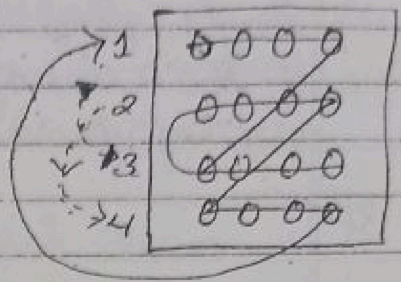
- Raster Display Technology:- a stroke, a line drawing.

- Vector Display Technology:-
  - for vector display;
    - user-mode application (.exe file)
      - ↳ Graphic Software Package
        - ↳ move to (point, point)
        - line to (point, point)
          - display processor
            - → steady image
            - (1 sec at 60 time)
              - ↳ 60Hz as refresh rate

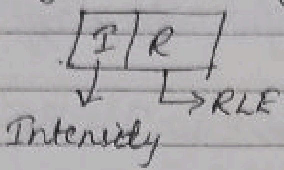
- RDT :-
  - *Vertical-Horizontal Retrace:- Scan-line
    - one to one
    - down to up



* Inter-laced Monitors :- even once then odd.



* RLE :- Run-Length Encoding



- RLE + AI-based Animation

- Roles of Vector Display & Frame Buffer:-
  * VD = access bit value of memory location.
    1 sec = 60 times
    = processor i.e: CPU.

  * FB = works in absence of VD
    e.g: projector.

- max. no. of intensities achievable out of a single device = 2^n
  where n = no. of bits used to represent a pixel.

- Graphic cards = 24 bit video card = 24 bits
  = 16.7 million colors.

Q: If the max^{m} no. of intensities achievable out of a single pixel on the screen is 256 & the total resolution of the screen is 640 * 480. What will be the required size of frame buffer for display purpose?

Sol^{n}

max^{m} no. of intensities $= 2^n = \cancel{256} = 2^8$

$\Rightarrow \cancel{2^{256} = 2}$

$\Rightarrow 256 = 2^n$

$\Rightarrow 2^8 = 2^n \quad \therefore n = 8$

$\therefore$ Required size frame buffer $= 8 * 640 * 480$

$= 2457600$ bits

Q: How long would it take to load a 640 * 480 frame buffer if 10^5 bits can be transferred per second?

Sol^{n}

$$\Rightarrow \frac{640 * 480}{10^5} = 3.072$$

Q: In case of a raster system with resolution 640*480. How many pixels could be accessed per second by a display controller that refreshes the screen at a rate of 60 frames per second?

Sol^{n}

frequency $= \frac{1}{T} = 60$ frames/sec.

$\therefore T = 0.0166$

Then, $T = 640 * 480 * x$ sec

$$\Rightarrow \frac{0.0166}{640 * 480} = x \quad \therefore x = 5.42 * 10^{-8} \text{ sec}$$

$= 0.542 * 10^{-9}$ ns.

Q: If pixels are accessed from a frame buffer with an average access time of 20 ns to glow a single pixel & the total resol^n of the screen is 640*480, will there a flickering effect seen on the screen?

sol^n

To glow all pixels $= 640 * 480 * 20 * 10^{-9}$
$= 6.14 * 10^{-3}$

Then,

$$f = \frac{1}{T} = \frac{1}{6.14 * 10^{-3}} = 162.760 \text{ frames/sec.}$$

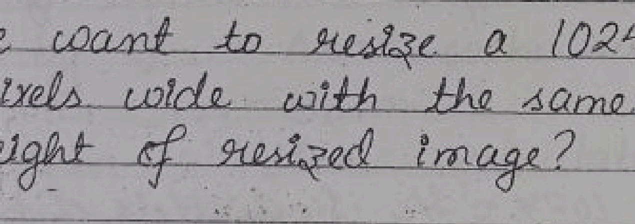
Aspect Ratio:- Ratio of width to its height measured in unit length or no. of pixels.

if height = 2 inches
aspect ratio = 1.5
then width = 3

Q: If we want to resize a 1024 * 768 (height) image to one i.e. 640 pixels wide with the same aspect ratio, what will be height of resized image? $\frac{1024}{768}$ 1/s

Q: If we use direct loading of RGB values with 2 bits per primary color. How many possible colors do we have for each pixel?

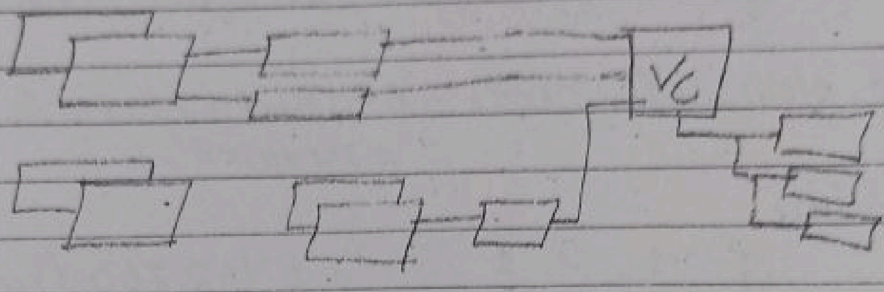
A.R = w/h



• bit plane = page
• frame buffer = copy? (collecⁿ of bit plane)

Date ______
Page ______

Q: If we use 5 bits each for red & blue and 6 bits for green for a total of 16 bits per pixel, how many possible simulations do we have?



Q: Compute the size (in inches) of a 640 * 480 image at 240 pixels per inch.

Solⁿ 640*480 image
240 pixels per inch (1" = 240)

Q: Compute the resolⁿ of a 2*2 inch image that has 512*512 pixels.

? Q: Aspect ratio $= \frac{W}{H} = \frac{1024}{640} = 3/4$. find pixels of height.

$\Rightarrow A = \frac{3}{4}$

640 pixels width; $\frac{3}{4} = \frac{640}{H}$ $\therefore H = \frac{640}{3/4} = 480$ pixel.

Chapter:- Two Dimensional Algorithm

DIGITAL DIFFERENTIAL ANALYZER:-

(i) Slope intercept eq^n of a st. line in $y = mx + c$.

(ii) For points (x_1, y_1) (x_2, y_2);

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

(iii) Line drawing algorithm depend on the equations, for given Δx we can compute 'y' interval $\Delta y = \Delta x . m$.

(iv) These equations from the basis for determining deflection voltages in analog devices.

(v) DDA is a scan conversion algorithm based on the calculating Δx & Δy.

(vi) We sample at unit interval in one co-ordinate location & determine the corresponding integer value nearest the line path for other coordinate.

~~Line with +ve slope (moving from L-R)~~

* Line with +ve slope (moving from L-R)

(1) If slope is less than equal to 1 we sample at unit interval in 'x' direction (increase 'x' co-ordinate value by 1) i.e: $\Delta x = 1$, each successive 'y' value is computed as $y_{k+1} = y_k + m$

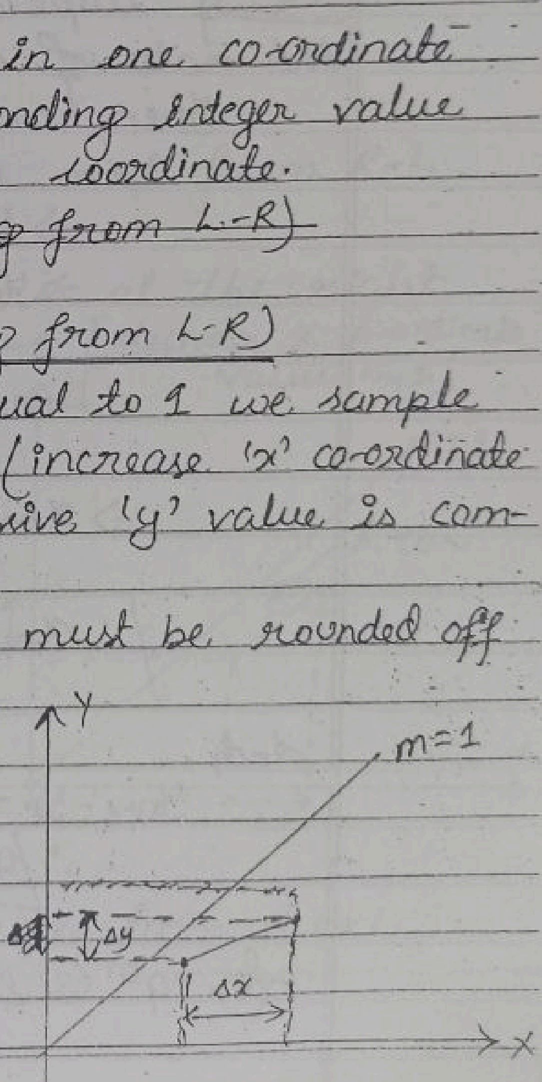
then 'y' value calculated must be rounded off to nearest integer.

when $m < 1$,

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

$$= \frac{y_{k+1} - y_k}{x_{k+1} - x_k}$$

$$= \frac{y_{k+1} - y_k}{1}.$$



$\Rightarrow m = y_{k+1} - y_k$ $\boxed{\therefore y_{k+1} = y_k + m}$ ... ①

$\Delta x = 1$

i.e. $x_{k+1} - x_k = 1$ ... ②

so, $x_{k+1} = x_k + 1$ ... ③

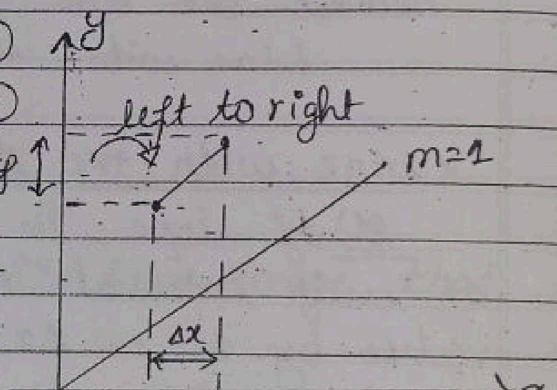
$\therefore$ ① & ③ is the required when moving from L-R & slope 'm' < 1.

(b) If slope is greater than 1 ($m > 1$), reverse the roles of x and y.

Then, $m = \frac{\Delta y}{\Delta x}$; unit y intervals ($\Delta y = 1$);

$\Rightarrow y_{k-1} - y_k = 1$ ... ①

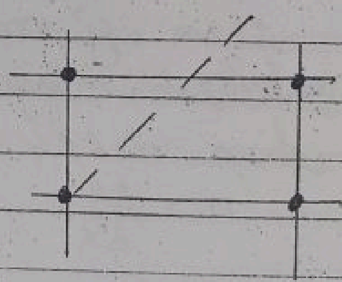
$\Rightarrow y_{k-1} = y_k + 1$ ... ②



$\therefore m = \frac{1}{x_{k+1} - x_k}$

$\Rightarrow x_{k+1} - x_k = \frac{1}{m}$

$\boxed{\therefore x_{k+1} = x_k + \frac{1}{m}}$ ... ③



And,

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

$\therefore \boxed{y_{k+1} = y_k + 1}$ ... ④

So, eq^n ③ & ④ is required when moving L-R when $m > 1$.

+ve slope (moving from R-L):-

(c) When the absolute value of a -ve slope is greater than 1, we use $\Delta y = -1$ [decrease 'y' coordinate value by 1]

Then,

$$y_{k+1} - y_k = -1 \quad (\because \Delta y = -1)$$

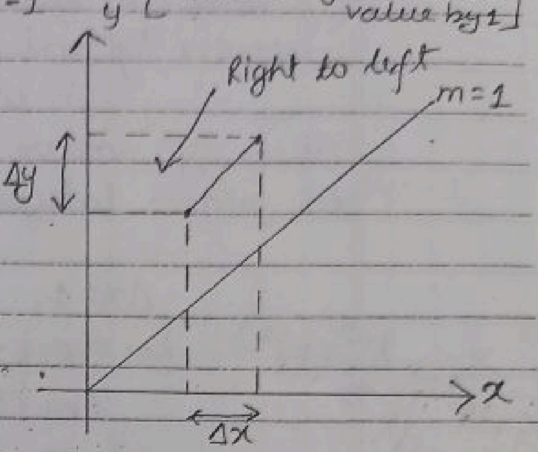
$$\Rightarrow y_{k+1} = y_k - 1 \cdots ①$$

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

$$\Rightarrow m = \frac{-1}{x_{k+1} - x_k}$$

$$\Rightarrow x_{k+1} - x_k = -\frac{1}{m}$$

$$\boxed{\therefore x_{k+1} = x_k - \frac{1}{m}} \cdots ②$$



eq^n ① & ② is required when $m > 1$ from R-L.

(d) When the start endpoint is at the right (for the same slope), we set $\Delta x = -1$ [decrease x co-ordinate value by 1]

Then,

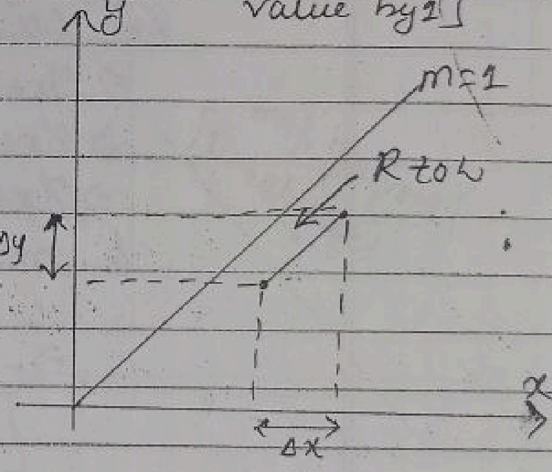
$$x_{k+1} - x_k = -1$$

$$\Rightarrow x_{k+1} = x_k - 1 \cdots ①$$

& $m = \frac{\Delta y}{\Delta x} = \frac{\Delta y}{-1}$

$$\Rightarrow -m = y_{k+1} - y_k$$

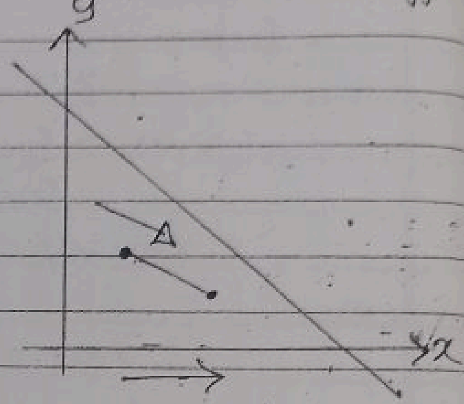
$$\boxed{\therefore y_{k+1} = y_k - m} \cdots ②$$



Then required eq^n ① & ② moves from R to L when $m < 1$.

* Line with -ve slope:-

* If $|m| \leq 1$ we sample at unit interval in 'x' directio (if ~~~~ increase 'x' coordinate value by 1) i.e: $\Delta x = 1$, each successive 'y' value is computed as ~~fmt~~ $y_{k+1} = y_k + m$.
where $|m|$ is the magnitude of slope.
Then 'y' value calculated must be rounded off to nearer Integer.



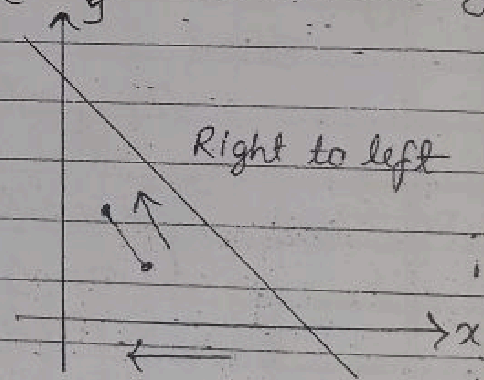
[Moving from left to Right]

$$\Delta x = 1,$$
$$\text{i.e: } x_{k+1} - x_k = 1$$
$$\therefore x_{k+1} = x_k + 1 \cdots (1)$$
$$\& \ m = \frac{\Delta y}{\Delta x} = \frac{\Delta y}{1}$$
$$\Rightarrow m = y_{k+1} - y_k$$
$$\boxed{\therefore y_{k+1} = y_k + m}$$

* If $|m| \leq 1$ (magnitude of slope) (decrease 'x' co-ordinate value by 1)

[Moving from Right to Left]

$$\text{i.e: } \Delta x = -1$$
$$\Rightarrow y_{k+1} = y_k - m \ \& \cdots (1)$$
$$\Rightarrow x_{k+1} - x_k = -1$$
$$\Rightarrow x_{k+1} = x_k - 1 \cdots (2)$$
$$\& \ m = \frac{\Delta y}{\Delta x} = \frac{\Delta y}{(-1)}$$
$$\Rightarrow -m = y_{k+1} - y_k$$
$$\boxed{\therefore y_{k+1} = y_k - m}$$



'y' value calculated must be round off to nearest Integer.

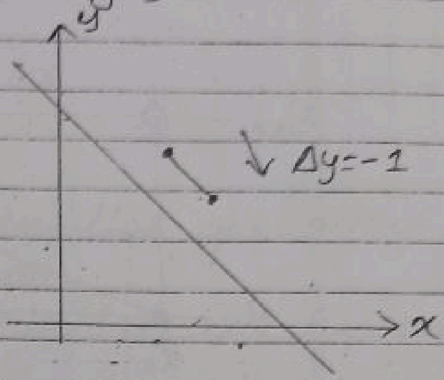
Line with -ve slope:-

* If $|m| > 1$ [moves Left to Right]

$\Delta y = -1$ (y decrease procedure by 1)

Then,

$\Rightarrow x_{k+1} = x_k - \frac{1}{m} \quad \ldots (1)$



& $\Rightarrow y_{k+1} - y_k = -1 \quad \ldots (2)$

$\Delta y = -1$

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

Then,

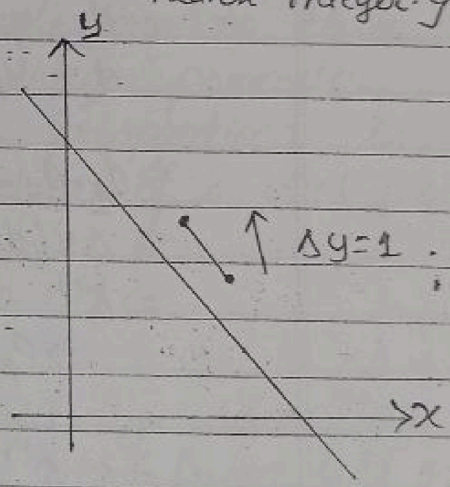
$m = \frac{\Delta y}{\Delta x} = \frac{-1}{\Delta x}$

$\Rightarrow \Delta x = -\frac{1}{m}$

so, $x_{k+1} - x_k = -\frac{1}{m}$

$\boxed{\therefore x_{k+1} = x_k - \frac{1}{m}}$ {'x' value calculated must be round off to nearest integer.}

* If $|m| > 1$ [moves left to right]



$\Delta y = 1$

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

$\boxed{\therefore y_{k+1} = y_k + 1} \quad \ldots (1)$

& $m = \frac{\Delta y}{\Delta x} = \frac{1}{\Delta x}$

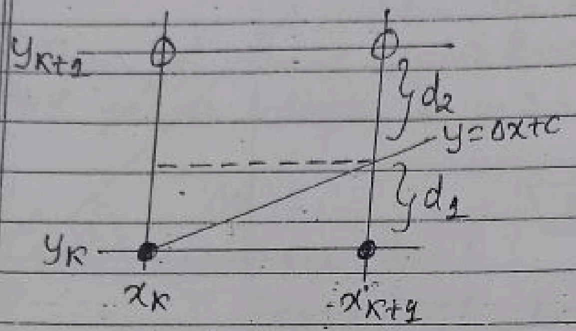
$\Rightarrow \Delta x = \frac{1}{m}$

$\Rightarrow x_{k+1} - x_k = \frac{1}{m}$

$\boxed{\Rightarrow x_{k+1} = x_k + 1/m} \quad \ldots (2)$

Imp BRESENHAM'S LINE DRAWING ALGORITHM :

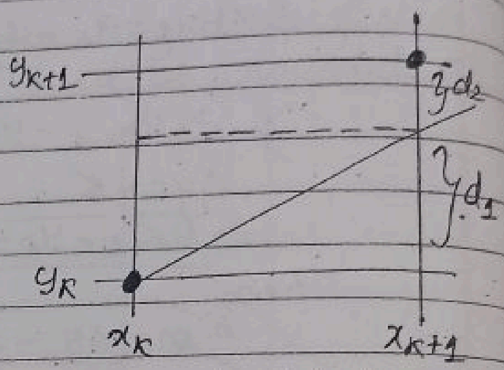
FOR $|m| \leq 1$; $m = \frac{\Delta y}{\Delta x}$



$d_1 - d_2 < 0$

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

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



Imp

$d_1 - d_2 > 0$

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

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

EQUATION OF LINE $= m(x_k + 1) + c$

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

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

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

$\Rightarrow \Delta x(d_1 - d_2) = 2\Delta y(x_k + 1) + 2c\Delta x - 2y_k\Delta x - \Delta x$

$\Rightarrow k^{th}$ step $(P_k) = 2\Delta y\, x_k + 2\Delta y + 2c\Delta x - 2y_k\Delta x - \Delta x$

$\Rightarrow P_k = 2\Delta y\, x_k - 2\Delta x \cdot y_k + b$

where, $b = 2\Delta y + 2c\Delta x - \Delta x$

$\Rightarrow k+1^{th}$ step $(P_{k+1}) = 2\Delta y \cdot x_{k+1} - 2\Delta x\, y_{k+1} + b$

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

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

Case ①, if $P_k < 0$, $y_{k+1} = y_k$

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

Case ⑪, if $P_k \geq 0$, $y_{k+1} = y_k + 1$

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

Imp

* STEPS:-

\# Accurate and easier raster line generating algorithm developed by Bresenham.

\# Vertical axis represents horizontal axis represents pixel columns.

\# For line with slope < 1 sample at unit 'x' interval in x-direction, determine which of the 2 possible pixel positions is closer to the line path at each sample step.

\# If we have plotted pixel at location (1,1) next pixel to plot would be either (2,1) or (2,2).

\# Bresenham's algorithm gives solution to this problem by testing the sign of integer parameter (decision parameter) whose value is proportional to the difference between the separation of 2 pixel positions from actual line path.

* For line with slope ($|m| \leq 1$)

\# Pixel positions are determined by sampling at unit 'x' interval starting at (x_0, y_0) from left end.

\# Idea is to find the value of 'y' closest to the line path.

\# For any k^{th} step - if pixel at $(x_k\ y_k)$ has been displayed then next pixel to plot in column x_{k+1}

is either (x_{k+1}, y_k) or (x_{k+1}, y_{k+1})
At x_{k+1} label vertical pixel separations from line path as d_1, d_2 now y coordinate at x_{k+1} will be $y = m(x_k+1)+c$

So, the distance between the ideal location & the lower pixel is $d_1 = y - y_k$ or $m(x_k+1)+b$
and the distance between the upper pixel & the ideal location is $d_2 = y_{k+1} - y$ or

$$d_2 = y_{k+1} - m(x_{k+1}) - b$$

Thus the difference $(d_1 - d_2) = 2m(x_{k+1}) - 2y_k + 2b - 1$

Substituting $m = \frac{\Delta y}{\Delta x}$ & denoting,

$\Delta x(d_1 - d_2)$ by P_k (the decision parameter)

$$P_k = \Delta x(d_1 - d_2) = 2\Delta y x_k - 2\Delta x . y_k + C \cdots (1)$$

where, $C = 2\Delta y + \Delta x . 2b - \Delta x$

Now, at $k+1^{th}$ step, decision parameter will be

$$P_{k+1} = 2\Delta y . x_{k+1} - 2\Delta x . y_{k+1} + C \cdots (11)$$

Now, subtracting (1) & (11),

$$P_{k+1} = P_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k) \ldots (111)$$

where, $x_{k+1} - x_k = 1$ or $\Delta x = 1$ or $|m| < 1$

Case (1)

if $P_k < 0$ then,

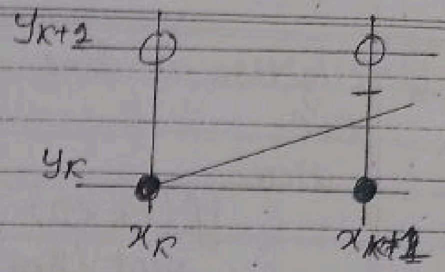
pixel at y_k is closer to line path

Then, $d_1 < d_2$ i.e. $P_k = -ve$ so the next pixel to plot in this case will be (x_{k+1}, y_k)

i.e. $y_{k+1} = y_k$ &

eq^n (iii) becomes,

$$P_{k+1} = P_k + 2\Delta y \cdots (a)$$



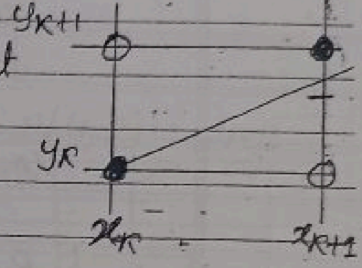
Case (2)

if $P_k > 0$ then pixel at y_{k+1} is closer to line path and $d_2 < d_1$

so, $P_k = +ve$ & next pixel to plot is (x_{k+1}, y_{k+1}) i.e. $y_{k+1} - y_k = 1$

so, eq^n (iii) becomes,

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



This recursive calculation of decision parameter is performed at each 'x' position starting from Left to Right end point.

Q: The initial decision parameter $P_0 = ?$

$\Rightarrow$ We have,

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

$= 2m(x_0 + 1) - 2y_0 + 2c - 1$ if $(x_0\ y_0)$ is the starting point.

$$d_1 - d_2 = 2mx_0 + 2m - 2y_0 + 2c - 1$$

$$= 2(mx_0 - y_0 + c) + 2m - 1 \quad [\because y - mx - c = 0$$

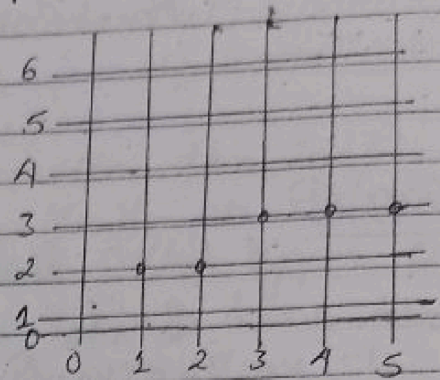
$$\Delta x(d_1 - d_2) = 2\Delta y - \Delta x$$

$$\boxed{\therefore P_0 = 2\Delta y - \Delta x}$$

Q: Digitise a line with end points A(1,2) B(5,3) using BLA.

Here,

if ($P_k < 0$)
  plot (x_{k+1}, y_k)
  $P_{k+1} = P_k + 2\Delta y$
else
  plot (x_{k+1}, y_{k+1})
  $P_{k+1} = P_k + 2\Delta y - 2\Delta x$



so, $P_0 = 2\Delta y - \Delta x = 2-4=2$ $\because \Delta y = 1$ $\Delta x = 4$

K	P_k	x_{k+1}	Y_{k+1}
0	$P_0 = -2$	2	2
1	$P_1 = -2+2=0$	3	3
2	$P_2 = 0+2-8=-6$	4	3
3	$P_3 = -6+2=-4$	5	3

DDA

Q: Digitise a line with end points A(3,2) B(7,5) using DDA ~~m~~

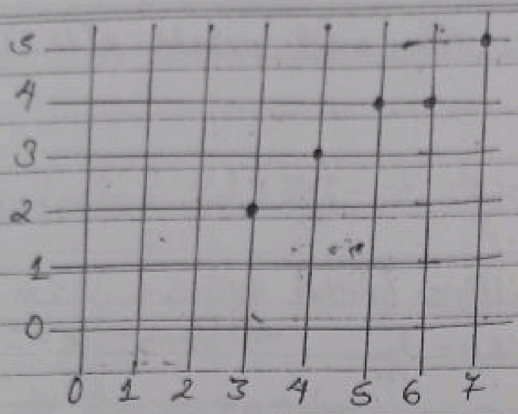
Here,

$m = 3/4 = 0.75$

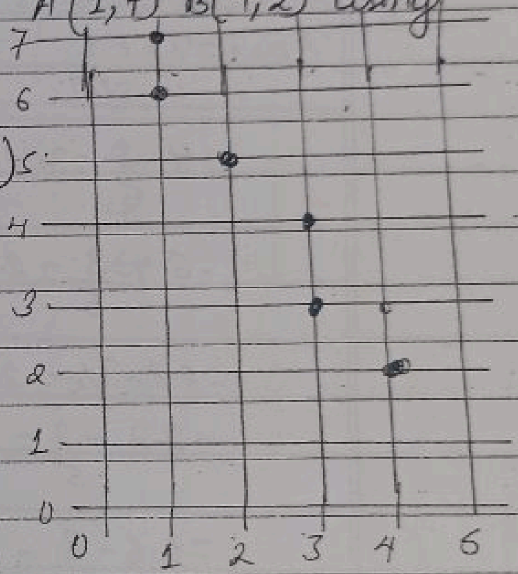
x_{k+1}	y_{k+1}	y_{k+1} (rounded value)
4	$2+0.75=2.75$	3
5	$2.75+0.75=3.5$	4
6	$3.5+0.75=4.25$	4
7	$4.25+0.75=5$	5

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

$y_{k+1} = y_k + m$



Q: Digitize a line with end points A(1,7) B(4,2) using DDA.



Here,

$m = \frac{-5}{3} = -1.66 > 1$ (negative; L to R)

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

Then,

$\Delta x_{k+1} = \Delta x - \frac{1}{m}$

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

$x_{k+1} = (x_k - 1/m)$	$y_{k+1} = y_k - 1$	rounded value (y_{k+1})
$1 + \frac{3}{5} = 1.6$	6	1
$1.6 + 3/5 = 2.2$	5	2
$2.2 + 3/5 = 2.3$	4	3
$2.3 + 3/5 = 3.4$	3	3
$3.4 + 3/5 = 4$	2	4

Date
Page

Q: Digitize a line with end points A(4,2) B(1,7) using DDA.

→

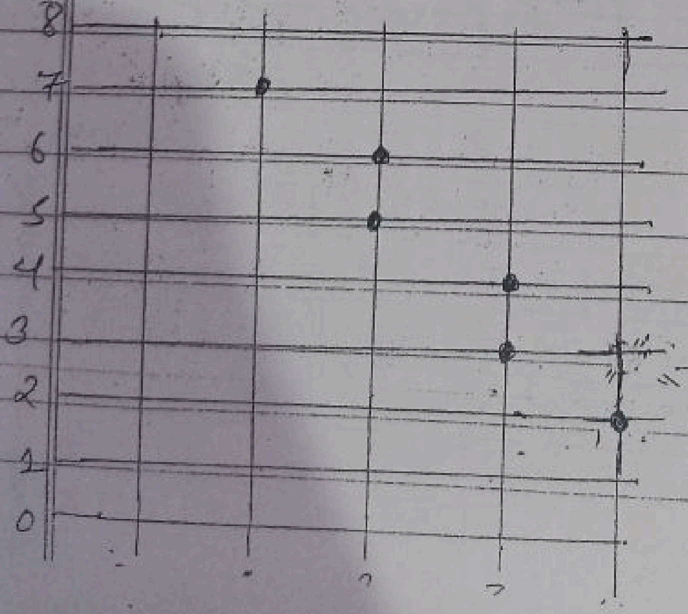
Slope (m) = $\frac{7-2}{1-4} = \frac{5}{3} = -1.67 1$

Slope > 1, negative (R-L)

$\Rightarrow x_{k+1} = x_k + \frac{1}{m}$ $\Rightarrow y_{k+1} - y_k = 1$

~~$y_{k+1} + 1 - y_k$~~

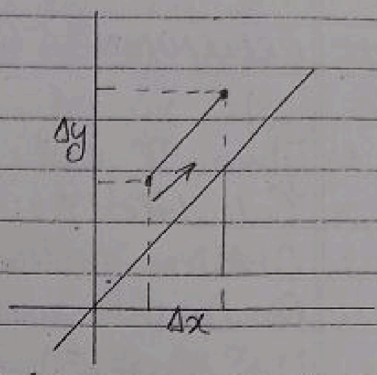
$x_{k+1} = x_k + \frac{1}{m}$	$y_{k+1} = y_k + 1$	Rounded value (x_{k+1})
$= 4 - \frac{3}{5} = 3.4$	3	3
$= 3.4 - \frac{3}{5} = 2.8$	4	3
$= 2.8 - 3/5 = 2.2$	5	2
$= 2.2 - \frac{3}{5} = 1.6$	6	2
$= 1.6 - \frac{3}{5} = 1$	7	1



Q. Digitize a line with end points A(2,4) B(5,8) using DDA

Soln

Slope $(m) = \frac{8-4}{5-2} = \frac{4}{3} = 1.33 > 1$

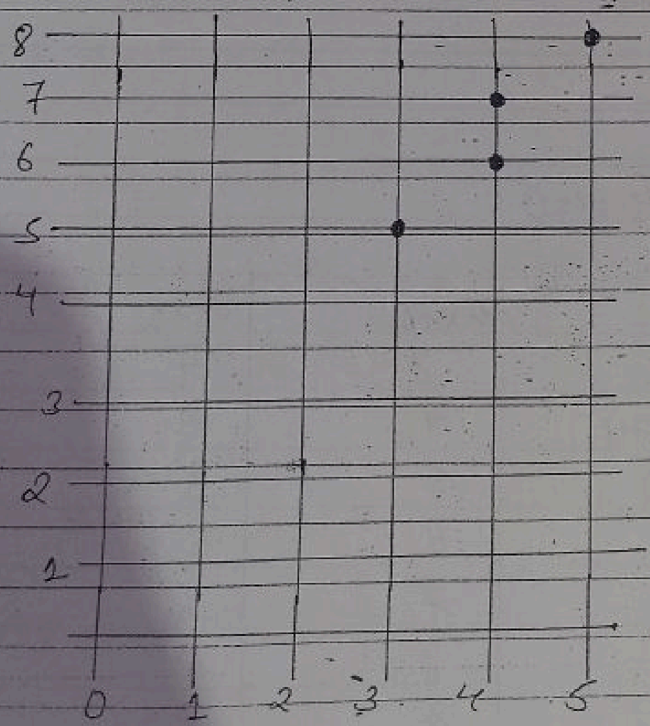


[m>1; moving from L-R]

$\Delta y = 1$

$\Rightarrow x_{k+1} = x_k + \frac{1}{m}$

$x_{k+1} = x_k + \frac{1}{m}$	$y_{k+1} = y_k + 1$	rounded value (x_{k+1})
$2 + 3/4 = 2.75$	5	3
$2.75 + 3/4 = 3.5$	6	4
$3.5 + 3/4 = 4.25$	7	4
$4.25 + 3/4 = 5$	8	5



Date
Page

~~BRESE~~

BRESENHAM'S LINE DRAWING ALGORITHM FOR $|m| \leq 1$.

1: Given 2 end points, plot first pixel (x_0, y_0)
2: Calculate $\Delta x, \Delta y, 2\Delta y - 2\Delta x$
3: Obtain initial decision parameter as $P_0 = 2\Delta y - \Delta x$
4: For each succeeding columns:
   if $P_k < 0$ then next pixel to plot is (x_{k+1}, y_k) &
   $P_{k+1} = P_k + 2\Delta y$
   else if
   $P_k \geq 0$ then next pixel to plot is (x_{k+1}, y_{k+1})
5: Repeat step 4 Δx times;
   $P_{k+1} = P_k + 2\Delta y - 2\Delta x$

Q: Use BLA to plot a line with end points $A(2,3)$ $B(16,10)$ $m = ?$

→ $\Delta y = 10 - 3 = 7$
$\Delta x = 16 - 2 = 14$
$P_0 = 2\Delta y - \Delta x = 14 - 14 = 0$

k	P_k	x_{k+1}	y_{k+1}
0	$P_0 = 0$	3	4
1	$P_1 = 0 + (-14) = -14$	4	4
2	$P_2 = -14 + 14 = 0$	5	5
3	$P_3 = -14$	6	5
4	$P_4 = 0$	7	6
5	$P_5 = -14$	8	6
6	$P_6 = 0$	9	7

7	$P_7 = -14$	10	7
8	$P_8 = 0$	11	8
9	$P_9 = -14$	12	8
10	$P_{10} = 0$	13	9
11	$P_{11} = -14$	14	9
12	$P_{12} = 0$	15	10
13	$P_{13} = -14$	16	10

Processes:-

$\Delta y = y_{k+1} - y_k$

$\Delta x = x_{k+1} - x_k$

$m = \frac{\Delta y}{\Delta x} = \frac{10-3}{16-2} = 0.5$

$|m| < 1$

* $y_{k+1} = y_k$

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

* $x_{k+1} = x_k + 1$

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

$\Delta x = 14$

$\Delta y = -7.$

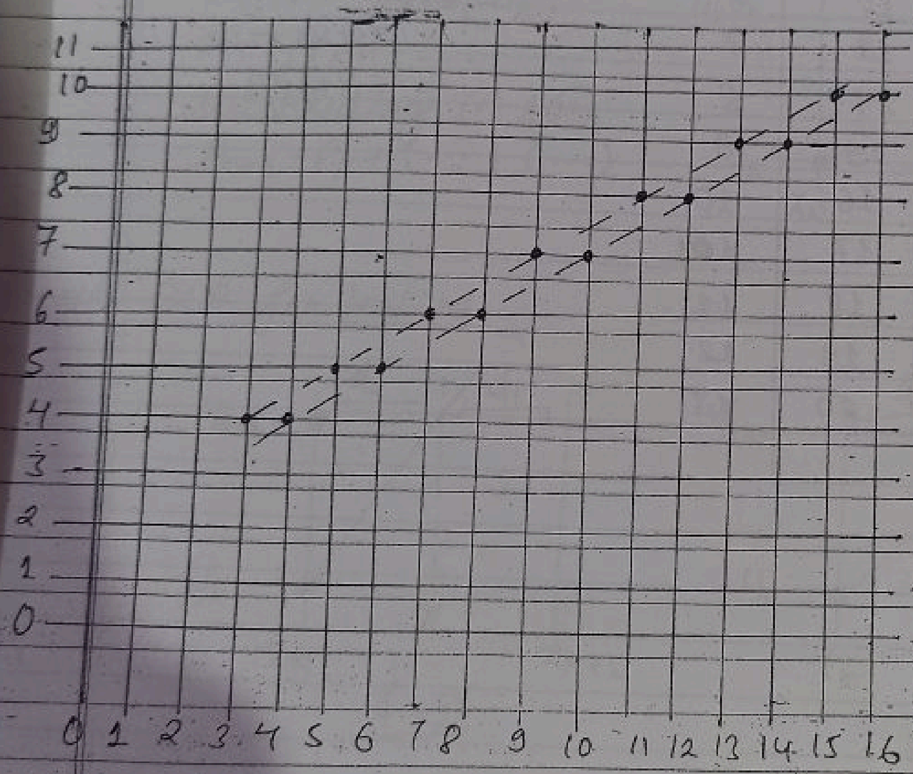
$P_0 = \underline{2\Delta y - \Delta x} = 0$

If $P_k > 0$, $x_k + 1, y_k$

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

If $P_k \geq 0$, x_{k+1}, y_{k+1}

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



Date ______
Page ______

Q: Use BLA to plot a line with end points A(10,5) B(20,13).

→ $m = \frac{13-5}{20-10} = \frac{8}{10} = 0.8$

$P_0 = 2\Delta y - \Delta x = 2 \times 4 - 5 = 8 - 5 = 3$

k	P_k	x_{k+1}	y_{k+1}
0	$P_0 = 3$	11	6
1	$P_1 = 1$	12	7
2	$P_2 = -1$	13	7
3	$P_3 = 7$	14	8
4	$P_4 = 5$	15	9
5	$P_5 = 3$	16	10
6	$P_6 = 1$	17	11
7	$P_7 = -1$	18	11
8	$P_8 = 7$	19	12
9	$P_9 = 5$	20	13

if $P_k < 0$

(x_{k+1}, y_k)

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

else

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

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

$= 3 + 2 \times 4 - 10$

$= 3 + 8 - 10$

$y = m(x_k+1)+c$

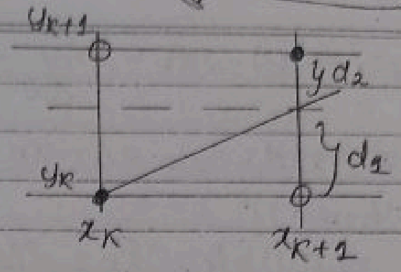
$d_1 = y - y_k$

$= m(x_k+1)+c-y_k$

$d_2 = y_{k+1} - y$

$= y_{k+1} - m(x_k+1) - c$

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



$d_1 - d_2 > 0$

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

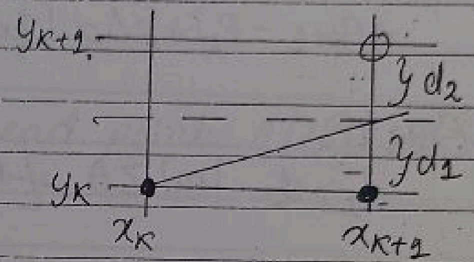
$y_{k+1} = y_k + 1$

k^{th}, $P_k = \Delta x(d_1 - d_2)$

$k+1^{th}$, $P_{k+1} = \cdots$

substituting ① & ⑪,

$P_{k+1} = P_k + (\cdots)(\cdots)$



$d_1 - d_2 < 0$

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

$y_{k+1} = y_k$

BLA for $|m| > 1$

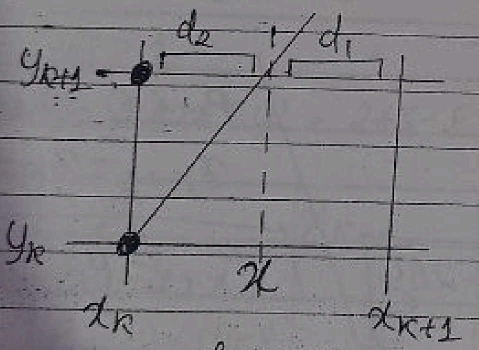


fig ①

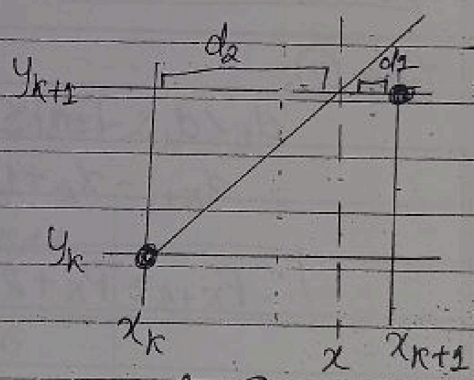


fig ⑪

Now, $d_1 = x_k + 1 - x$

$d_2 = x - x_k$

Again,

$x = m(y_k+1)+c$

$\therefore d_1 - d_2 = x_k + 1 - x - x + x_k$

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

Date
Page

$\Rightarrow (d_1 - d_2) = 2x_k - \frac{2\Delta y}{\Delta x} y_k - \frac{2\Delta y}{\Delta x} - 2C$

$\Rightarrow \Delta x(d_1 - d_2) = 2\Delta x \cdot x_k - 2\Delta y \cdot y_k - 2\Delta y - 2C$

Then,

$\Rightarrow k^{th}$ step, $P_k = 2\Delta x \cdot x_k - 2\Delta y \cdot y_k + b$

$\Rightarrow (k+1)^{th}$ step, $P_{k+1} = 2\Delta x \cdot x_{k+1} - 2\Delta y \cdot y_{k+1} + b$

$\therefore P_{k+1} - P_k = 2\Delta x \cdot x_k - 2\Delta y \cdot y_k - 2\Delta x \cdot x_{k+1} + 2\Delta y \cdot y_{k+1}$

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

Case ①,

$d_1 > d_2 \quad |m| \geq 0 \quad \cancel{[P_k \geq 0]}$

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

$\boxed{\therefore P_{k+1} = P_k - 2\Delta y}$ $\cancel{\therefore x_k =}$

$P_k > 0$ then,

$x_{k+1} = x_k$

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

Case ⑪,

$d_1 < d_2 \quad |m| \leq 0$

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

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

$P_k < 0$ then

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

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

Q: Digitize a line with end-points A(2,5) B(4,10) using BLA.

soln $\Delta x = 4 - 2 = 2$, $\Delta y = 10 - 5 = 5$

$\therefore m = \frac{\Delta y}{\Delta x} = \frac{5}{2} = 2.5$

$P_k > 0$,

$x_{k+1} = x_k$

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

$P_k \le 0$ then $x_{k+1} = x_k + 1$

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

$P_0 = 2\Delta x - \Delta y$
$= 2*2 - 5 = 4-5 = -1$
$= -1 - 2*2 + 2*5$
$= -1 - 4 + 10$
$= -5 + 10 = 5$

k	P_k	x_{k+1}	y_{k+1}
0	$P_0 = -1$	3	6
1	$P_1 = 5$	3	7
2	$P_2 = 1$	3	8
3	$P_3 = -3$	3	9
4	$P_4 = 3$	4	10

Q: Digitize a line with end points A(2,4) B(5,12) using BLA

sol^n $\Delta x = 5 - 2 = 3$

$\Delta y = 12 - 4 = 8$

$$m = \frac{\Delta y}{\Delta x} = \frac{8}{3}$$

$|m| > 1$

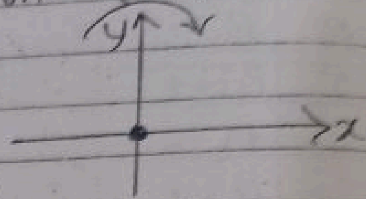
$P_0 = 2\Delta x - \Delta y = 2*3 - 8 = 6 - 8 = -2$

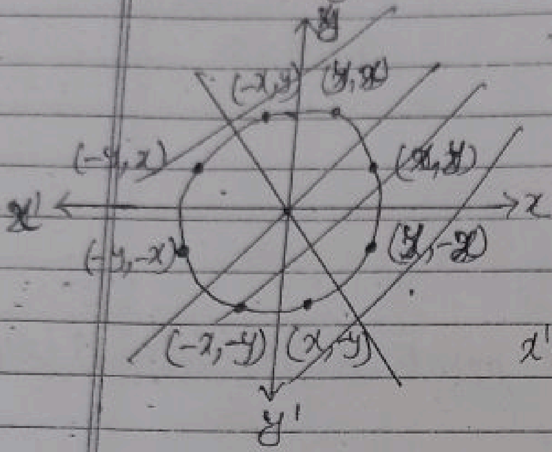
k	P_k	x_{k+1}	y_{k+1}
0	$P_0 = -2$	3	5
1	$P_1 = -2-6+16 = 8$	4	6
2	$P_2 = 8-6 = 2$	3	7
3	$P_3 = 2-6 = -4$	4	8
4	$P_4 = +4-6+16 = 6$	4	9
5	$P_5 = 6-6 = 0$	5	10
6	$P_6 = 0+10 = 10$	5	11
7	$P_7 = 10-6 = 4$	5	12

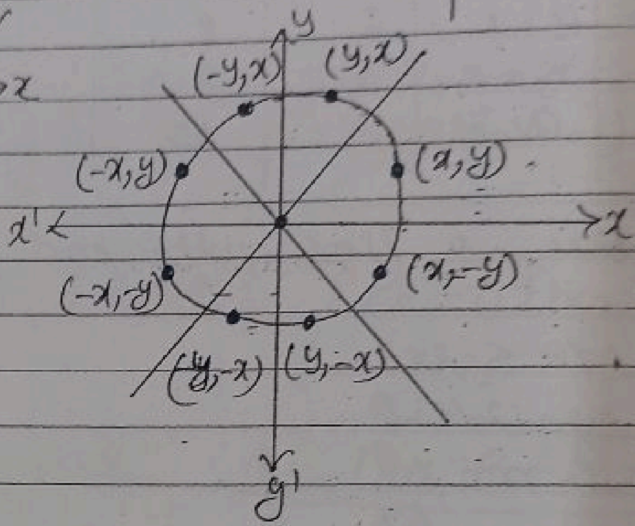
Mid-Point Circle Algorithm:-

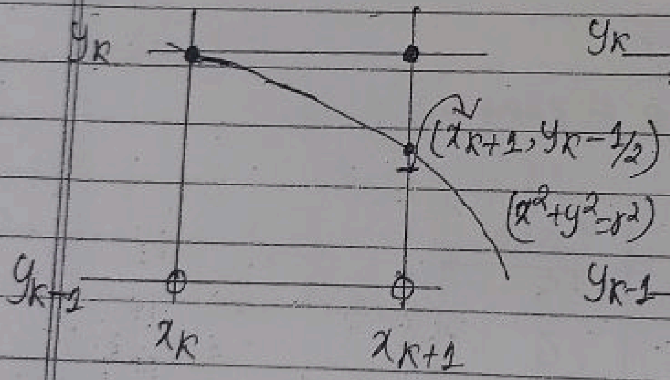
(i) Circle is centered at origin

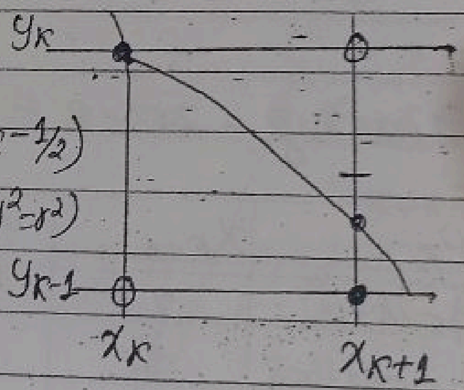
(ii) Moving in clock wise direction. r(0,r)











$$\boxed{\begin{aligned} x_{k+1} &= x_k + 1 \\ y_{k+1} &= y_k \end{aligned}}$$

$$\boxed{\begin{aligned} x_{k+1} &= x_k + 1 \\ y_{k+1} &= y_k - 1 \end{aligned}}$$

k^{th} step:-

$$P_k = f\,circle\left(x_k + 1,\ y_k - \tfrac{1}{2}\right) = (x_k+1)^2 + \left(y_k - \tfrac{1}{2}\right)^2 - r^2$$

$k+1^{th}$ step:-

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

$$= (x_{k+1})^2 + 2(x_{k+1}) + 1 + (y_{k+1} - 1/2)^2 - r^2$$

Subtracting P_k from P_{k+1};

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

Case ①

$P_k < 0$ then,

$y_{k+1} = y_k$

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

$\therefore$ eq^n becomes;

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

Case ⑪

$P_k < 0$ then,

$P_{k+1} = P_k + 2x_{k+1} + 1 + \left(y_k - \frac{3}{2}\right)^2 - \left(y_k - \frac{1}{2}\right)^2$

Again,

$P_{k+1} = P_k + 2x_{k+1} + 1 + y_k^2 - 3y_k + \frac{9}{4} - y_k^2 + y_k - \frac{1}{4}$

$= P_k + 2x_{k+1} + 1 - 2y_k + 2$

$= P_k + 2x_{k+1} + 1 - 2(y_k - 1)$

$\boxed{\text{so, } P_{k+1} = P_k + 2x_{k+1} - 2y_{k+1} + 1}$

Also,

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

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

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

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

Mid-Point Circle Algorithm:-

Assumptions:- Circle is centered at origin, moving in clock-wise direction.

Equation of circle is defined by $x^2+y^2=r^2$, to apply mid-point method we define a circle function as

$$F_{circle}(x,y) = x^2+y^2-r^2$$

Now,

$$F_{circle}(x,y)\begin{cases} <0 \text{ if } (x,y) \text{ is inside circle boundary} \\ =0 \text{ if } (x,y) \text{ is on circle boundary} \\ >0 \text{ if } (x,y) \text{ is outside circle boundary} \end{cases}$$

The circle function $F_{circle}(x,y)$ serves as the decision parameter, select the next pixel along circle path according to the sign of the circle funⁿ evaluated at the mid-point betⁿ 2 candidate pixels.

Start at $(0,r)$ take until steps in 'x' direⁿ (sample in x-direⁿ; $x_{k+1} = x_k+1$)

Assuming position (x,y) has been selected in previous step, we determine next position (x_{k+1}, y_{k+1}) as either (x_k+1, y_k) or (x_k+1, y_k-1) along circle path by evaluating the decision parameter (circle function). The decision parameter is the circle function evaluated at the mid-point betⁿ these 2 pixels.

i.e. $P_k = f_{circle}\left(x_k+1, y_k-\frac{1}{2}\right)$

$$= (x_k+1)^2 + (y_k-1/2)^2 - r^2 \ldots (1)$$

At next sampling position $(x_{k+1}+1 = x_k+1)$

$\Rightarrow P_{k+1} = f_{circle}\left(x_{k+1}+1, y_{k+1}-1/2\right)$

$$=[(x_{k+1})+1]^2+(y_{k+1}-\tfrac{1}{2})^2-r^2$$
$$=(x_{k+1})^2+(x_{k+1})+1$$
$$=(y_{k+1}-\tfrac{1}{2})^2-r^2 \ldots (ii)$$

Now, subtracting (i) & (ii)

$$P_{k+1}=P_k+2(x_k+1)+(y^2_{k+1}-y_k^2)-(y_{k+1}-y_k)+1 \ldots (iii)$$

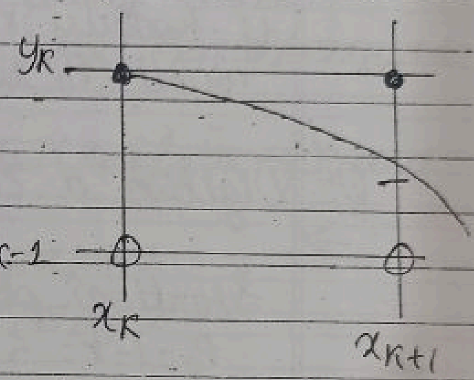
where,

y_{k+1} is either y_k or y_{k-1}, depending on the sign of P_k.

Case (i)

if $P_k<0$ then mid-point is inside the circle so, pixel on the scanline 'y_k' is closer to circle boundary &

$y_{k+1}=y_k$ eq^n (iii) becomes

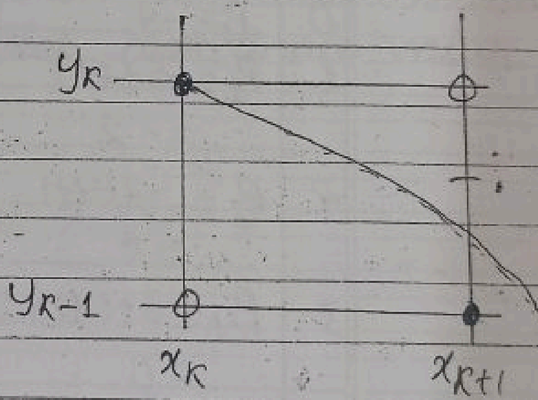


$$P_{k+1}=P_k+2x_{k+1}+1$$

where, $x_{k+1}=x_k+1$

Case (ii)

if $P_k\geq 0$ then mid-point is outside the circle boundary so, we select pixel on the scan-line y_{k-1} i.e. $y_{k+1}=y_k-1$



so, eq^n (iii) becomes;

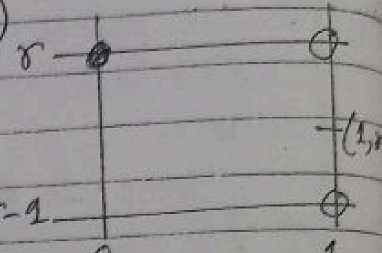
$$P_{k+1}=P_k+2x_{k+1}-2y_{k+1}+1 \text{ Ok}$$

where,

$$x_{k+1}=x_k+1$$
$$y_{k+1}=y_k-1$$

The initial decision parameter is obtained by evaluating the circle function at the start position $(x_0, y_0) = (0, r)$; next point to plot is $(1, r)$ on $(1, r-1)$

So, mid-point coordinate is $(1, r - \frac{1}{2})$



Then,

$P_0 = f_{circle}(1, r - \frac{1}{2}) = 1 + (r - \frac{1}{2})^2 - r^2$

$\Rightarrow P_0 = \frac{5}{4} - r$

if radius 'r' is specified as an integer, we can round P_0 to $P_0 = 1 - r$

Q: Digitize a circle with radius 6 pixels.

→

Starting pixel = (0, 6)

$P_0 = 1 - r = 5$

if $P_k < 0$,

$y_{k+1} = y_k$

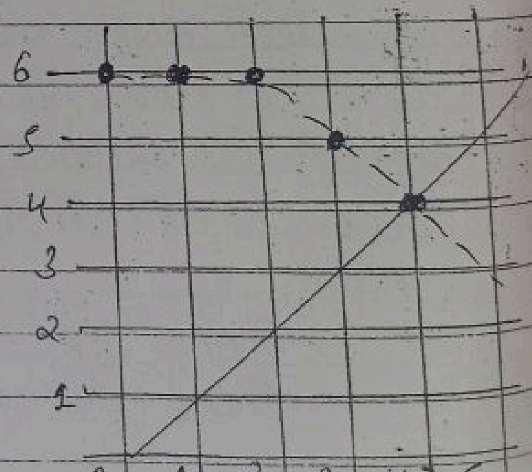
$P_{k+1} = P_k + 2x_{k+1} + 1$

else

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

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

k	P_k	x_{k+1}	y_{k+1}
0	$P_0 = -5$	1	6
1	$P_1 = -5+2+1 = -2$	2	6
2	$P_3 = -2+4+1 = 3$	3	5
3	$P_4 = 3+6-10+1 = 0$	4	4



Mid-point Circle Algorithm:-

1: Input radius 'r' and circle (x_c, y_c) and obtain the first point on the circumference of a circle centered on origin as $(x_0, y_0) = (0, r)$

2: Calculate initial value of decision parameter as $P_0 = 1 - r$

3: At each x_k position starting at $k=0$ perform following tests: if $P_k < 0$

then next pixel to plot along circle centered on $(0,0)$ is (x_k+1, y_k) and $P_{k+1} = P_k + 2x_{k+1} + 1$

else

the next point to plot is (x_k+1, y_k-1) and

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

where $2x_{k+1} = 2x_k + 2$

$2y_{k+1} = 2y_k - 2$

4: Determine symmetry point in other 7 Octants.

5: Move each calculated pixel position (x, y) onto circular path centered on (x_c, y_c) and plot coordinate values

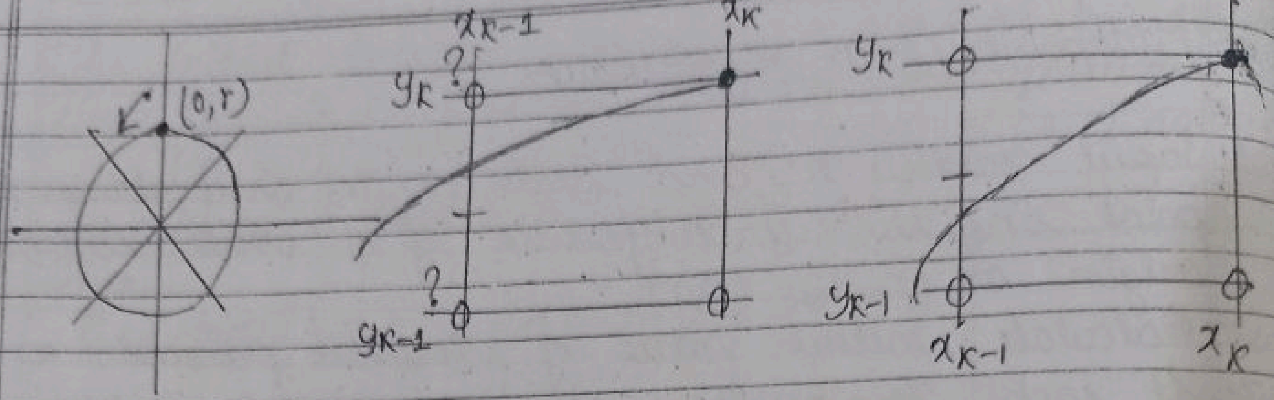
$X = x + x_c$

$Y = y + y_c$

6: Repeat steps 3 through 5 until $x \geq y$.

Date
Page

Q:



Here,

$$P_k = F_{circle}\left(x_k - 1, y_k - \frac{1}{2}\right)$$

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

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

Case I

$x_{k+1} = x_k - 1$

$y_{k+1} = y_k$

Case II

$x_{k+1} = x_k - 1$

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

Date ______
Page ______

Q. Digitize a circle with radii of 7 pixels using mid-point circle approach.

sol^n Starting pixel = (0,7)

$P_0 = 1 - r = 1 - 7 = -6$

$P_k < 0$ = mid-point lies in circle

if $P_k < 0$, $y_{k+1} = y_k$

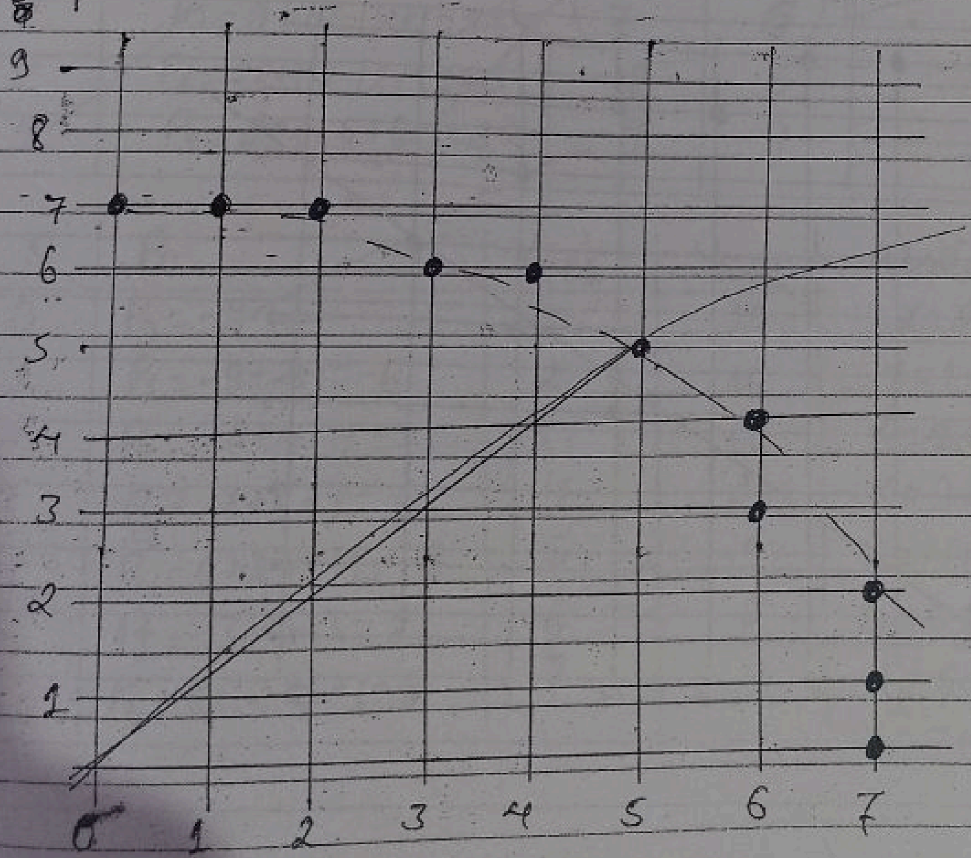
$P_{k+1} = P_k + 2x_{k+1} + 1$

else

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

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

k	P_k	x_{k+1}	y_{k+1}
0	$P_0 = -6$	1	7
1	$P_1 = -6+2+1 = -3$	2	7
2	$P_2 = 2$ ($\because -3+4+1$)	3	6
3	$P_3 = 2+6-12+1 = -3$	4	6
4	$P_4 = -3+8+1 = 6$	5	5
5			
6			
8			



Q: For 8 pixels:-

sol^n $P_0 = 1 - r = 1 - 8 = -7$

Starting pixel = (0,8)

k	P_k	x_{k+1}	y_{k+1}
0	$P_0 = -7$	1	8
1	$P_1 = -7+2+1 = -4$	2	8
2	$P_2 = -4+4+1 = 1$	3	7
3	$P_3 = 1+2*3-2*7+1 = -6$	4	7
4	$P_4 = 3+10-12+1 = 3$	5	6
5	$P_5 = +2$	6	5 (flip case)
6	$P_6 = 2+12-10+1 = 5$	5	4
7	$P_7 = 5+10-8+1 = 8$	4	3

if $P_k < 0$,

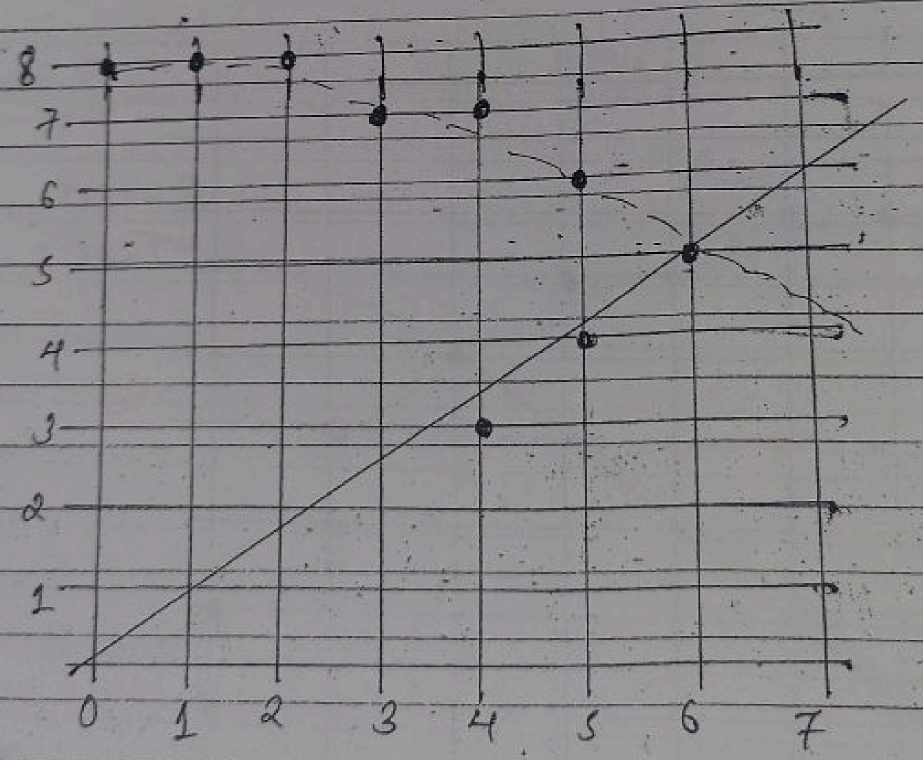
$y_{k+1} = y_k$

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

else

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

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



Q: Digitize a circle with radii of 10 pixels and circle centered at (4,5).

Sol^n $P_0 = 1 - r = 1 - 10 = -9$

Starting pixel = (0,10)

$\Rightarrow$ (4,15)

K	P_{K+1}	x_{k+1}	y_{k+1}
0	$P_0 = -9$	0	10
1	$P_1 = -9 + 2\times0 + 1 = -8$	1	10
2	$P_2 = -8 + 2 + 1 = -5$	2	10
3	$P_3 = -5 + 4 + 1 = 0$	3	10
4	$P_4 = 0 + 6 - 20 + 1 = 23$	4	9
5	$P_5 = 23 + 8 - 18 + 1 = 14$	5	8
6	$P_6 = 14 + 10 - 16 + 1 = 9$	6	7
7	$P_7 = 9 + 12 - 14 + 1 = 22$	7	6
8	$P_8 = 22 + 14 - 12 + 1 = 25$	8	5
9	$P_9 = 25 + 16 - 10 + 1 = 32$	9	4

flip-case

K	P_K	x_{k+1}	y_{k+1}
0	$P_0 = -9$	1	10
1	$P_1 = -9 + 2 + 1 = -6$	2	10
2	$P_2 = -6 + 4 + 1 = -1$	3	10
3	$P_3 = -1 + 6 + 1 = 6$	4	9
4	$P_4 = 6 + 8 - 18 + 1 = -3$	5	9
5	$P_5 = -3 + 10 + 1 = 8$	6	8
6	$P_6 = 8 + 12 - 16 + 1 = 5$	7	7

Also,

$X = X + X_C$ $Y = Y + Y_C$

$X = 1 + 4 = 5$ $Y = 10 + 5 = 15$

$X = 2 + 4 = 6$ $Y = 10 + 5 = 15$

$X = 3 + 4 = 7$ $Y = 9 + 5 = 14$

$X = 4 + 4 = 8$ $Y = 9 + 5 = 14$

$X = 5 + 4 = 9$ $Y = 8 + 5 = 13$

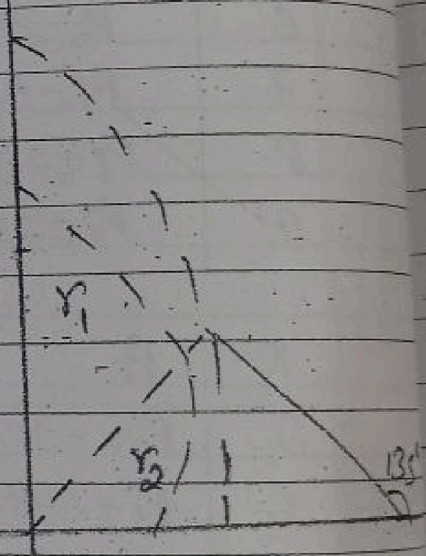
$X = 6 + 4 = 10$ $Y = 8 + 5 = 13$

$X = 7 + 4 = 11$ $Y = 7 + 5 = 12$

Next method

k	P_k	$x_{k+1}+4$	$y_{k+1}+5$
0	$P_0=-9$	5	15
1	$P_1=-9+10+1=2$	6	14
2	$P_2=2+12-28+1=-13$	7	14
3	$P_3=-13+14+1=2$	8	13
4	$P_4=2+16-26+1=-7$	9	13
5	$P_5=-7+18+1=12$	10	12
6	$P_6=12+20-26+1=7$	11	12
7			

Mid-point Ellipse Algorithm:-



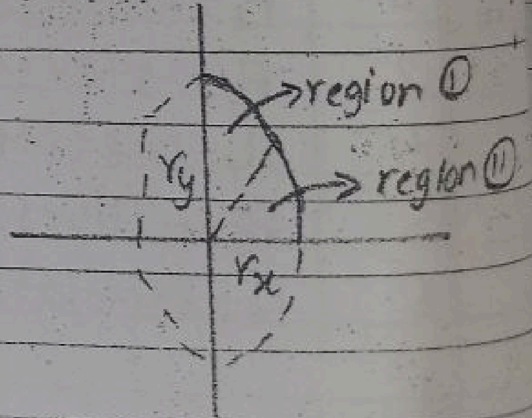
Here,

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

Equation of ellipse:-

$$f_{ellipse}=r_y^2x^2+r_x^2y^2-r_x^2r_y^2$$

$$\text{slope}=2xr_y^2+2yr_x^2\frac{dy}{dx}=0$$



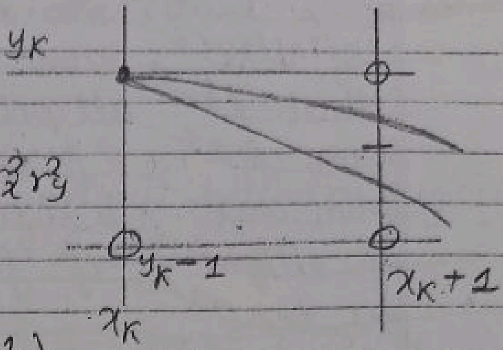
$$\Rightarrow \frac{dy}{dx}=-\frac{2xr_y^2}{2yr_x^2}$$

$$\Rightarrow -1=-\frac{2xr_y^2}{2yr_x^2}$$

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

$\Rightarrow 2xr_y^2 \geq 2yr_x^2$ then goes to region (II)

Region - I



$$P1_k = f_{ellipse}\left(x_k + 1,\ y_k - \frac{1}{2}\right)$$

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

In $k+1^{th}$ step;

$$P1_{k+1} = f_{ellipse}\left(x_{k+1} + 1,\ y_{k+1} - \frac{1}{2}\right)$$

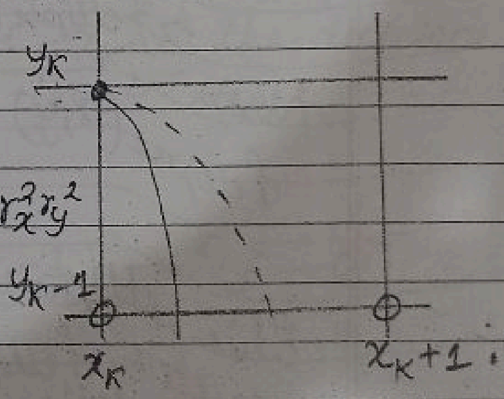
$$= r_y^2 \left[(x_k + 1) + 1\right]^2 + r_x^2 \left(y_{k+1} - \frac{1}{2}\right)^2 - r_x^2 r_y^2$$

$$= P1_k + 2r_y^2 (x_k + 1) + r_y^2 + r_x^2 \left[\left(y_{k+1} - \frac{1}{2}\right)^2 - \left(y_k - \frac{1}{2}\right)^2\right]$$

Then, $x_{k+1} = x_k + 1$

$y_{k+1} = y_k$ or, $y_k - 1$

Region - II



$$P2_k = f_{ellipse}\left(x_k + \frac{1}{2},\ y_k - 1\right)$$

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

$$P2_{k+1} = f_{ellipse}\left(x_{k+1} + \frac{1}{2},\ y_{k+1} - 1\right)$$

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

Then,

$x_{k+1} = x_k$ or $x_k + 1$

and $y_{k+1} = y_k - 1$

Definition :- An ellipse is defined as a set of points such that the sum of distances from 2 fixed points is the same for all points. If the distance to two fixed points from any $P(x,y)$ on ellipse be d_1, d_2 then general eqn of ellipse is d_1+d_2 = constant on expressing distance d_1, d_2 in terms of focal coordinates $F_1=(x_1,y_1)$
$F_2=(x_2,y_2)$

We have,

$$\sqrt{(x-x_1)^2+(y-y_1)^2}+\sqrt{(x-x_2)^2+(y-y_2)^2} = \text{constant}$$

Mid-point ellipse method is applied throughout the first quadrant in two points, according to the slope of ellipse.

The ellipse eqn is given by; $\frac{x^2}{r_x^2}+\frac{y^2}{r_y^2}=1$

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

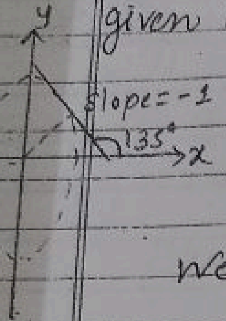
Now,

$f_{ellipse}(x,y)$ <0 if (x,y) is inside ellipse boundary
$=0$ if (x,y) is on ellipse boundary
>0 if (x,y) is outside ellipse boundary

This ellipse function serves as the decision parameter $f_{ellipse}(x,y)$.

We select next pixel along ellipse path according to sign of ellipse fun evaluated at the mid-point betn two candidate pixels.

Start at $(0, r_y)$ take unit steps in 'x'-direction until we reach boundary between region 1 and 2 then switch to unit steps in 'y'-direction for remainder of curve in first quadrant at each step test the value of the slope of the curve. The ellipse slope is given by: $2r_y^2x + 2r_x^2y\frac{dy}{dx} = 0$



$\Rightarrow \frac{dy}{dx} = -\frac{2r_x^2y}{2r_y^2x}$ & at boundary between region 1 and 2 $\frac{dy}{dx} = -1$

so, $2r_y^2x = 2r_x^2y$

we move out of region 1 when $2xr_y^2 \geq 2r_x^2y$

Region 1

Assuming that position (x_k, y_k) has been selected at previous step, we determine next position (x_{k+1}, y_{k+1}) as either (x_k+1, y_k) or (x_k+1, y_k-1) along elliptic path by evaluating the decision function.

Elliptic function $P1_k = f_{ellipse}(x_k+1, y_k-\frac{1}{2})$

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

At next sampling position $(x_{k+1}+1 = x_k+2)$ the decision parameter for region 1 will be $(k+1^{th}$ step);

$$P1_{k+1} = f_{ellipse}(x_{k+1}+1, y_{k+1}-\tfrac{1}{2})$$

$$= r_y^2[(x_k+1)+1]^2 + r_x^2(y_{k+1}-\tfrac{1}{2})^2 - r_x^2r_y^2 \quad \cdots (ii)$$

Subtracting (i) & (ii),

$$P1_{k+1} = P1_k + 2r_y^2(x_k+1) + r_y^2 + r_x^2\left[(y_{k+1}-\tfrac{1}{2})^2 - (y_k-\tfrac{1}{2})^2\right] \quad \cdots (iii)$$

where, y_{k+1} is either y_k or y_{k-1} depending on the sign of $P1_k$.

Case (1):-

If $P1_k < 0$ then mid-point is inside ellipse so pixel on scanline 'y_k' is closer to ellipse boundary and $y_{k+1} = y_k$

so, eq^n (iii) becomes,

$$P1_{k+1} = P1_k + 2r_y^2 x_{k+1} + r_y^2 \ldots (a)$$

where, $x_{k+1} = x_k + 1$

$$2r_y^2 x_{k+1} = 2r_y^2 x_k + 2r_y^2$$

Case (2):-

if $P1_k \geq 0$ then mid-point is on or outside ellipse, so we select pixel on scanline '$y_k - 1$' then

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

so, eq^n (iv) becomes;

$$P1_{k+1} = P1_k + 2r_y^2(x_{k+1}) + r_y^2 - 2r_x^2(y_k - 1)$$

$$= P1_k + 2r_y^2 x_{k+1} + r_y^2 - 2r_x^2 y_{k+1}$$

where, $y_{k+1} = y_k - 1$

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

and $2r_x^2 y_{k+1} = 2r_x^2 y_k - 2r_x^2$

$\Rightarrow 2r_y^2 x_{k+1} = 2r_y^2 x_k + 2r_y^2$

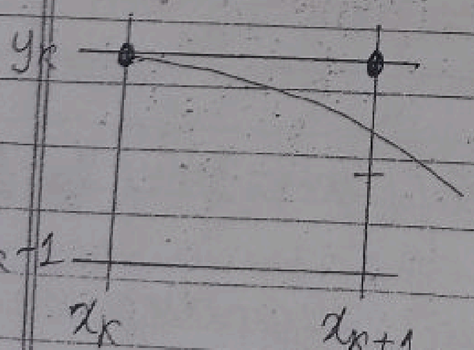


fig:- Case (1)

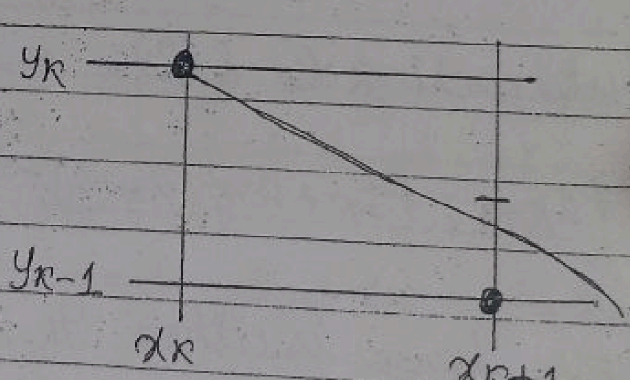
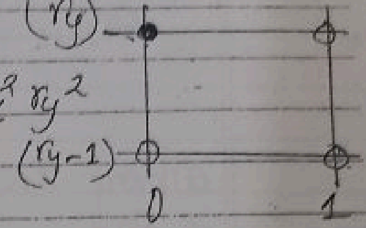


fig:- Case (2)

Q. The initial decision parameter for region 1 at $P1_0 = ?$

→ The starting point is $(0, r_y)$ so, next pixel to plot will be either $(1, r_y)$ or $(1, r_y - 1)$
so, mid-point co-ordinate here is;



$$f_{ellipse}\left(1, r_y - \frac{1}{2}\right) = r_y^2 + r_x^2\left(r_y - \frac{1}{2}\right)^2 - r_x^2 r_y^2$$

Thus, $P1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2$

Region 2 :-

Now, we sample in 'y'-dire^n, the mid-point here is taken bet^n horizontal pixels at each step.
Assuming (x_k, y_k) has already been plotted next pixel to plot (x_{k+1}, y_{k+1}) is either $(x_k, y_k - 1)$ or ~~$(x_{k+1},$~~ $(x_k + 1, y_k - 1)$ and mid-point coordinate is $(x_k + \frac{1}{2}, y_k - 1)$.

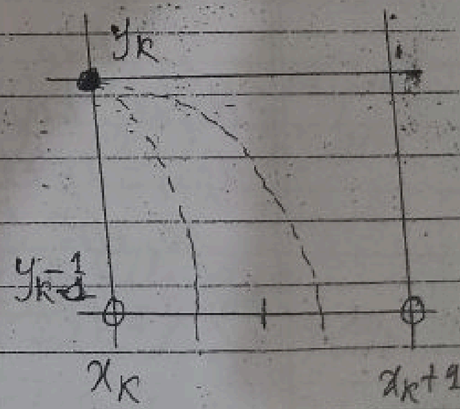
Then,

$$P2_k = f_{ellipse}\left(x_k + \frac{1}{2}, y_k - 1\right)$$

$$= r_y^2\left(x_k + \frac{1}{2}\right) + r_x^2 (y_k - 1)^2 - r_x^2 r_y^2 \quad \cdots (iv)$$

where $P2_k$ is the decision parameters.

Now,
at next sampling position, the next pixel to plot will be either $(x_{k+1} + 1, y_{k+1} - 1)$ or $(x_{k+1}, y_{k+1} - 1)$



$$P2_{k+1} = f_{ellipse}\left(x_{k+1} + \frac{1}{2}, y_{k+1} - 1\right)$$

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

$= r_y^2(x_{k+1}+\frac{1}{2}) + r_x^2(y_{k-1}-1)^2 - r_x^2 r_y^2 \ldots$ (V)

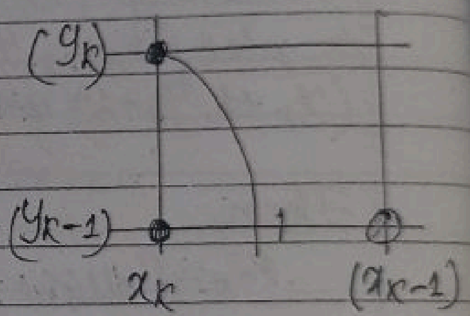
Subtracting (IV) & (V),

$P2_{k+1} = P2_k - 2r_x^2(y_k - 1) + r_x^2 + r_y^2\left[(x_{k+1}+\frac{1}{2})^2 - (x_k+\frac{1}{2})^2\right] \ldots$ (VI)

where x_{k+1} is either x_k+1 or x_k depending on sign of $P2_k$.

Case (1)

if $P2_k > 0$ then mid-point is outside ellipse boundary so we select pixel at x_k i.e. $x_{k+1} = x_k$ and eq^n^ (IV) becomes,



$P2_{k+1} = P2_k - 2r_x^2(y_k - 1) + r_x^2$

$= P2_k - 2r_x^2 y_{k+1} + r_x^2$

where,

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

& $2r_x^2 y_{k+1} = 2r_x^2 y_k - 2r_x^2$

Case (2)

if $P2_k < 0$ then mid-point is inside or on ellipse boundary so we select pixel at 'x_k+1' i.e. $x_{k+1} = x_k + 1$

so, eq^n^ (VI) becomes;

$P2_{k+1} = P2_k - 2r_x^2(y_k - 1) + r_x^2 + r_y^2\left[(x_{k+1}+\frac{1}{2})^2 - (x_k+\frac{1}{2})^2\right]$

$= P2_k - 2r_x^2(y_k - 1) + r_x^2 + r_x^2 \cdot 2r_y^2\left[x_k + 1\right]$

$= P$

$= P2_k - 2r_x^2 y_k + 1 + r_x^2 + 2r_y^2 x_k + 1$

where $x_{k+1} = x_k + 1$

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

and,

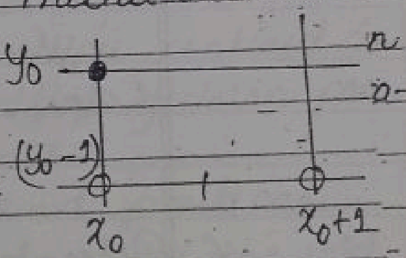
$\Rightarrow 2r_y^2 x_{k+1} = 2r_y^2 x_k + 2r_y^2$

$\Rightarrow 2r_x^2 y_{k+1} = 2r_x^2 y_k - 2r_x^2$

Region ②:- The initial position $(x_0\ y_0)$ is taken as the last position selected in region 1 & the initial decision parameter in region 2 will be;

$$P2_0 = f_{ellipse}\left(x_0 + \frac{1}{2}, y_0 - 1\right)$$

$$= r_y^2\left(x_0 + \frac{1}{2}\right)^2 + r_x^2(y_0 - 1)^2 - r_x^2 r_y^2$$



Mid-Point Ellipse Algorithm:-

1: Input $r_x\ r_y$ and ellipse center (x_c, y_c) and obtain first point on ellipse centered on origin as $(x_0, y_0) = (0, r_y)$

2. Compute initial decision parameter for region 1 as

$$P1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2$$

3. At each x_k position in region 1, starting at $k=0$, perform tests: If $P1_k < 0$, next point pixel to plot is (x_{k+1}, y_k) and and $P1_{k+1} = P1_k + 2r_y^2 x_{k+1} + r_y^2$

else if $P1_k \geq 0$ then next point along ellipse is ~~(x_{k+1}, y_k)~~ (x_k+1, y_k-1) and $P1_{k+1} = P1_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_y^2$ with,

$$2r_y^2 x_{k+1} = 2r_y^2 x_k + 2r_y^2 \text{ and}$$

$$2r_x^2 y_{k+1} = 2r_x^2 y_k - 2r_x^2$$

and continue until $2r_y^2x \geq 2r_x^2y$.

4: Calculate initial value of decision parameter in region ② using last point plotted in region 1 (x_0, y_0) as $P2_0 = r_y^2\left(x_0+\frac{1}{2}\right)^2 + r_x^2(y_0-1)^2 - r_x^2r_y^2$

5: At each y_k position in region ② starting at $k=0$ perform tests:

if $P2_k > 0$ then next point to plot is (x_k, y_k-1) and ~~P2~~ $P2_{k+1} = P2_k - 2r_x^2y_{k+1} + r_y^2$

else if $P2_k \leq 0$ then next point to plot is ~~$(x_{k+1}, y$~~ (x_k+1, y_k-1) and

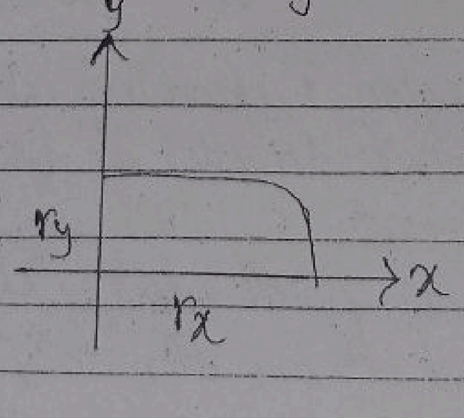
$$P2_{k+1} = P2_k + 2r_y^2x_{k+1} - 2r_x^2y_{k+1} + r_x^2.$$

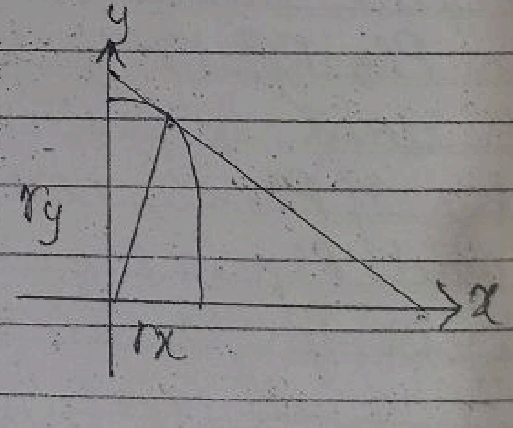
6: Determine symmetry points in other 3 quadrants.

7: Move each calculated pixel position (x,y) onto elliptical path centered on (x_c, y_c) and plot coordinate values $x = x_c + x$, $y = y_c + y$.

8: Repeat steps for region ① until $2r_y^2x \geq 2r_x^2y$.

Q: Trace the algorithm





Date ____
Page ____

Assessment pg:-128

Q: Digitize an ellipse $r_x = 8, r_y = 6$

Sol^n

Given input ellipse parameters $r_x = 8, r_y = 6$, we illustrate the steps in the mid-point ellipse algorithm by determining raster positions along the ellipse path in the 1st quadrant. Initial values and increments for the decision parameters calculations are:-

$$2r_y^2 x = 0 \text{ (with increment } 2r_y^2 = 72)$$

$$2r_x^2 y = 2r_x^2 r_y \text{ (with increment } -2r_x^2 = -128)$$

For region 1: The initial point for the ellipse centered on the origin is $(x_0, y_0) = (0, 6)$ and the initial decision parameter value is: $P1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2 = -332$

Successive decision parameter values and positions along the ellipse path are calculated using the mid-point method as:-

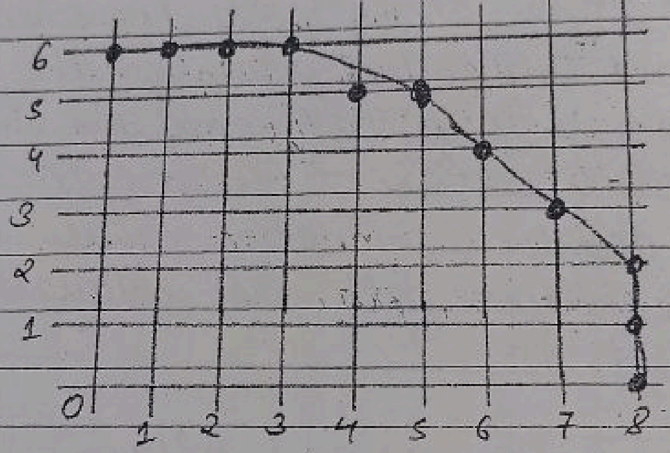
k	$P1_k$	(x_{k+1}, y_{k+1})	$2r_y^2(x_{k+1})$	$2r_x^2 y_{k+1}$
0	-332	(1,6)	72	768
1	-224	(2,6)	144	768
2	-44	(3,6)	216	768
3	208	(4,5)	288	640
4	-108	(5,5)	360	640
5	288	(6,4)	432	512
6	244	(7,3)	504	384

We, now move out of region 1, $\because 2r_y^2 x > 2r_x^2 y$.

For region 2, the initial point is $(x_0, y_0) = (7, 3)$ and the initial decision parameter is $P2_0 = f\left(7 + \frac{1}{2}, 2\right) = -151$

The remaining positions along the ellipse path in the 1st quadrant are then calculated as:-

k	$P2_k$	(x_{k+1}, y_{k+1})	$2r_y^2 x_{k+1}$	$2r_x^2 y_{k+1}$
0	-151	(8,2)	576	256
1	233	(8,1)	576	128
2	745	(8,0)	—	—



Q: Digitize an ellipse with $r_x = 6$, $r_y = 8$

soln

Starting pixel = (0,8)
Ending pixel = (6,0)

Region 1:

$$P1_0 = r_y^2 + \frac{1}{4} r_x^2 - r_x^2 r_y = -215$$

K	$P1_K$	x_{k+1}	y_{k+1}	$2r_y^2 x_{k+1}$	$\geq 2r_x^2 y_{k+1}$
0	-215	1	8	128	576
1	-23	2	8	256	576
2	297	3	7	384	304
3	241	4	6	512	482

Region 2

$P2_0 = \left(4+\frac{1}{2}, 6\right) = -108$

K	$P1_K$	x_{k+1}	y_{k+1}
4	-108	4	6
5	208	5	5
6	-44	6	5
7	544	6	4
8	436	6	3
9	400	6	2
10	292	6	1
11	250	6	0

Area Filling Algorithms:-

(i) Scan Line
(ii) Boundary fill
(iii) Flood fill.

(ii) *Boundary fill (4-connected)*

```
void b_fill (int x, int y){
   newcolor = ~~pixel~~ PixelIntensity (x,y);
   if(newcolor != boundary-color AND newcolor !=
                                       oldcolor)
   { PlotPixel (x, y,newcolor);
     b_fill (x+1, y);
     b_fill(x-1, y);
     b_fill (x, y+1);
     b_fill (x,y-1);
   }
}
```

* *Boundary fill (8-connected)*

```
void b_fill (int x, Int y){
   newcolor= PixelIntensity (x,y);
if (newcolor != boundary_color AND newcolor != oldcolor)
  { PlotPixel (x, y, newcolor);
    b_fill (x+1,y);
    b_fill (x-1, y);
    b_fill (x, y+1);
    b_fill (x, y-1);
    b_fill (x+1, y+1);
```

```
    b_fill (x-1, y+1);
    b_fill (x+1, y-1);
    b_fill (x-1, y-1);
}}
```

(iii) Flood fill

```
void f_fill (int x, int y){
newcolor= PixelIntensity (x,y);
if(newcolor= = oldcolor)
  { PlotPixel (x,y, newcolor);
      f_fill(x+1, y);
      f_fill (x-1, y);
      f_fill (x, y+1);
      f_fill (x, y-1);
}}
```

Chapter:- Two Dimensional Geometric Transformations And Viewing

2D Transformation:-

(i) Translation
(ii) Rotation
(iii) Scaling
(iv) Shear
(V) Reflection

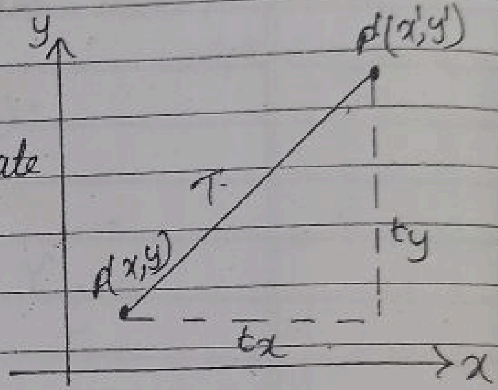
(2,3) - Reference point

$\begin{bmatrix} 2 \\ 3 \end{bmatrix}$ - columnwise Representation

Cm = SROT (R to L - listing)

Matrix multiplication is not commutative.

(i) Translation



Repositioning an object along a straight line path from one coordinate location (x,y) to another (x',y')

i.e: $x' = x + t_x$

$y' = y + t_y$ where (t_x, t_y) is called translation vector writing the eq^n in matrix form using column ~~vector~~ vectors to represent coordinate position and translation vectors.

$x' = x + t_x$

$y' = y + t_y$

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

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

$$P' = P + T \quad \text{or,} \quad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Date
Page

(ii) Rotation

Repositioning an object along a circular path specify rotation angle 'θ' and position (x_r, y_r) of rotation point about which the object is to be rotated.

'+' value for θ defines counter clockwise rotation, '−' value for θ ~~value~~ defines clockwise rotation about a point.

if (x,y) is the original point, 'r' the constant distance from origin 'ϕ' the original angular displacement from x-axis. Now, point (x,y) is rotated through angle 'θ' in a counter clock-wise direction.

Expressing the transformed coordinates interms of 'θ' and 'ϕ' as;

$$x' = r\cos(\phi+\theta)$$
$$= r\cos\phi.\cos\theta - r\sin\phi.\sin\theta \quad \ldots \text{①}$$
$$y' = r\sin(\phi+\theta)$$
$$= r\cos\phi.\sin\theta + r\sin\phi.\cos\theta \quad \ldots \text{②}$$

We know that;

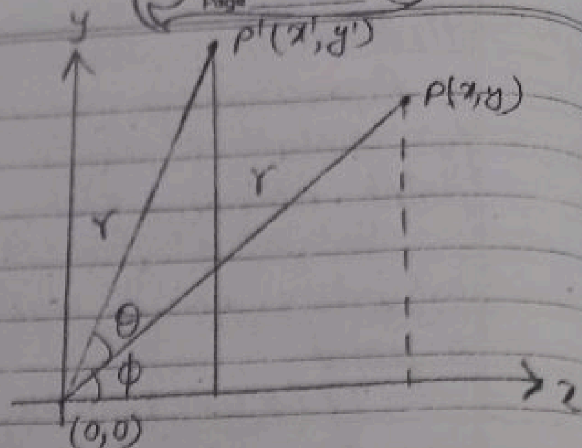
original coordinates of a point in polar coordinates. $x = r\cos\phi$ $y = r\sin\phi$ and substituting these values in ① and ②

$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$ using column vector representation for coordinates in matrix form; $P' = RP$

Assumptions:- (i) about origin
(ii) In anti-clockwise direction
(iii) by 'θ' angle.

Then,
$P' = R_\theta P$

$$\underbrace{\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}}_{\text{Rotation matrix}} \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}}_{\text{points}}$$



*anti-clockwise rotation leads to any 'P' point.

(iii) Scaling

It changes the size of an object (magnify / reduce in size). In case of polygon scaling multiply coordinate values (x, y) of each vertex by scaling factor $s_x s_y$ to produce the final transformed coordinates (x', y').

S_x scales an object in x-direction.

S_y scales an object in y-direction.

$$P = \begin{bmatrix} x \\ y \end{bmatrix}, \; S = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}, \; P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

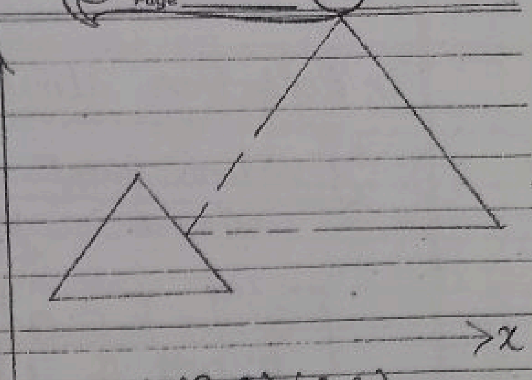
$$\Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\boxed{\therefore P' = S \cdot P}$$

Date ______
Page ______

Q. Values greater than 1 for $S_x S_y$ produce enlargement values less than 1 for $S_x S_y$ reduce size of object $S_x = S_y = 3$ (same) produce the uniform scaling.

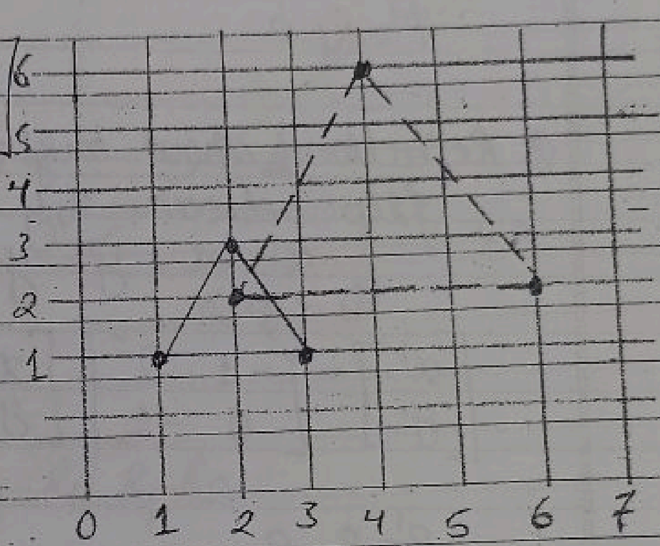
$S_x = 3; S_y = 4$ (different) produce non-uniform scaling.



$\Rightarrow P' = S \cdot P$

$$= \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & 3 & 2 \\ 1 & 1 & 3 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 6 & 4 \\ 2 & 2 & 6 \end{bmatrix}$$



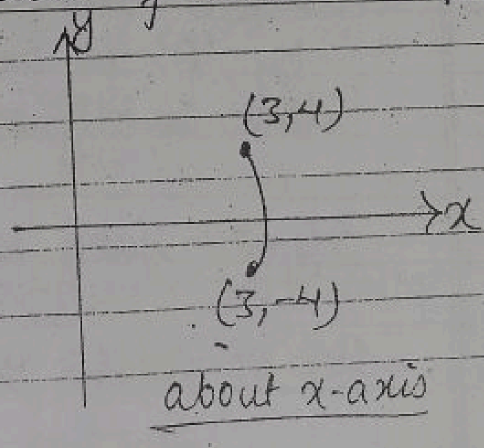
(iv) Reflection.

Generates a mirror image of an object.

a: Reflection about x-axis or about line keep 'x'-value same but flip 'y'-value of coordinate points:-

$x' = x \qquad y' = -y$

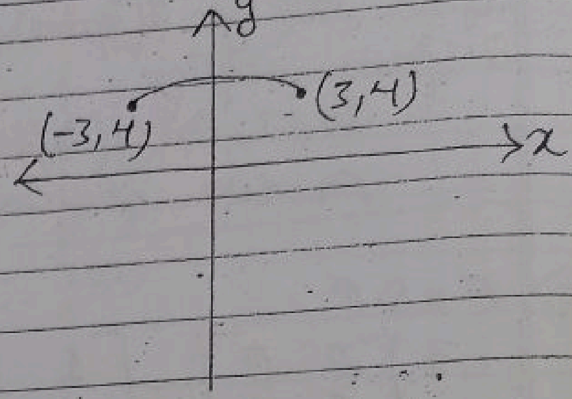
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



Date
Page

b: Reflection about y-axis ~~or~~ about line 'x=0' keep 'y' value same but flip x-value of coordinate points

$x' = -x \quad y' = y$



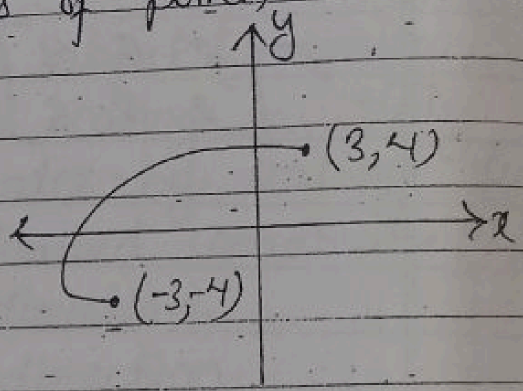
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$P' = R_f \cdot P$

c: Reflection about origin:-

flip between x, y coordinates of point;

$x' = -x, \; y' = -y$



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$P' = R_f \cdot P$

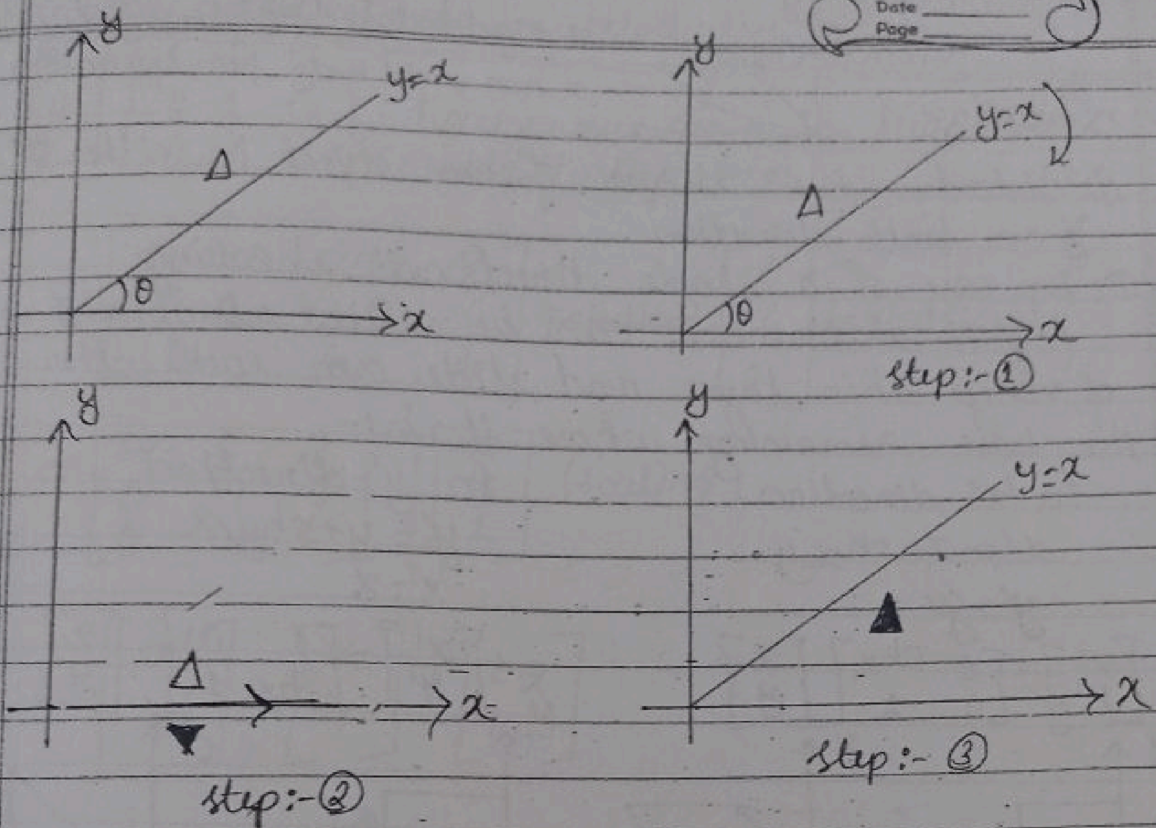
d: Reflection about line y=x

Steps:-

Step 1:- The line y=x along with object is rotated by 'θ' 45° angle in clock-wise dir^n so that the line is aligned with x-axis.

Step 2:- Take reflection of object about x-axis

Step 3:- Rotate the line along with object in reverse direction.



Composite matrix $(C_m) = R_\theta' . R_f . R_\theta$

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

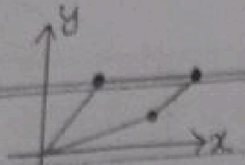
$$= \begin{bmatrix} \cos 45^\circ & -\sin 45^\circ \\ \sin 45^\circ & \cos 45^\circ \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \cos 45^\circ & \sin 45^\circ \\ -\sin 45^\circ & \cos 45^\circ \end{bmatrix}$$

$$= \begin{bmatrix} \cos 45^\circ & \sin 45^\circ \\ \sin 45^\circ & -\cos 45^\circ \end{bmatrix} \begin{bmatrix} \cos 45^\circ & \sin 45^\circ \\ -\sin 45^\circ & \cos 45^\circ \end{bmatrix}$$

$$= \begin{bmatrix} \cos^2 45^\circ - \sin^2 45^\circ & \cos 45^\circ . \sin 45^\circ + \sin 45^\circ . \cos 45^\circ \\ \sin 45^\circ . \cos 45^\circ + \cos 45^\circ . \sin 45^\circ & \sin^2 45^\circ - \cos^2 45^\circ \end{bmatrix}$$

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

(V) Shearing:-



x rigid body

x non-rigid body transformation

(i) Distorts the shape of an object in either x or y or both direction.

(ii) in case of a single directional shearing (e.g: in x-direction can be viewed as objects made of very thin layer and slide over each other with the base remaining where it is:-

In 'x'-direction

$x' = x + sh_x . y$

$y' = y$

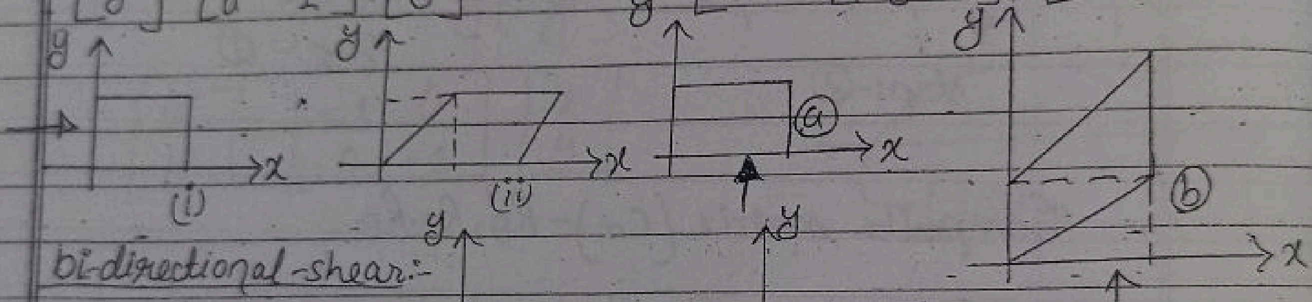
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

In 'y'-direction

$y' = y + sh_y . x$

$x' = x$

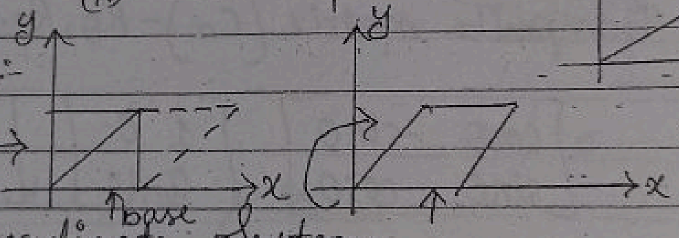
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ Sh_y & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$



bi-directional-shear:-

$x' = x + sh_x y$

$y' = y + sh_y . x$



Homogenous Co-ordinate System:-

$P' = P + T$ (additive) [translation]

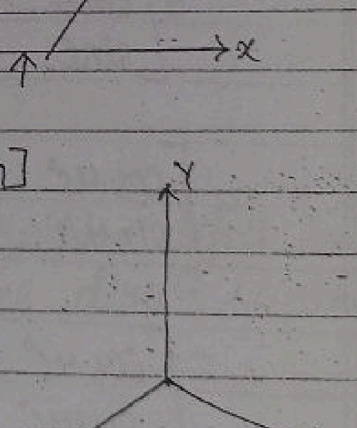
$P' = R * P$ } multiplicative

$P' = S * P$ } [scaling, rotating]

$C_m = STRT$ S: Scale

T: Translate

R: Rotate



There is problem in the size of matrix so, rotation & scaling matrix can also change in homogenous form.

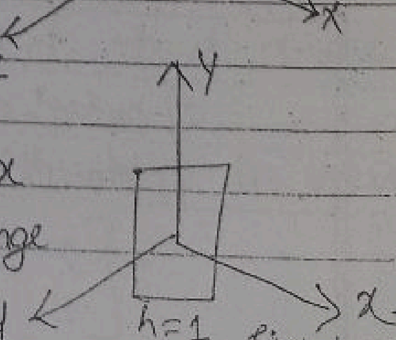


fig-homogenous

To change additive into multiplicative form (translation)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$ = homogenous form in translation (multiplicative form)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cancel{\text{Rotation}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$ (rotation)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$ (scaling)

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

If a body is translate into x & y direction, then it lies in parallel line. This phenomena is called collinearity, ~~above~~ given form shows the fine transformation.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \rightarrow \begin{matrix} x' = x + t \\ y' = y + t \end{matrix}$$

Rotating/scaling → translation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & i & h \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

projection — overall transformation

* <u>Affine transformation:-</u>

$3D \xrightarrow{\text{Projection}} 2D$ [Projection means lose the dimension]

$2D \xrightarrow{\text{Projection}} 1D$

* Composite transformation:- sequence of transformations (composed of more than 1 form of transformation)

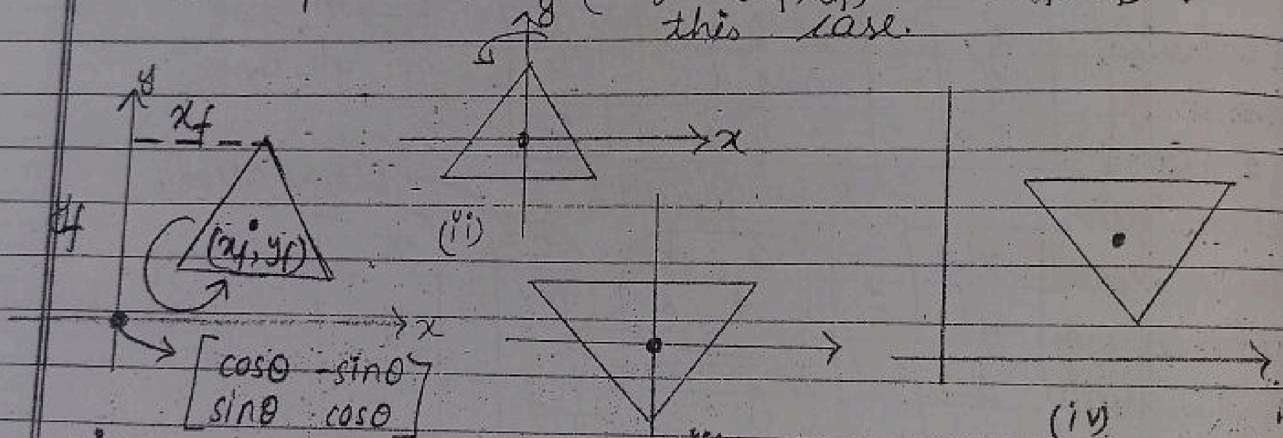
e.g: $C_m = S.T.R.T.$ $\boxed{P' = C_m . P}$ (P = initial location)

Rotation of an object about any fixed point (x_f, y_f) (fixed point rotation)

Steps:-

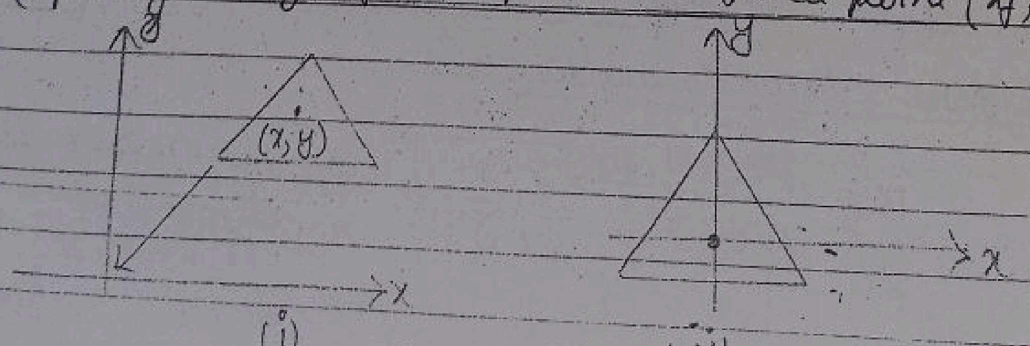
1. The fixed point along with the object is translated to the coordinate origin.
2. The object is rotated about origin by the specified angle.
3. The object is translated back to its original location

The composite matrix $(C_m) = T_{(x_f, y_f)} \cdot R_\theta \cdot T_{(-x_f, -y_f)}$ for this case.

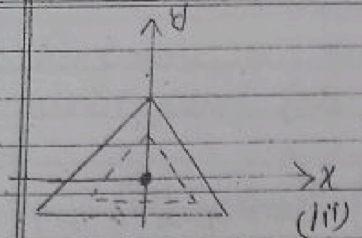


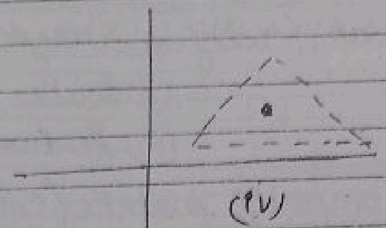
(i) (ii) (iii) (iv)

Steps:- (Scaling of an object about fixed point (x_f, y_f) (fixed point scaling):-



(i) (ii)





(i) The fixed point along with the object is translated to the coordinate origin.
(ii) The fixed point along with the object is scaled.

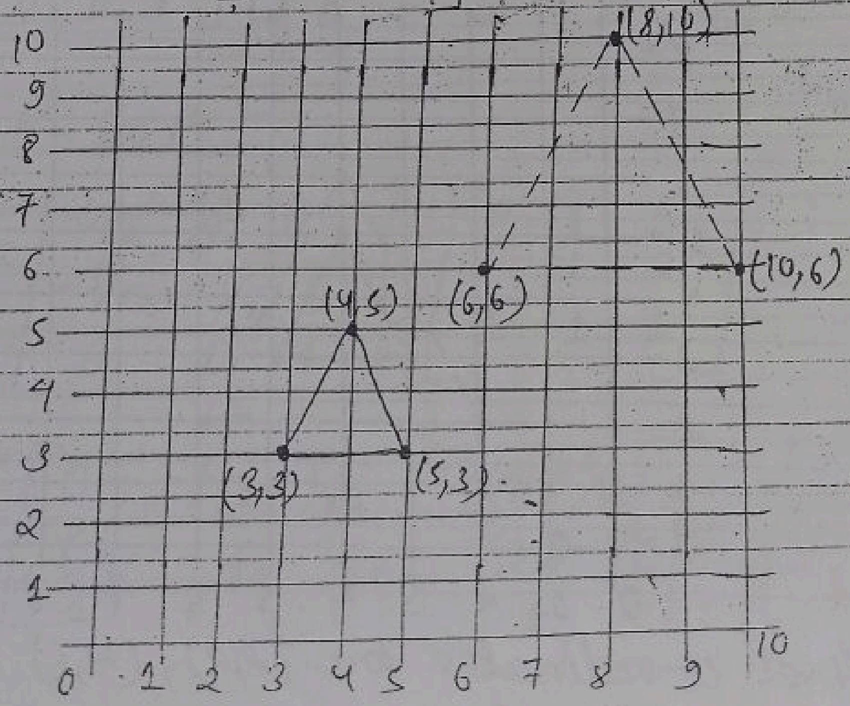
(iii) The fixed point along with the object is translated.
(iv) $C_m = T(x_f, y_f) \cdot S(s_x, s_y) \cdot T(-x_f, -y_f)$

Q: Enlarge an triangle A(3,3), B(4,5), C(5,3) by twice its original size about point.

soln

About origin; $P' = S.P$ $(s_x = s_y = 2)$

$$= \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 & 4 & 5 \\ 3 & 5 & 3 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 6 & 8 & 10 \\ 6 & 10 & 6 \\ 1 & 1 & 1 \end{bmatrix}$$



Date
Page

Next.

Q: About point (4,4):- Ans. (i) Translate
(ii) Scale
(iii) Re-translate

Composite matrix $(Cm) = T_{(4,4)}\ S_{(sx=sy=2)}\ T_{(-4,-4)}$

$$\Rightarrow C_m = \begin{bmatrix} 1 & 0 & 4 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -4 \\ 0 & 1 & -4 \\ 0 & 0 & 1 \end{bmatrix}$$

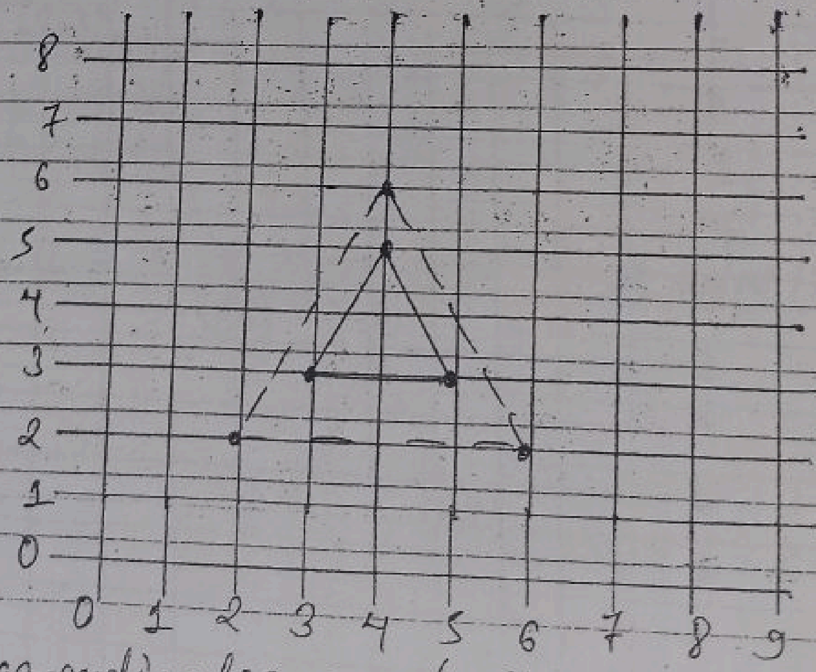
$$= \begin{bmatrix} 1 & 0 & 4 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & -8 \\ 0 & 2 & -8 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 & -4 \\ 0 & 2 & -4 \\ 0 & 0 & 1 \end{bmatrix}$$

Then,

$P' = Cm \cdot P$

$$= \begin{bmatrix} 2 & 0 & -4 \\ 0 & 2 & -4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 & 4 & 5 \\ 3 & 5 & 3 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 4 & 6 \\ 2 & 6 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

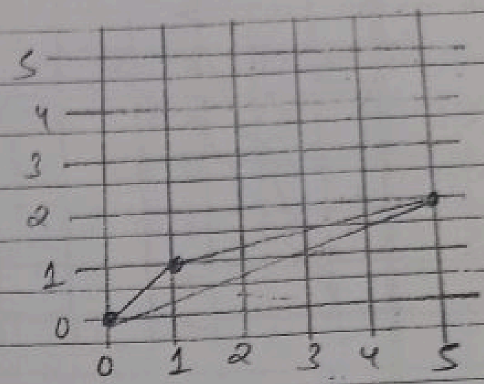
Notes:-
Way of Ans:-
(i) steps
(ii) Cm
(iii) P'
(iv) final point



∴ final co-ordinates are (2,2), (4,6) & (6,2).

Q: Perform a rotation of triangle A(0,0), B(1,1), C(5,+2) about point (-1,-1).

Ans: $C_m = T_{(-1,-1)} R_\theta T_{(1,1)}$



$$C_m = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

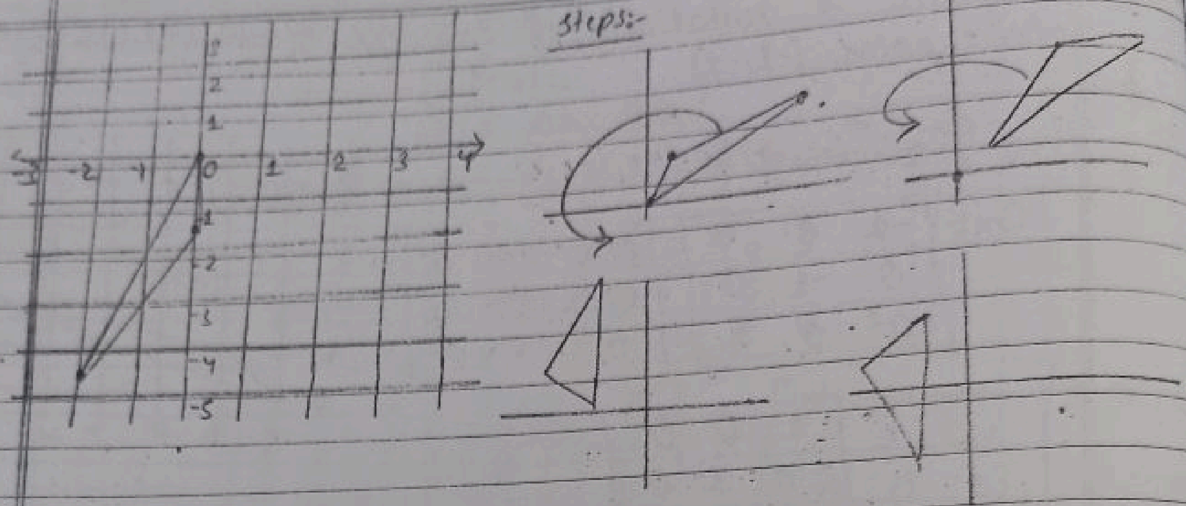
$$= \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Then, $P' = C_m \cdot P$

$$= \begin{bmatrix} -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -1/\sqrt{2} + 1/\sqrt{2} & -5/\sqrt{2} + 2/\sqrt{2} \\ 0 & -1/\sqrt{2} - 1/\sqrt{2} & -5/\sqrt{2} - 2/\sqrt{2} \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -2.121 \\ 0 & -1.414 & -4.9497 \\ 1 & 1 & 1 \end{bmatrix}$$

Steps:-



Q: Compute the composite matrix to scale the y-coordinate of a point to make the image twice as tall, shift it down by one unit and then rotate clock-wise by 30°. Compute the new coordinates of point (3, -4) after these transformations:-

Soln

Steps:-

(i) Scale the y-coordinate by twice its original size

(ii) Translate in y-direction by -1 unit.

(iii) Rotate in clock-wise direction by 30°.

$$C_m = R_\theta T_{(0,-1)} S_{(1,2)}$$

$$= \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos 30 & \sin 30 & 0 \\ -\sin 30 & \cos 30 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & -1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \sqrt{3}/2 & 2 * 1/2 & -1/2 \\ -1/2 & 2 * \sqrt{3}/2 & -\sqrt{3}/2 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \sqrt{3}/2 & 1 & -1/2 \\ -1/2 & \sqrt{3} & -\sqrt{3}/2 \\ 0 & 0 & 1 \end{bmatrix}$$

Then,

$P' = Cm.P$

$$= \begin{bmatrix} \sqrt{3}/2 & 1 & -1/2 \\ -1/2 & \sqrt{3} & -\sqrt{3}/2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} +3 \\ -4 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{3\sqrt{3}}{2} - 4 - \frac{1}{2} & \frac{-3}{2} - 4\sqrt{3} - \frac{\sqrt{3}}{2} & 1 \end{bmatrix}$$

$$= \begin{bmatrix} -1.90 & -9.29 & 1 \end{bmatrix}$$

Q. A triangle A(1,2) B(2,3) C(3,2) is required to be rotated by 45° in clock-wise direction about point (2,3) then translated in x-direction by 4 units. Find the final coordinates of the triangle.

s/n

Steps:-

(i) Translate the point (2,3) to origin.

(ii) Rotate the triangle by angle 45° about origin.

(iii) ~~Re~~ Translate the triangle into previous point.

(iv) Translate the triangle by (2,3).

$$C_m = T_{(6,3)}\, T_{(2,3)}\, R_\theta\, T_{(-2,-3)}$$

$$= \begin{bmatrix} 1 & 0 & 6 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45^\circ & \sin 45^\circ & 0 \\ -\sin 45^\circ & \cos 45^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -2 \\ 0 & 1 & -3 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 6 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45^\circ & \sin 45^\circ & -2\cos 45^\circ - 3\sin 45^\circ \\ -\sin 45^\circ & \cos 45^\circ & +2\sin 45^\circ - 3\cos 45^\circ \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 6 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & (-2 * 1/\sqrt{2} - 3 * 1/\sqrt{2}) + 2 \\ -\sin 45^\circ & 0 & 2\sin 45^\circ - 3\cos 45^\circ \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 6 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & -1.535 \\ -1/\sqrt{2} & 0 & -0.7071 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.7071 & 0.7071 & 4.465 \\ -0.7071 & 0 & -1.1213 \\ 0 & 0 & 1 \end{bmatrix}$$

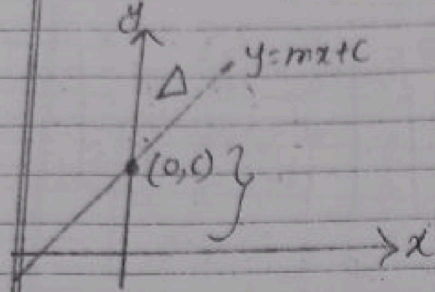
Now,

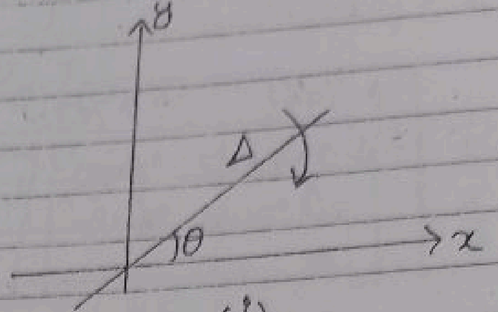
$P' = Cm \cdot P$

$$= \begin{bmatrix} 0.7071 & 0.7071 & 4.465 \\ -0.7071 & 0 & -1.1213 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

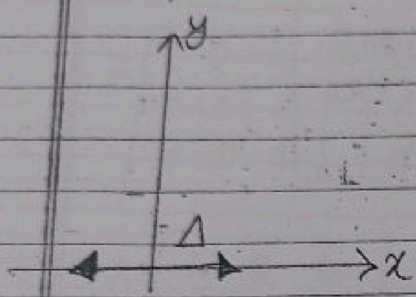
$$= \begin{bmatrix} 6.5863 & 8.0005 & 8.0005 \\ -1.8284 & -2.5355 & -3.2426 \\ 1 & 1 & 1 \end{bmatrix}$$

Reflection of an object about arbitrary line $y=mx+c$

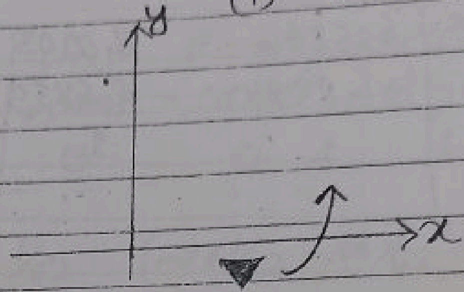




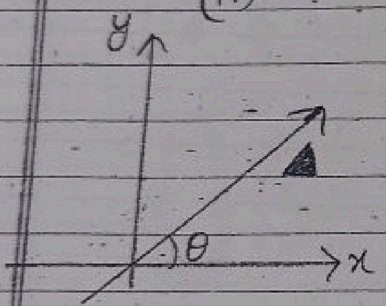
(i)



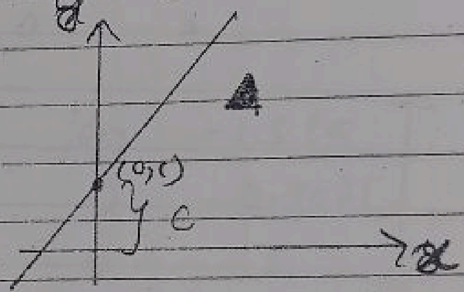
(ii)



(iii)



(iv)



(v)

$$C_m = T_{(0,c)}\, R'_{\theta}\, R_{f_x}\, R_{\theta}\, T_{(0,-c)}$$

Translate → Rotate → Reflect → Rotate → Translate

if $(\theta - 90^\circ)$ then y-axis

Steps:-

(i) Translate the line (along with object) by $t_x = 0, t_y = -c$ as 'c' is the y-intercept at point $(0,c)$ so that line passes through origin.

(ii) Rotate by the desired angle 'θ' where $\tan\theta = m$ or $\theta = \tan^{-1} m$ about origin.

(iii) Take reflection of object about X or Y-axis depending on the direction of rotation.

(iv) Again rotate in reverse direction.

(v) Retranslate by $t_x = 0$; $t_y = c$

Then $C_m = T'_{(0,c)} R'_\theta R_f R_\theta T_{(0,-c)}$

Imp for test

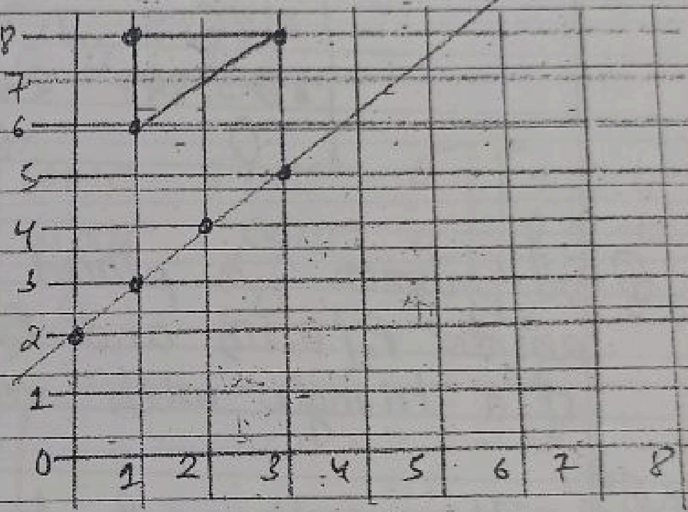
Q. Reflect a triangle A(1,8) B(3,8) C(1,6) about line $y = x+2$.

Soln

$y = x+2$

so;

x	y
0	2
1	3
2	4
3	5



$\theta = \tan^{-1} m$

$= \tan^{-1} 1 = 45^\circ$

Then,

$C_m = T'_{(0,2)} R'_{45^\circ} R_{fx} R_{45^\circ} T_{(0,-2)}$

anti-clockwise ; clock-wise

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45^\circ & -\sin 45^\circ & 0 \\ \sin 45^\circ & \cos 45^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45^\circ & \sin 45^\circ & 0 \\ -\sin 45^\circ & \cos 45^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 1 \\ 0.7071 & -0.7071 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45^\circ & \sin 45^\circ & 0 \\ -\sin 45^\circ & \cos 45^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.7071 & 0.7071 & -0.4547 \\ 0.999 & -7\times10^{-5} & 1.9998 \\ 0 & 0 & 1 \end{bmatrix}$$

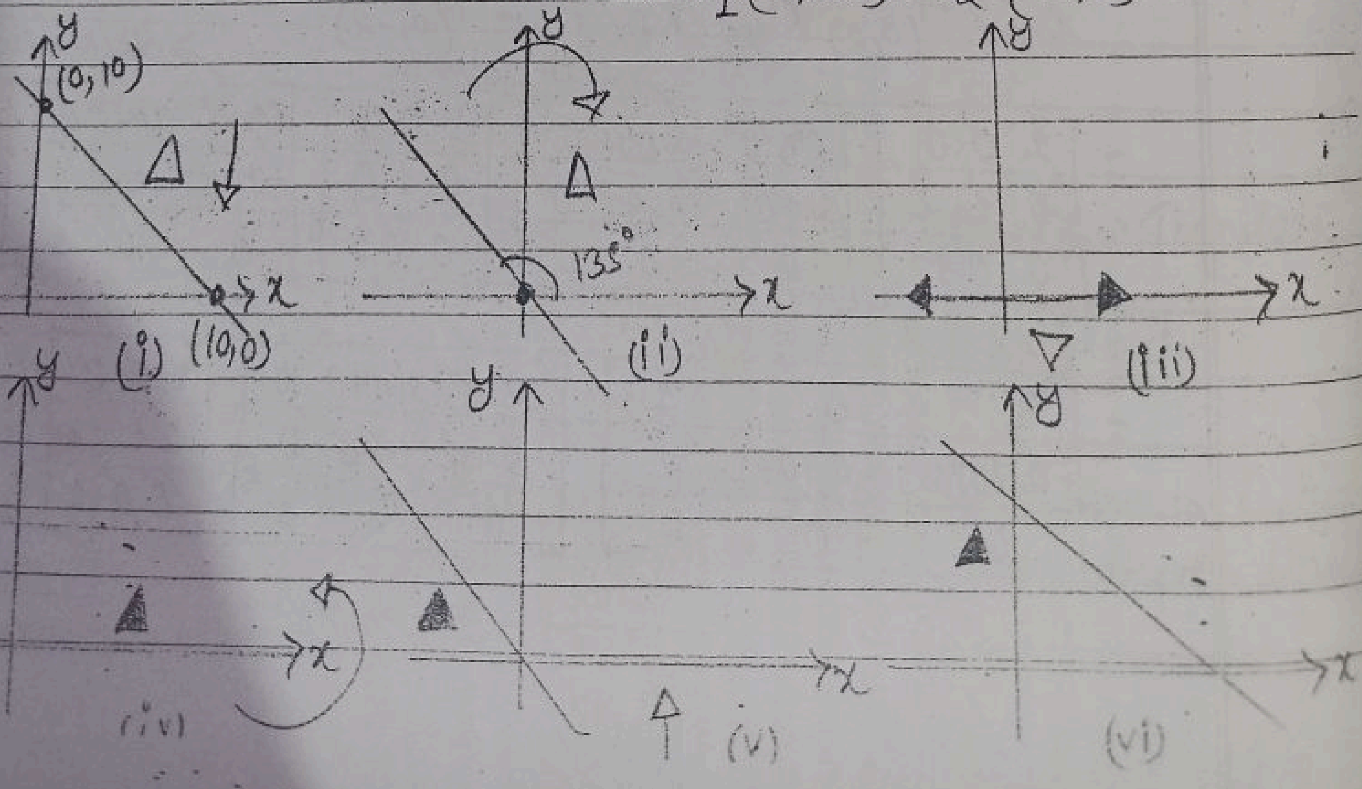
Now,

$$P' = \begin{bmatrix} 0.7071 & 0.7071 & -0.4547 \\ 0.999 & -7\times10^{-5} & 1.9998 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 3 & 1 \\ 8 & 8 & 6 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 6.3639 & 7.7781 & 4.495 \\ 2.99844 & 2.99644 & 2.99838 \\ 0 & 0 & 1 \end{bmatrix}$$

Q. A mirror is placed such that it passes through points $P_1(0,10)$ and $P_2(10,0)$, find the reflected view of a triangle with vertices $A(5,50)$, $B(20,40)$ $C(10,70)$

Soln The vertices of triangles $A(5,50)$ $B(20,40)$ $C(10,70)$
$P_1(0,10)$ $P_2(10,0)$



Translate → Rotate → Reflect → Rotate → Translate

$C_m = T'_{(0,10)} R'_\theta R_{fx} R_\theta T_{(0,-10)}$

$$C_m = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 10 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 135^\circ & -\sin 135^\circ & 0 \\ \sin 135^\circ & \cos 135^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos 135^\circ & \sin 135^\circ & 0 \\ -\sin 135^\circ & \cos 135^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -10 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} -0.7071 & 0.7071 & 0 \\ 0.7071 & 0.7071 & 10 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 135^\circ & \sin 135^\circ & 0 \\ -\sin 135^\circ & \cos 135^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -10 \\ 0 & 0 & 1 \end{bmatrix}$$

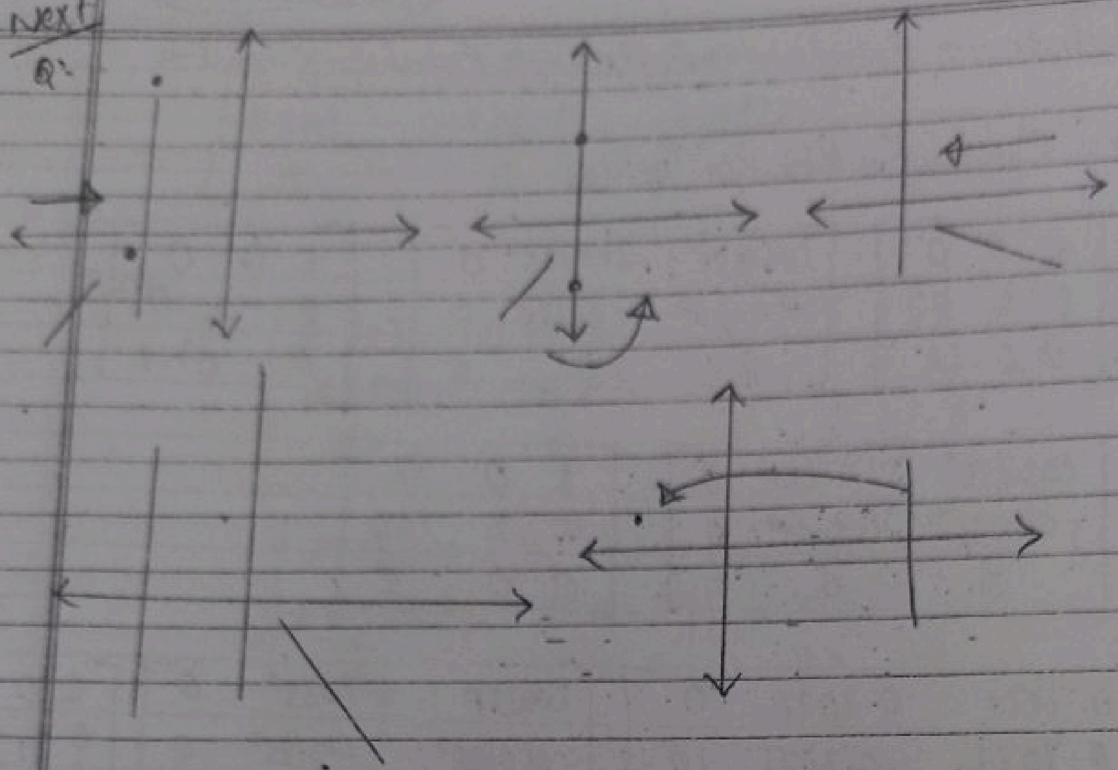
$$= \begin{bmatrix} 0 & -0.999 & 9.9999 \\ -0.999 & 0 & 10 \\ 0 & 0 & 1 \end{bmatrix}$$

$P' = C_m . P$

$$= \begin{bmatrix} 0 & -0.999 & 9.9999 \\ -0.999 & 0 & 10 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 5 & 20 & 10 \\ 50 & 40 & 70 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} -49.995 & -39.96 & -59.9301 \\ -4.995 & -19.98 & 0.01 \\ 0 & 0 & 1 \end{bmatrix}$$

Next Q:



$$C_m = T_{(-2,0)}\, R_{fy}\, T_{(2,0)}$$

$$= \begin{bmatrix} 1 & 0 & -2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} -1 & 0 & -2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & -4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P' = C_m . P = \begin{bmatrix} -1 & 0 & -4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -3 & -1 \\ -4 & -3 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 3 & 1 \\ -4 & -3 \\ 0 & 0 \end{bmatrix}$$

Composite Matrix:- Using matrix representation it is possible to setup a matrix for any sequence of transformations as a composite transformations by calculating the product of individual transformation.

For column matrix representation of coordinate positions we form composite transformation by multiplying matrices inorder from right to left.

1: Two successive translations are additive:-

Let 2 successive translations vectors be (tx_1, ty_1) (tx_2, ty_2) applied to a coordinate position P

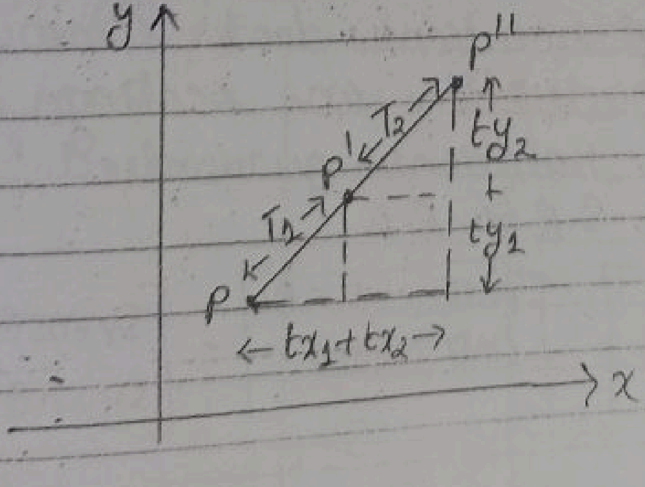
Then; $P' = T(tx_2, ty_2) \cdot \{T(tx_1, ty_1) \cdot P\}$

$$= \{T(tx_2, ty_2) \cdot T(tx_1, ty_1)\} P$$

here, the composite transformation matrix for this sequence of transformation is:-

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

$$\Rightarrow T(tx_2, ty_2) \cdot T(tx_1, ty_1) = T(tx_1 + tx_2, ty_1 + ty_2)$$



2: Two successive rotations are Additive:- Let 2 successive rotation vectors $R_{\theta_1}, R_{\theta_2}$ be applied to a coordinate position P then;

$$P' = \cancel{R\theta}\, R_{\theta_2} \cdot \{R_{\theta_1} \cdot P\}$$
$$= \{R_{\theta_2} \cdot R_{\theta_1}\} \cdot P$$

here, the composite transformation matrix for this sequence of transformation is:-

$$\Rightarrow \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} \cos\theta_2 \cdot \cos\theta_1 - \sin\theta_2 \cdot \sin\theta_1 & -\cos\theta_2 \cdot \sin\theta_1 - \sin\theta_2 \cdot \cos\theta_1 & 0 \\ \sin\theta_2 \cdot \cos\theta_1 + \cos\theta_2 \cdot \sin\theta_1 & -\sin\theta_2 \cdot \sin\theta_1 + \cos\theta_2 \cdot \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

$$\Rightarrow R(\theta_2) \cdot R(\theta_1) = R(\theta_1+\theta_2)$$

Q: 2D scales and rotations don't commute in general, under what restriction on scaling is commutation of scales and rotations guaranteed?

$\Rightarrow$

$$S \cdot R \neq R \cdot S.$$

$$S \cdot R = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} S_x\cos\theta & -S_x\sin\theta \\ S_y\sin\theta & S_y\cos\theta \end{bmatrix}$$

$$R.S. = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} = \begin{bmatrix} \cos\theta S_x & -\sin\theta S_y \\ S_x \sin\theta & \cos\theta S_y \end{bmatrix}$$

True incase of uniform scaling i.e: $S_x = S_y$

Q: Scaling followed by rotation is equivalent to shearing (bi-directional):-

Ans>

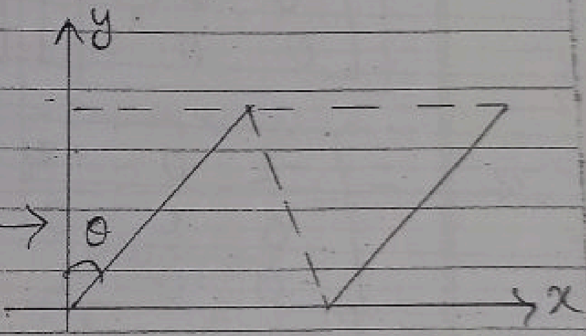
$$R.S. = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \approx \text{Shearing} = \begin{bmatrix} 1 & Sh_x \\ Sh_y & 1 \end{bmatrix}$$

$$= \begin{bmatrix} S_x\cos\theta & -S_y\sin\theta \\ S_x\sin\theta & S_y\cos\theta \end{bmatrix} \approx \begin{bmatrix} 1 & Sh_x \\ Sh_y & 1 \end{bmatrix}$$

So,

$S_x\cos\theta = 1$ $\quad$ $-S_y\sin\theta = Sh_x$

$S_x\sin\theta = Sh_y$ $\quad$ $S_y\cos\theta = 1$



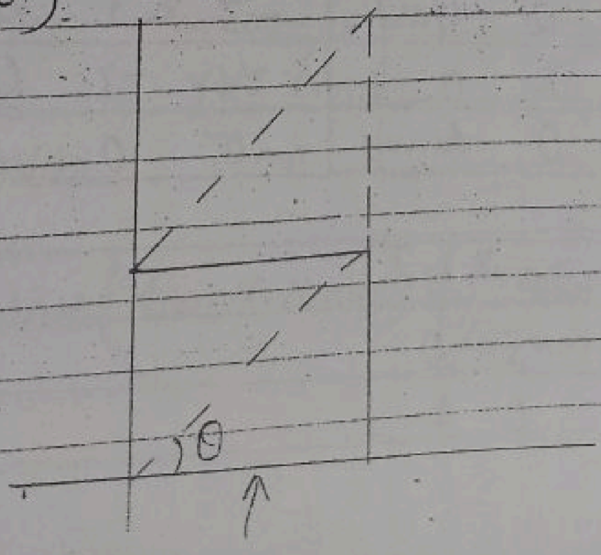
So, $S_x = \dfrac{1}{\cos\theta}$, $S_y = \dfrac{1}{\cos\theta}$

$Sh_x = \dfrac{x'-x}{y} = -\tan\theta$

$Sh_x = \left(-\dfrac{1}{\cos\theta}\right) * \sin\theta = -\tan\theta$

$Sh_y = \tan\theta$

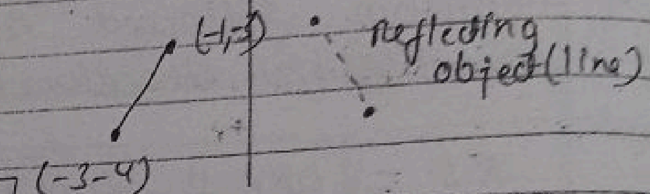
$Sh_y = \tan\theta$



$Sh_y = \dfrac{y'-y}{x} = \tan\theta$

Q: Reflect a line with end points $A(-3,-4)$ $B(-1,-3)$ about line $x=-2$;

Hint $C_m = T_{(-2,0)} \cdot R_{fy} T_{(2,0)}$



$$= \begin{bmatrix} 1 & 0 & -2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} -1 & 0 & -2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} -1 & 0 & -4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Again

$\Rightarrow P' = C_m \cdot P$

$$= \begin{bmatrix} -1 & 0 & -4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -3 & -1 & 0 \\ -4 & -3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

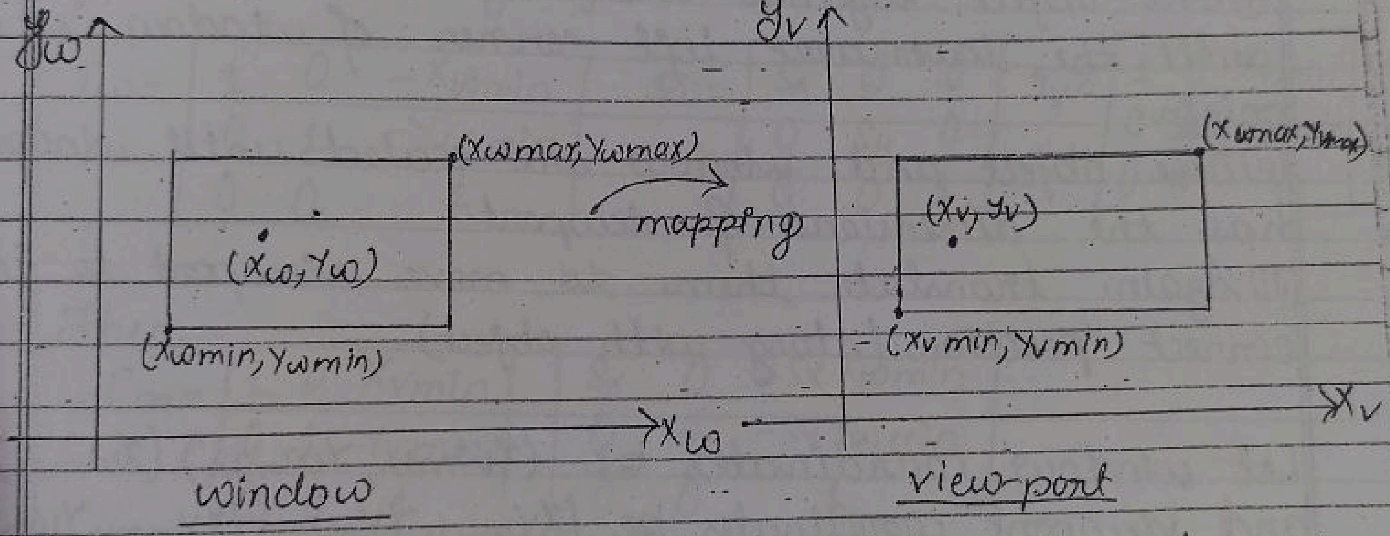
$$= \begin{bmatrix} 3 & 1 & -4 \\ -4 & -3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Window to View-port Mapping:-

Window:- a rectangular area specified in world coordinate system (defines what to view).

Viewport:- a rectangular area on the display device to which a window is mapped (defines where it is to be displayed)

· The mapping of a part of a world coordinate scene to device coordinate is called viewing transformation.



To maintain relative placement in viewport as in window;

$$\Rightarrow \frac{X_w - X_{wmin}}{X_{wmax} - X_{wmin}} = \frac{X_v - X_{vmin}}{X_{vmax} - X_{vmin}}$$

$$\Rightarrow \frac{Y_w - Y_{wmin}}{Y_{wmax} - Y_{wmin}} = \frac{Y_v - Y_{vmin}}{Y_{vmax} - Y_{vmin}}$$

Solving;

$$X_v = X_{vmin} + (X_w - X_{wmin}) \left(\frac{X_{vmax} - X_{vmin}}{X_{wmax} - X_{wmin}}\right)$$

Date
Page

$$Y_V = Y_{vmin} + (Y_w - Y_{wmin})\left(\frac{Y_{vmax} - Y_{vmin}}{Y_{wmax} - Y_{wmin}}\right)$$

$\Rightarrow X_V = X_{Vmin} + (X_w - X_{wmin})\, S_x$

$\Rightarrow Y_V = Y_{Vmin} + (Y_w - Y_{wmin})\, S_Y$;

where S_x, S_y are scaling factors.

Alternative Approach:-

* To transform a window to view-port:-

(i) The object together with its window is translated until the lower-most left corner of window is at origin.

(ii) The object and window are scaled until window has the dimensions of viewport.

(iii) Again translate them to move viewport to its correct position (along with object).

Let window coordinates be (X_{wmin}, Y_{wmin}) (X_{wmax}, Y_{wmax}) and viewport coordinates be (X_{Vmin}, Y_{Vmin}) (X_{Vmax}, Y_{Vmax})

The viewing transformation are:-

(i) Translate window to origin by;

$$T_x = -X_{wmin} \text{ and } T_Y = -Y_{wmin}$$

(ii) Scale window to the size of viewport;

$$S_x = \frac{(X_{Vmax} - X_{Vmin})}{(X_{wmax} - X_{wmin})} \qquad S_Y = \frac{(Y_{Vmax} - Y_{Vmin})}{(Y_{wmax} - Y_{wmin})}$$

(iii) Translate by

$T_x = X_{vmin}$ & $T_y = Y_{vmin}$

All these steps can be represented by the following composite transformation:-

$C_m = T_w * S_{wv} * T_v$

where, C_m = Composite Transformation (here, viewing transformation)

T_w = Translate window to origin

T_v = Translate viewport to origin

S_{wv} = Scaling of window to viewport

$$T_w = \begin{bmatrix} 1 & 0 & -X_{wmin} \\ 0 & 1 & -Y_{wmin} \\ 0 & 0 & 1 \end{bmatrix} \quad S_w = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad T_v = \begin{bmatrix} 1 & 0 & X_{vmin} \\ 0 & 1 & Y_{vmin} \\ 0 & 0 & 1 \end{bmatrix}$$

Then,

$$C_m = \begin{bmatrix} 1 & 0 & X_{vmin} \\ 0 & 1 & Y_{vmin} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S_x & 0 & -S_x X_{wmin} \\ 0 & S_y & -S_y Y_{wmin} \\ 0 & 0 & 1 \end{bmatrix}$$

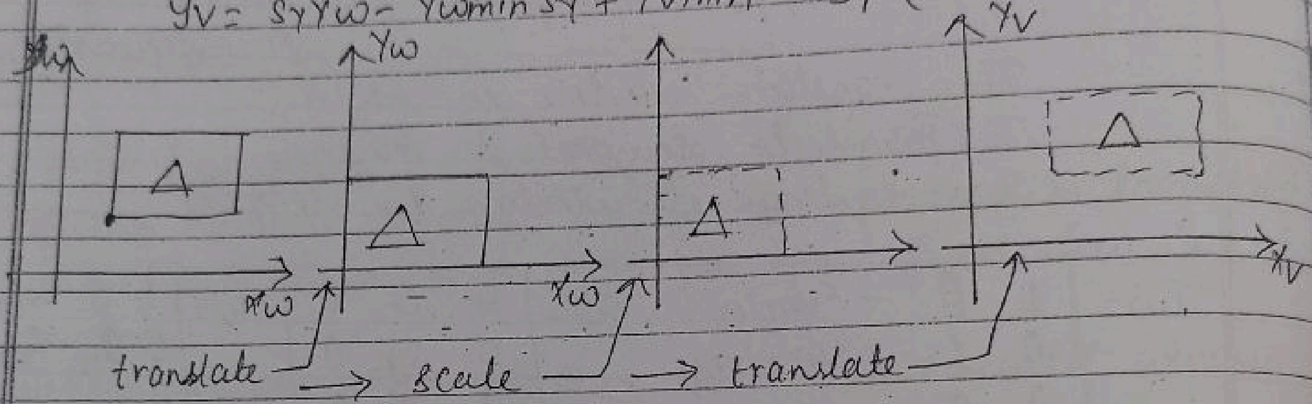
$$= \begin{bmatrix} S_x & 0 & -S_x X_{wmin} + X_{vmin} \\ 0 & S_y & -S_y Y_{wmin} + Y_{vmin} \\ 0 & 0 & 1 \end{bmatrix}$$

viewport to window:-

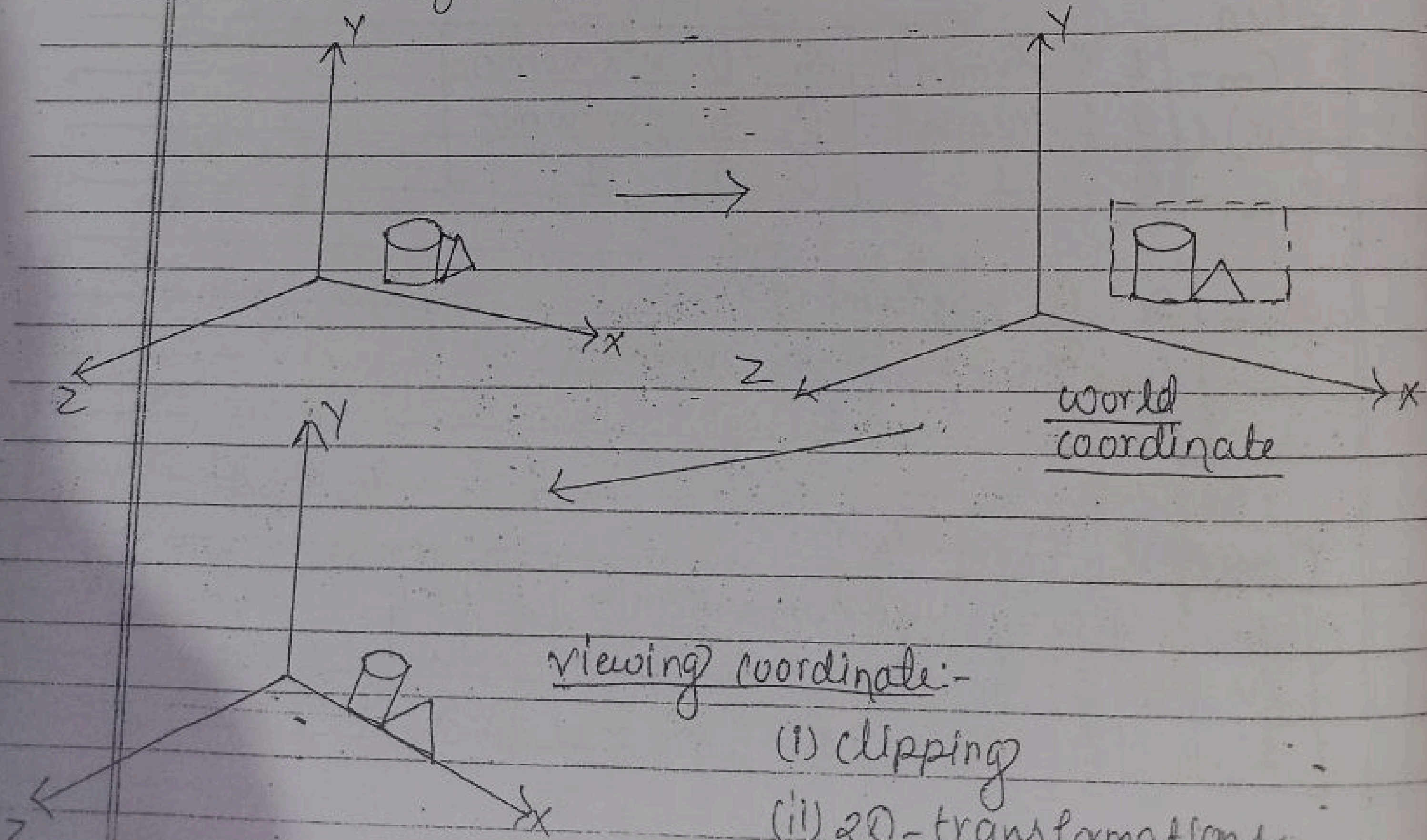
$$\begin{bmatrix} X_v \\ Y_v \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & -S_x . X_{wmin} + X_{vmin} \\ 0 & S_y & -S_y . Y_{wmin} + Y_{vmin} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x_v \\ y_v \\ 1 \end{bmatrix} = \begin{bmatrix} S_x X_w - S_x X_{wmin} + X_{vmin} \\ S_y Y_w - S_y Y_{wmin} + Y_{vmin} \\ 1 \end{bmatrix}$$

$$\therefore X_v = S_x X_w - X_{wmin} S_x + X_{vmin} = S_x (X_w - X_{wmin}) + X_{vmin}$$
$$Y_v = S_y Y_w - Y_{wmin} S_y + Y_{vmin} = S_y (Y_w - Y_{wmin}) + Y_{vmin}$$



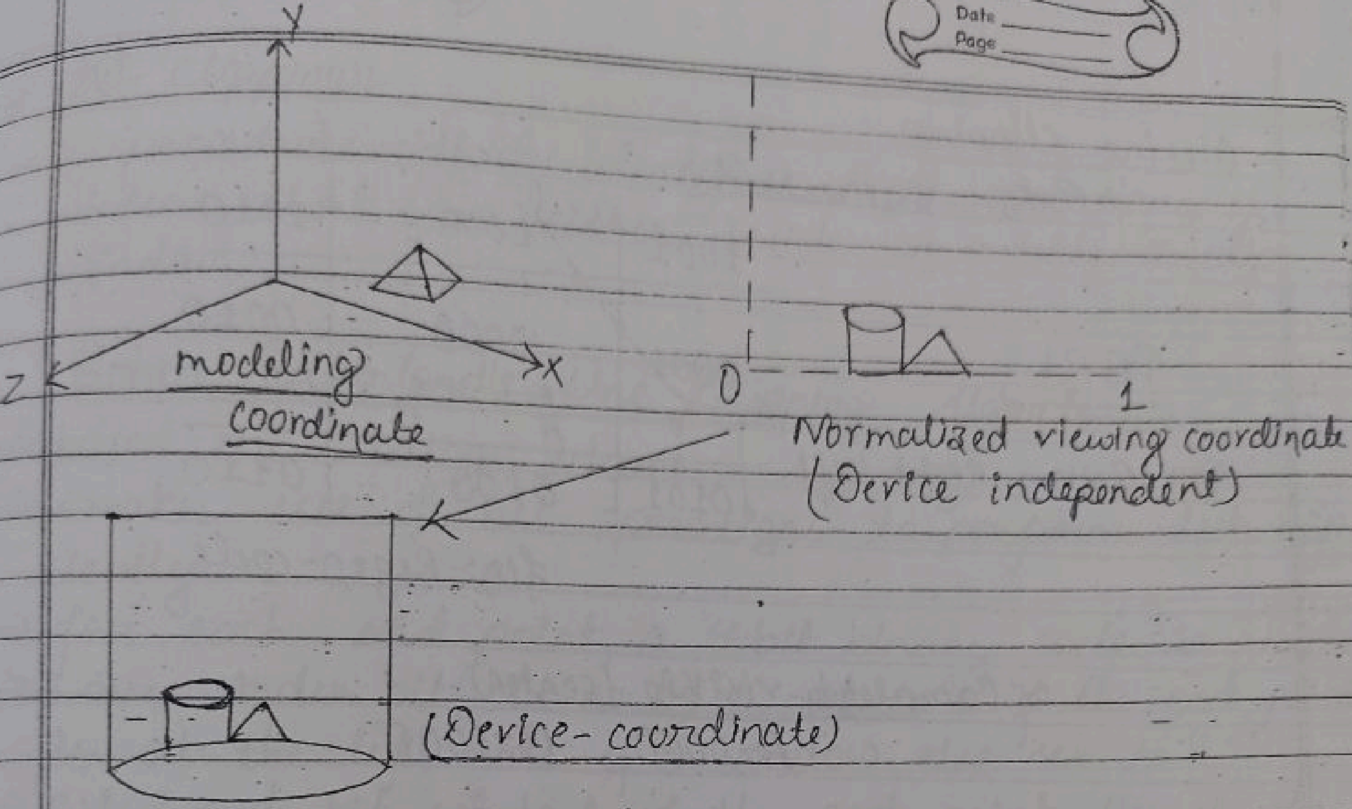
2D- Viewing Pipe-line



viewing coordinate:-

(i) clipping

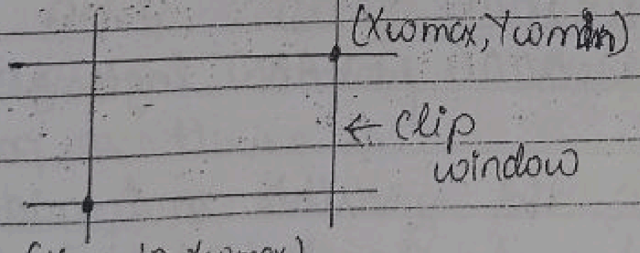
(ii) 2D-transformations



2D Clipping:-

(i) Point Clipping
(ii) Line Clipping
(iii) Polygon Clipping

(i) Point Clipping



$X_{wmin} \leq x \leq X_{wmax}$

$Y_{wmin} \leq y \leq Y_{wmax}$

(ii) Line Clipping

* Cohen-Sutherland :-

TBRL

↓

Top Bottom Right Left

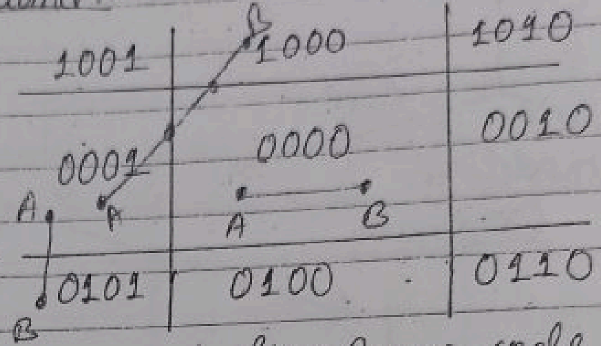


fig:- Regon-code

a: Completely-visible (center)

A=0000   B=0000

b: Completely invisible (up-down)

A=0001 > same bit

B=0101

AND =0001 (∵ presence of 0001 as a result presence of '1' so invisible)

c: Partially visible (R-L)

A=0001

B=1000

AND 0000 (∵ AND logic gives 0000 as result so, partially visible and mid-point is need to be find)

2D- Clipping:-

→ a procedure that identifies those operations of picture that are either inside or outside a clip-window.

<u>Cohen Sutherland's Line Clipping Algorithm:-</u>

→ based on coding scheme.

→ makes use of bit operations to perform test efficiently.

→ for each end point, a 4-bit binary code is used.

→ Lower order bit (bit no. 1) is set to "1" if the end point is at the left side of a window else set to "0".

→ Bit 2, next bit set to 1 if the end point lies at the right side.

→ Bit 3, set to 1 if point lies at the bottom of window

→ Higher order bit (bit 4) set to 1 if point lies above the window.

By numbering bit positions in region code as 1-4 from R-L coordinate regions can be co-related with bit positions as:-

Bit 4	Bit 3	Bit 2	Bit 1
Top	Bottom	Right	Left

Value of 1 in any bit position indicates that the point is in that relative position else bit position is set to 0.

1001	1000	1010
0001	0000	0010
0101	0100	0110

The algorithm is as follows:-

Step 1: Establish region codes for all line endpoints

Bit 1 is set to "1" if $X < X_{wmin}$ else set to "0"

Bit 2 is set to "1" if $X > X_{wmax}$ else set to "0"

Bit 3 is set to "1" if $Y < Y_{wmin}$ else set to "0"

Bit 4 is set to "1" if $Y > Y_{wmax}$ else set to "0"

Step 2: Determine which lines are completely inside or completely outside the window using following tests:-

(a) if both end points of line have region code "0000" line is completely inside window.

(b) if logical AND operation of region code of two end-points is not "0000" then line is completely outside (same-bit position of two end-points having "1" line is completely outside)

Step 3: If both the tests in step 2 failed then the line is not completely inside nor completely outside. So, we need to find the intersection of end points with boundaries of window

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

(a) if bit 1 is "1" then line intersects with left boundary so, $y_i = y_1 + m(x - x_1)$

where $x = x_{wmin}$

(b) if bit 2 is "1" then line intersects with right boundary so, $y_i = y_1 + m(x - x_1)$

where $x = x_{wmax}$

ⓒ if bit 3 is "1" then line intersects with the bottom boundary so, $x_i = x_1 + \frac{1}{m}(y - y_1)$
where $y = y_{wmin}$

ⓓ if bit 4 is "1" then line intersects with the top boundary so, $x_i = x_1 + \frac{1}{m}(y - y_1)$
where $y = y_{wmax}$

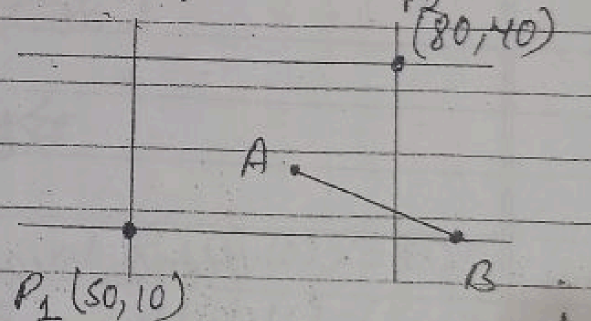
where x_i and y_i are the x and y intercepts for that line.

Step 4: Repeat step 1 to step 3 until line is completely accepted or rejected.

Q: Use Cohen Sutherland's line clipping algorithm to clip a line with end points A(70,20), B(100,10) against a clip window whose lowermost left corner is at $P_1(50,10)$ and uppermost at $P_2(80,40)$.

Sol^n

Step 1: Establish region codes for end points A and B



Region code for A A(70,20)	Region code for B B(100,10)
70 !< 50 bit1 = 0	100 !< 50 bit 1 = 0
70 !> 80 bit2 = 0	100 > 80 bit 2 = 1
20 !< 10 bit3 = 0	10 !< 10 bit 3 = 0
20 !> 40 bit4 = 0	10 !> 40 bit 4 = 0

Step 2: Perform logical AND operations of region codes of 2 end points;

$$A = 0000$$
$$B = 0100$$
$$\overline{AND = 0000}$$
(partial visibility)

Step 3: Find intersections with boundaries. The end-point B intersects with right boundary.

$y_i = y_1 + m(\cancel{x - x_1})$
$= 10 + \cancel{m}(80 - \cancel{70})$
$= \cancel{10 +}(0.3) \cancel{*} 10$

$y_i = y_1 + m(x - x_1) \ldots$ ①

where $x = 80$

$B(x_2, y_2) = (100, 10)$

$A(x_1, y_1) = (70, 20)$

$\therefore m = \dfrac{y_2 - y_1}{x_2 - x_1} = -0.33$

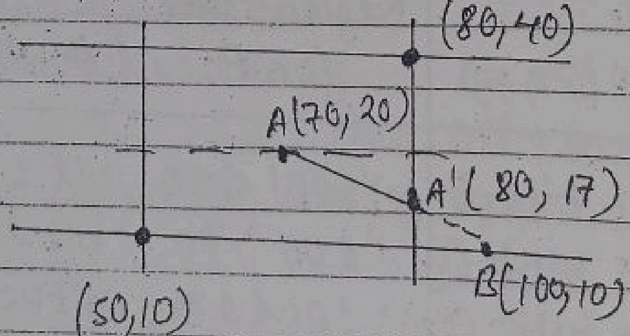
Then, in eq^n ①,

$y_i = 10 + (-0.33)(80 - 100)$

$\because$ at right boundary locates $B(100, 10)$ as (x_1, y_1)

$\Rightarrow y_i = 10 - (0.33 * -20)$
$= 16.6 = 17$

Hence, required $(x_i, y_i) = (80, 17)$



Date ______
Page ______

Q. A(30,15) B(70,30), P_1(50,10) P_2(80,40)

soln

Step1:

Region code for A	Region code for B
30 < 50 bit 1 = 0	70 !< 50 bit 1 = 0
30 !> 80 bit 2 = 0	70 !> 80 bit 2 = 1
15 !< 10 bit 3 = 0	30 !< 10 bit 3 = 0
15 !> 40 bit 4 = 1	30 !> 100 bit 4 = 1

Step 2:

A = 0001
B = 0101
――――
0001
(completely invisible)

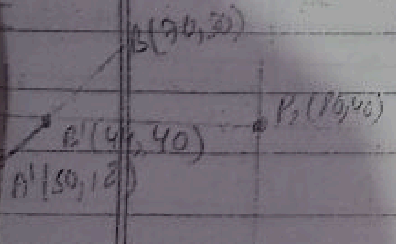
Step3: $m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{30-15}{70-30} = 0.375$

Then, $y_i = y_1 + (x - x_1)m$ at P_1(50,10); x = 50

$= 15 + (50-30)0.375$ A(30,15) = (x_1, y_1)

$= 22.5$

$x_i = x_2 + \frac{1}{m}(y - y_2)$



$= 70 + \frac{1}{0.37}(40-30)$ at P_2(80,40) y = 40

B(70,30) = (x_2, y_2)

$= 97.02$

Date ____
Page ____

Q: Clip a line with end points A (-13, 5) B(17, 11) against a clip window with lowermost left corner at (-8, -4) & uppermost right corner at (12, 8)

Solⁿ A (-13, 5)
B(17, 11)

Step 1:

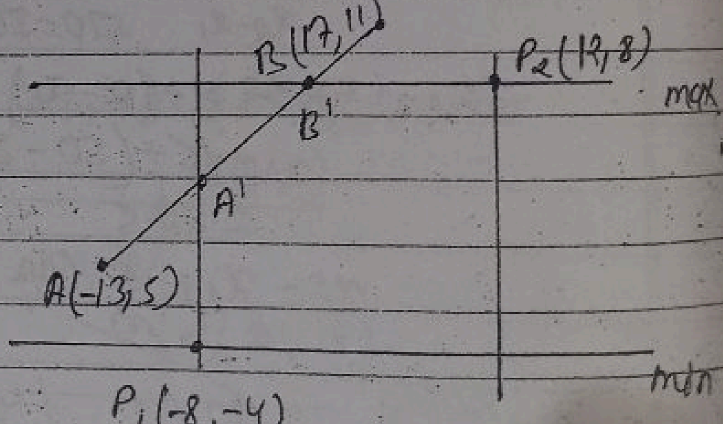
Region (A)	Region (B)
-13 < -8 bit1 = 1	17 !< -4 bit1 = 0
-13 !> 12 bit2 = 0	17 > 8 bit2 = 1
5 !< -4 bit3 = 0	11 !< -4 bit3 = 0
5 !> 8 bit4 = 0	11 > 8 bit4 = 1

Step 2:

A: 1000
B: 0101

0000 partially visible

Step 3:



$y_i = y_1 + m(x_2 - x_1)$

$= 5 + 0.2(-8 + 13)$

$= 6$

$m = \frac{11 - 5}{17 + 13} = 0.2$

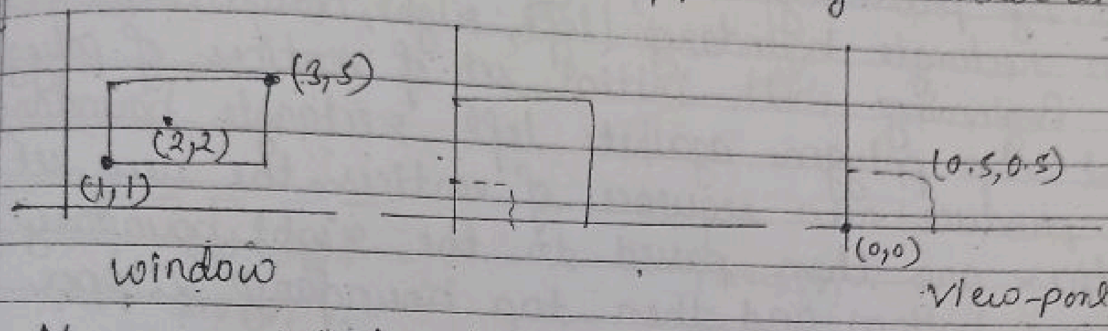
$x_i = x + \frac{1}{m}(y - y_1)$

$= 17 + \frac{1}{0.2}(8 - 11)$

$= 2$

Q. Find the normalization transformation window to viewport with windows lowermost left corner at (1,1) & uppermost right corner at (3,5) onto viewport having lower left corner at (0,0) upper right corner at (0.5,0.5).

Sol^n.



Map a point (2,2) window to view-port.

$C_m = S_{w\ to\ v}\ T_{(-1,-1)}$

$P' = C_m \cdot P$

$$\begin{bmatrix} x_v \\ y_v \\ 1 \end{bmatrix} = C_m \cdot \begin{bmatrix} x_w = 2 \\ y_w = 2 \\ 1 \end{bmatrix}$$

Q: Find window to viewport transformation matrix with window of circular shape having center at (w_{xc}, w_{yc}) & radii as w_r viewport of circular shape with center at (V_{xc}, V_{yc}) and radii as V_r.

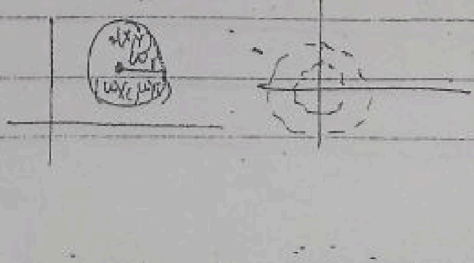
Hint:

$$T_w = \begin{bmatrix} 1 & 0 & -w_{xc} \\ 0 & 1 & -w_{yc} \\ 0 & 0 & 0 \end{bmatrix} \quad T_v = \begin{bmatrix} 1 & 0 & V_{xc} \\ 0 & 1 & V_{yc} \\ 0 & 0 & 1 \end{bmatrix}$$

$C_m = T_v \cdot S_{wv} \cdot T_w$

$P' = C_m \cdot P$

$$S_{wv} = \begin{bmatrix} V_r/w_r & 0 & 0 \\ 0 & V_r/w_r & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



where, $S_x = V_r/w_r$ $S_y = V_r/w_r$

Polygon Clipping: Sutherland- Hodgeman

A polygon can be clipped by processing polygon boundary as a whole against a clip window edge by processing all polygon vertices against each clip rectangle boundary (left, right, bottom, top) in turn.

Beginning with initial set of vertices of polygon, first clip polygon against left rectangle boundary to produce new sequence of vertices, the new set of vertices are then passed to the right boundary clipper, bottom and then top boundary clipper.

Four possible cases arise:-

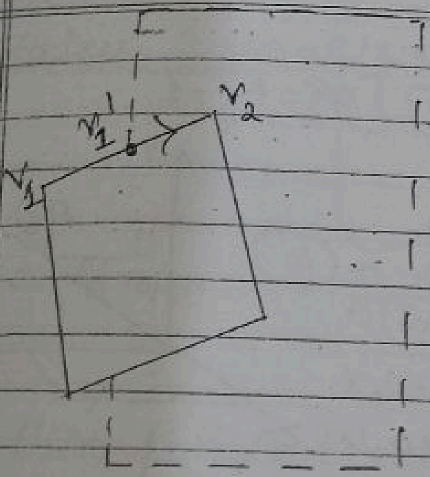
(i) if 1st vertex is outside window boundary & 2nd vertex is inside, both the intersection point of polygon edge with window boundary & second vertex are added to output vertex list.

(ii) If both input vertices are inside window boundary, only second vertices is added to list.

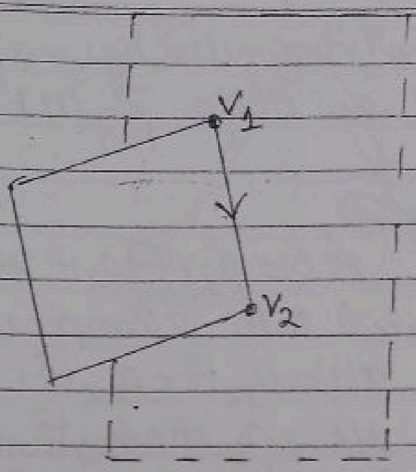
(iii) if 1st ~~i/p~~ input vertex is inside and 2nd vertex is outside, only the edge intersection with window boundary is added to vertex list.

(iv) if both input vectors are outside then nothing is added to the list.

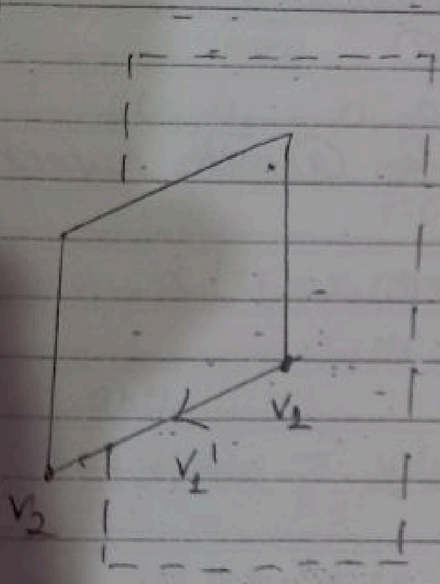
Once all vertices have been processed for one clip window boundary the output list of vertices is clipped with next window boundary.



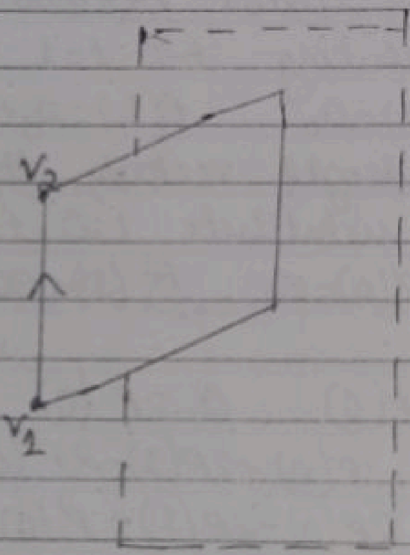
out → in
save $v_1' v_2$



in → in
save v_2



in → out
save v_1'

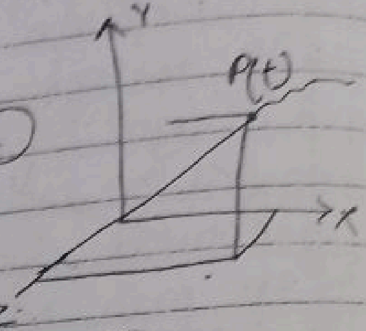


out → out
save none

30)

\# Cubic Curve (Parametric):

defined as $P(t) = \sum_{i=0}^{3} a_i t^i$ ... (I) $[0 <= t <= 1$



Expanding (I),

$$P(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0 \ldots \text{(II)}$$

Separated into 3 components:-

$$x(t) = a_{3x} t^3 + a_{2x} t^2 + a_{1x} t + a_{0x}$$
$$y(t) = a_{3y} t^3 + a_{2y} t^2 + a_{1y} t + a_{0y}$$
$$z(t) = a_{3z} t^3 + a_{2z} t^2 + a_{1z} t + a_{0z} \ldots \text{(III)}$$

defining a cubic curve using endpoints/tangent vectors is called <u>Hermite interpolation.</u>

Substituting $t=0$ $t=1$ In eq^n (II)

$$P(0) = a_0 \qquad P(1) = a_3 + a_2 + a_1 + a_0$$

To find tanget vectors differentiate (II) with respect to 't' and substitute $t=0$ $t=1$,

$$P'(0) = a_1 \qquad P'(1) = 3a_3 + 2a_2 + a_1$$

Now,

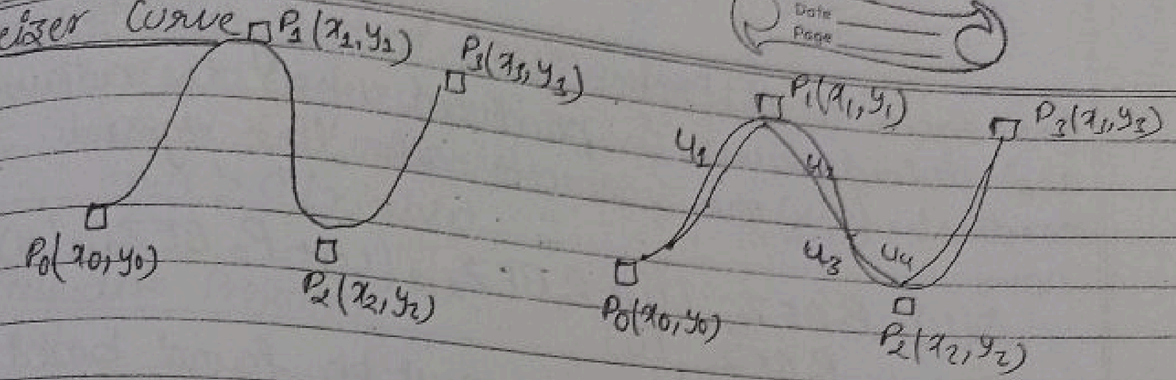
$$a_0 = P(0) \qquad a_1 = P'(0)$$
$$a_2 = -3P(0) + 3P(1) - 2P'(0) - P'(1)$$
$$a_3 = 2P(0) - 2P(1) + P'(0) + P'(1)$$

Substituting these values of 'a_i' in (II)

$$P(t) = (2t^3 - 3t^2 + 1)\,P(0) + (-2t^3 + 3t^2)\,P(1) + (t^3 - 2t^2 + t)\,P'(0) + (t^3 - t^2)\,P'(1)$$

By varying parameter 't' in these Blending function from 0 to 1 several points on curve segments can be found.

Beizer Curve



Here, P_0, P_1, P_2, P_3 are four control points
Number of segments in line ~~sing~~ segment : n seg

$i = 0$ to nseg

$u = \frac{1}{nseg}$ $[0,1]$ $1 \leq u < 0$

$u_0, u_1 \dots n$

$$x(u) = \sum_{j=0}^{n} x_j BEZ_{j,n}(u)$$

$$x(u) = x_0 BEZ_{0,3}(u) + x_1 BEZ_{1,3}(u) + x_2 BEZ_{2,3}(u) + x_3 BEZ_{2,3}(u)$$

and,

$$y(u) = \sum_{j=0}^{n} y_j BEZ_{j,n}$$

$$y(u) = y_0 BEZ_{0,3}(u) + y_1 BEZ_{1,3}(u) + y_2 BEZ_{2,3}(u) + y_3 BEZ_{2,3}(u)$$

The Beizer blending function $BEZ_{j,n}(u)$ is defined as,

$$BEZ_{j,n}(u) = \frac{n!}{j!\,(n-j)!} u^j (1-u)^{n-j}$$

where,

$$BEZ_{j,n}(u) = C(n,j)\, u^j (1-u)^{n-j}$$ Binomial coefficient

and $C(n,j) = \frac{n!}{j!\,(n-j)!}$

For each 'u' the coordinates x and y are computed and desired curve is produced when the adjacent coordinates (x, y) are connected with line segments.

now,

$$Q(u) = P_0 BEZ_{0,3}(u) + P_1 BEZ_{1,3}(u) + P_2 BEZ_{2,3}(u) + P_3 BEZ_{3,3}(u)$$

Four Blending functions must be found based on ~~Bernste~~ Bernstein Polynomials

$$BEZ_{0,3}(u) = \frac{3!}{0!\,3!} u^0 (1-u)^3 = (1-u)^3$$

$$BEZ_{1,3}(u) = \frac{3!}{1!\,2!} u^1 (1-u)^2 = 3u(1-u)^2$$

$$BEZ_{2,3}(u) = \frac{3!}{2!\,1!} u^2 (1-u) = 3u^2(1-u)$$

$$BEZ_{3,3}(u) = \frac{3!}{3!\,0!} u^3 (1-u)^0 = u^3$$

Substituting those in above eq^n,

$$Q(u) = (1-u)^3 P_0 + 3u(1-u)^2 P_1 + 3u^2(1-u) P_2 + u^2 P_3$$

when $u=0$ $Q(u)=P_0$ and when $u=1$, $Q(u)=P_3$

In matrix form $Q(u) = \left[(1-u)^3 \;\; 3u(1-u)^2 \;\; 3u^2(1-u) \;\; u^3\right] \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$

~~or~~,

$$\Rightarrow Q(u) = \left[(1-3u+3u^2-u^3) \;\; (3u-6u^2+3u^3) \;\; (3u^2-3u^3) \;\; u^3\right] \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

~~or~~,

$$\Rightarrow Q(u) = \left[u^3 \; u^2 \; u \; 1\right] \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

Face-detection method (OSM-method)

m n (as vectors)

$m \cdot n < 0$ (invisible)

$m \cdot n > 0$ (visible)

Prespective Projection

OpenGL

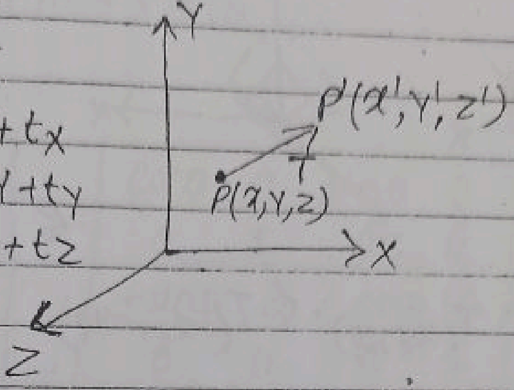
3D Transformations:-

* Translation:-

$x' = x + t_x$

$y' = y + t_y$

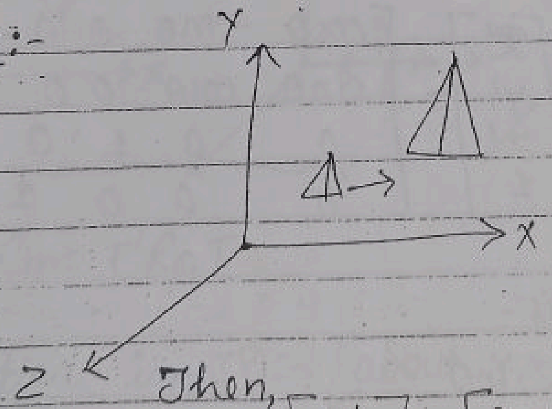
$z' = z + t_z$



Then,

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

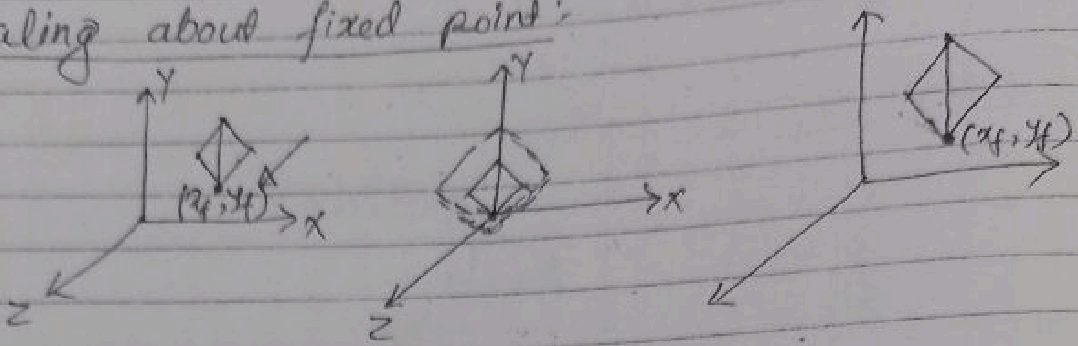
* Scaling:-



Then,

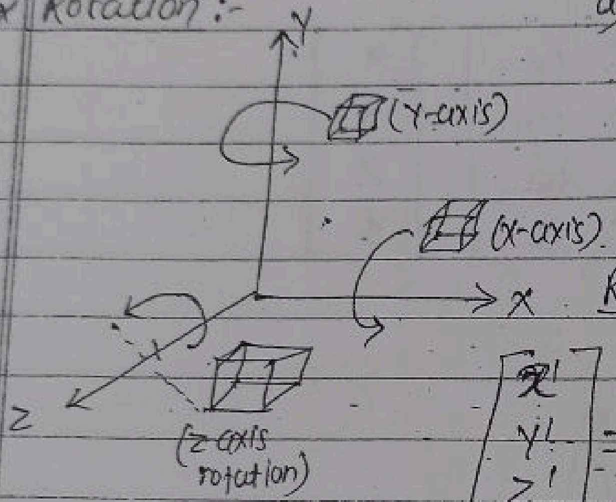
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Scaling about fixed point:-



$$C_m = T'_{(x_f, y_f)} \cdot S \; T_{(-x_f, -y_f)}$$

* **Rotation:-**



a) **Rotation about Z-axis:-**

$x' = X\cos\theta - Y\sin\theta$

$y' = X\sin\theta - Y\cos\theta$

$z' = z$

Rotate about Z-axis;

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

b) **Rotation about X-axis:-**

$y' = Y\cos\theta - Z\sin\theta$

$z' = Y\sin\theta + Z\cos\theta$

$x' = x$

Then,

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

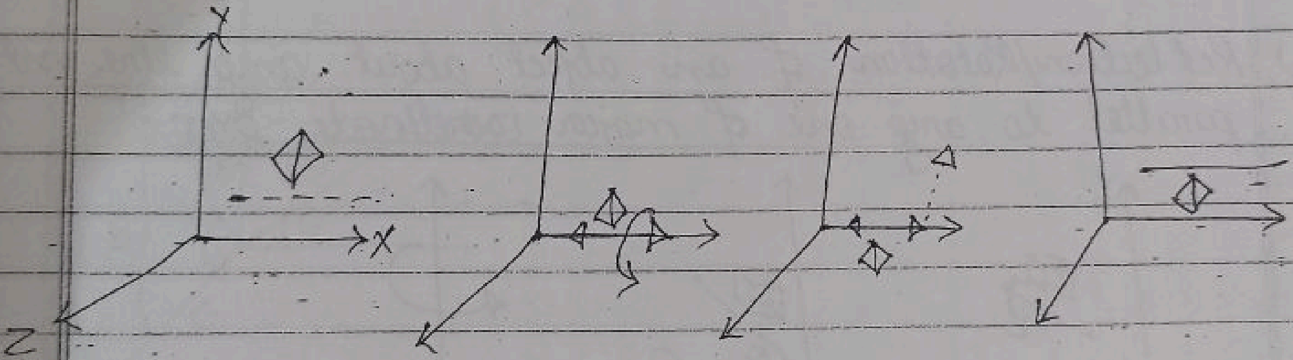
c) Rotation about Y-axis:-

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

Then,

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

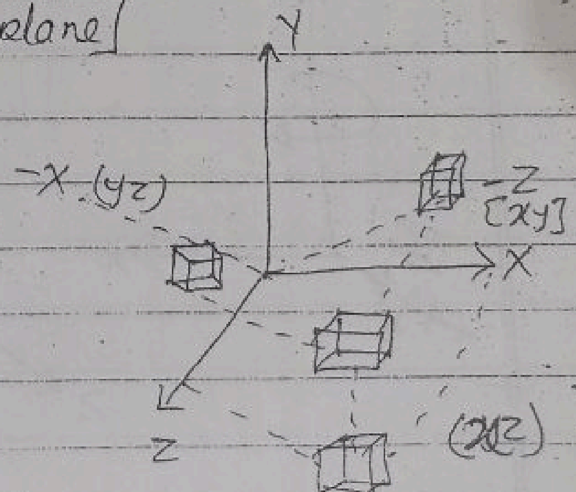
d)(i) Rotation of an object about any line parallel to any one of major coordinate axes:-



$C_m = T' R_\theta T$

(ii) Reflection in 3D:- [about yz plane]

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



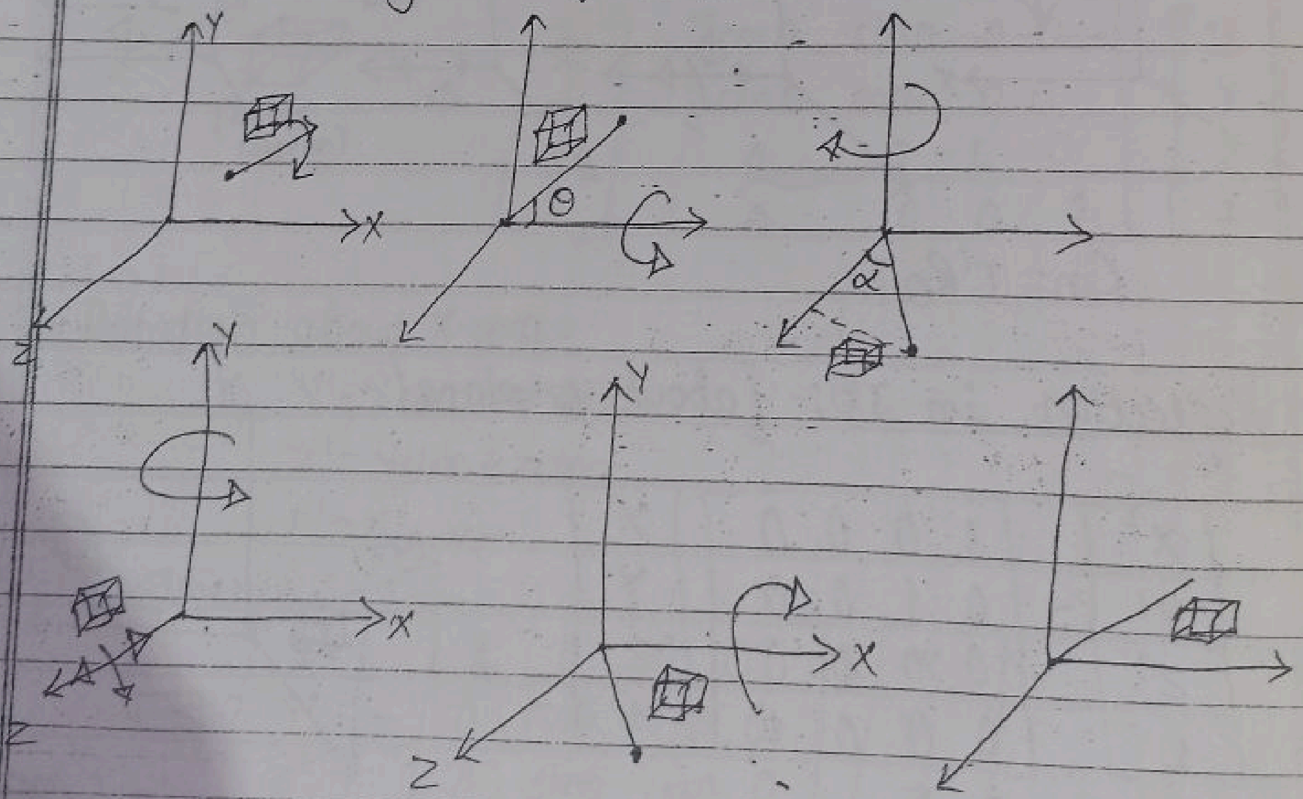
for xy:

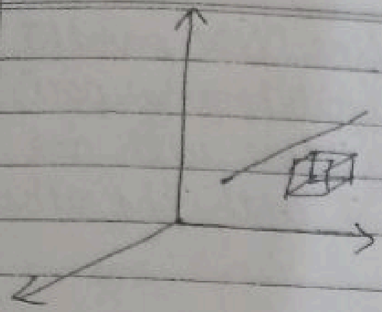
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

for xz;

$$\begin{bmatrix} x' \\ y \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

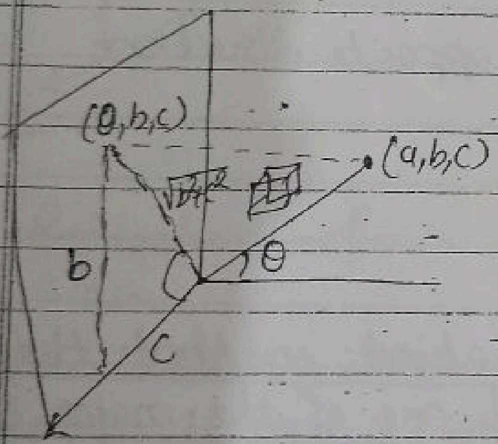
(e) Reflection/Rotation of an object about any line not parallel to any one of major coordinate axes:-





Translate → Rotate (R_x) → Rotate (R_y) → Reflect (yz-plane) → Rotate (R_y') → Rotate (R_x') → Translate

$C_m = T' R_x' R_y' R_{f_{yz}} R_y R_y T$



Reflection of an object with a line NOT parallel to any of the major coordinate axes (arbitrary axis)
(in 3D, to ~~dr~~ align an arbitrary line with one of the major coordinate axes, rotate it with the other two major coordinate axes)

Steps:-

1: Translate the object to origin with

$$T = \begin{bmatrix} 1 & 0 & 0 & -t_x \\ 0 & 1 & 0 & -t_y \\ 0 & 0 & 1 & -t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T' is the translation in opposite direction;

$$T' = \begin{bmatrix} 1 & 0 & 0 & +t_x \\ 0 & 1 & 0 & +t_y \\ 0 & 0 & 1 & +t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2: Rotate the line along with object so that the arbitrary line coincides with one of the major coordinate axes.

To align it with Z axis:-

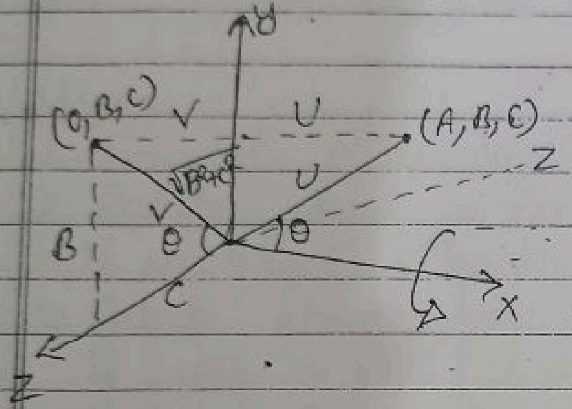
a: Rotate about X-axis in anticlockwise direction by 'θ' angle so that the shadow of line lies on z-axis.

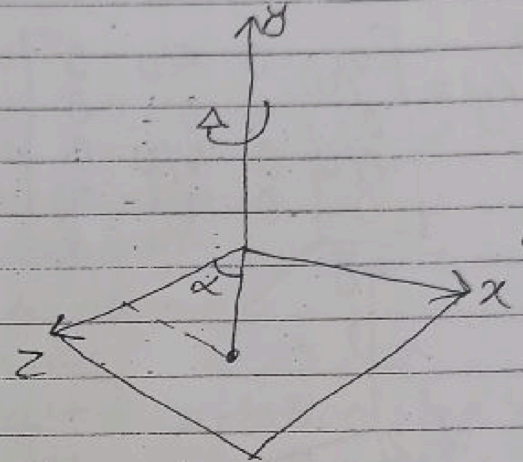
b: Rotate about Y-axis in clock-wise direction by 'α' angle so that the line is aligned with z-axis.

The transformation matrix in these case will be:-

The arbitrary axis (A, B, C) is translated to origin having its shadow in yz-plane.

So here $A=0$ and the shadow is $(0,B,C)$
Length of the arbitrary axis is;
$$U=\sqrt{A^2+B^2+C^2}$$
Length of the shadow is $V=\sqrt{b^2+c^2}$
we have,
$$\cos\theta=\frac{B}{V} \qquad \sin\theta=\frac{C}{V}$$





~~Reflection of~~
Now,
rotate about x-axis in clock-wise direction to place it on xz-plane with the transformation matrix; $R_x=\begin{bmatrix}1&0&0&0\\0&B/V&-C/V&0\\0&C/V&B/V&0\\0&0&0&1\end{bmatrix}$

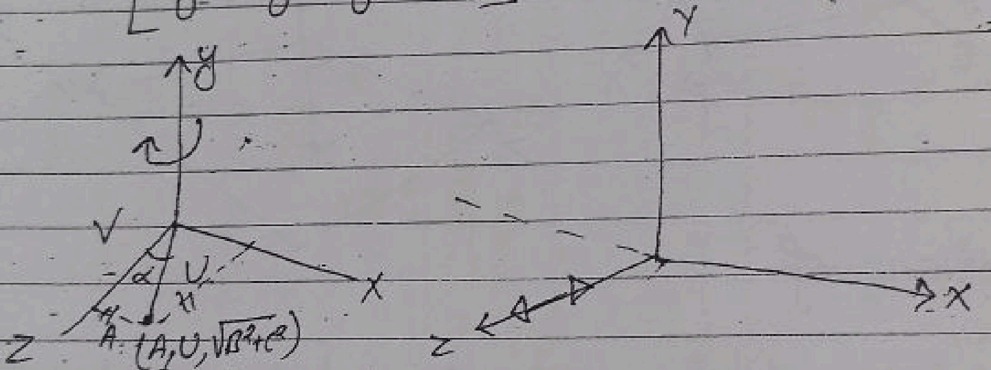
Rotation about x-axis in reverse direction is given by; $R_x'=\begin{bmatrix}1&0&0&0\\0&B/V&C/V&0\\0&-C/V&B/V&0\\0&0&0&1\end{bmatrix}$

Now,
Rotate about y-axis by 'α' angle in clock-wise direction so that the line is aligned with Z-axis;

$$R_y = \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ \sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} V/U & 0 & -A/U & 0 \\ 0 & 1 & 0 & 0 \\ A/U & 0 & V/U & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and,

$$R_y' = \begin{bmatrix} V/U & 0 & A/U & 0 \\ 0 & 1 & 0 & 0 \\ -A/U & 0 & V/U & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



3. Finally rotate or reflect the object about z-axis on yz-plane as we have aligned the arbitrary axis with the z-axis.

4. Reflection of object about yz-plane is given by:-

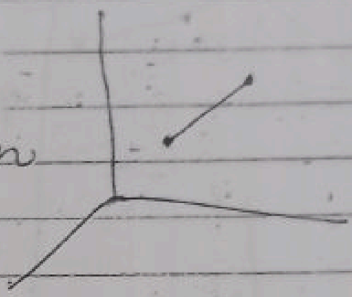
$$R_{f_{yz}} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4: Apply inverse rotations about y-axis then about x-axis as listed above.
5: Apply inverse translation

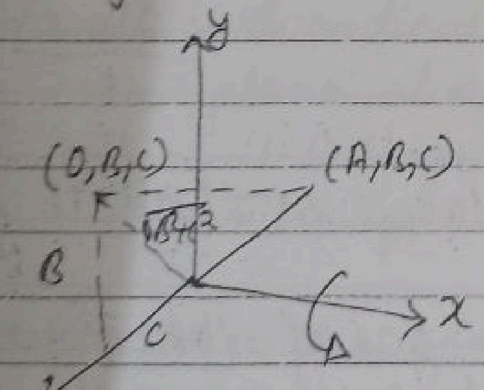
Q: Find the transformation matrix for rotating by an angle 'θ' with respect to a vector $N = A\hat{I} + B\hat{J} + C\hat{K}$ and a point in given vector is $P(a, b, c)$.

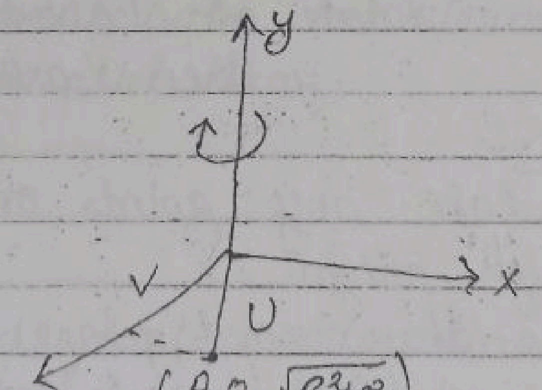
Sol^n

Steps:-



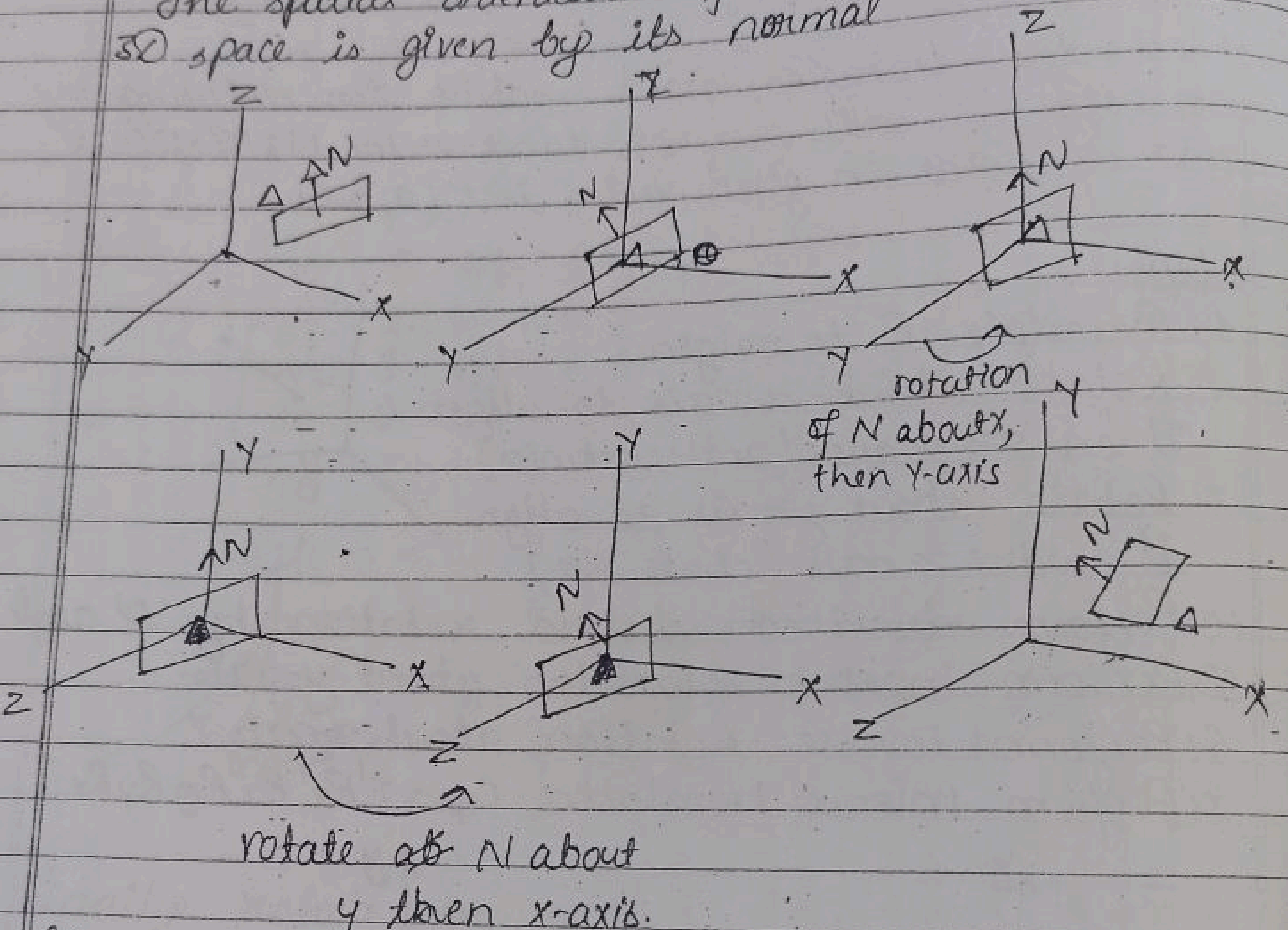
1: Translate P to origin
2: Rotate about x-axis to align it on xz-plane. (anti-clockwise)
3: Rotate about y-axis to align it with z-axis. (clock-wise)
4: Perform ~~described~~ desired rotation by 'θ' angle.
5: Perform inverse rotatation about y-axis.
6: Perform inverse rotation about x-axis.
7: Perform inverse translation; $C_m = T' R_x' R_y' R_\theta R_y R_x T$





Reflection of an object about a PLANE NOT parallel to any of the major coordinate axes:-

The spatial orientation of an arbitrary plane in 3D space is given by its normal.



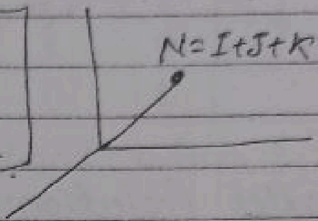
Steps:-

1: Translate any points on plane so that it passes through origin.
2: Aligin the normal (plane) with one of the major co-ordinate axis. (Perform rotations about the two other major axes)
3: Perform desired reflection about the plane.
4: De-align the normal (by rotating about the 2 major coordinate axes)
5: Apply inverse translation.

Q: Find the mirror reflection transformation with respect to a plane passing through point $P(2,2,2)$ and having a normal vector $N = I+J+K$.

Soln

$$T = \begin{bmatrix} 1 & 0 & 0 & -2 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T' = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$AN = \begin{bmatrix} \sqrt{2/3} & \sqrt{-1/6} & \sqrt{-1/6} & 0 \\ 0 & \sqrt{1/2} & \sqrt{-1/2} & 0 \\ \sqrt{1/3} & \sqrt{1/3} & \sqrt{1/3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$C_m = T' \cdot AN' \cdot R_f \cdot AN \cdot T$$

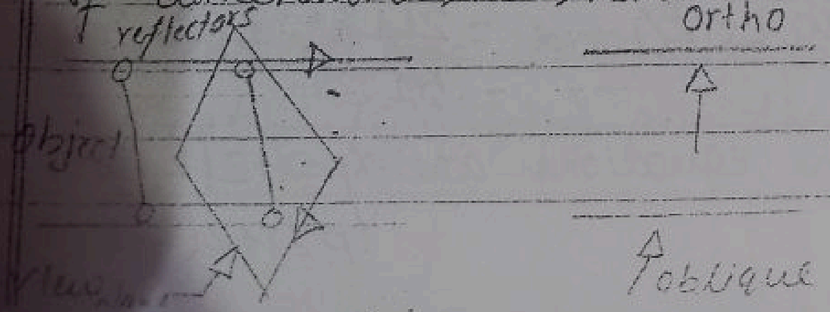
3D-Projections:-

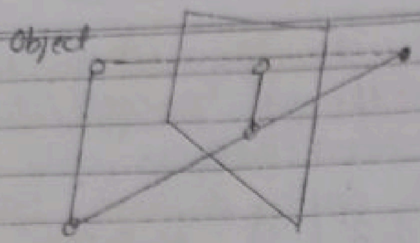
Parallel:- Coordinates points are transformed to view plane along parallel lines.

(i) Orthographic

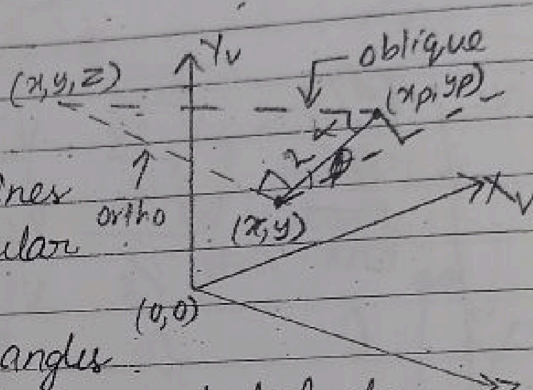
(ii) Oblique

Projection:- Transforming points in coordinate system of 'n' ~~direction~~ dimension into coordinate system of dimension less than 1.





Parallel Projection:-
(Oblique)



→ Obtained by projecting points along parallel lines that are not perpendicular to projection plane.

→ Often specified with 2 angles 'θ' and 'α'. Point (x,y,z) is projected to position (x_p, y_p) on viewplane. Orthographic projection coordinates on plane (x,y). Oblique projection line from (x,y,z) to (x_p, y_p) makes an angle 'α' with the line that joins (x_p, y_p) (x,y). This line of length 'L' is at an angle 'θ' with horizontal direction on projection plane. Expressing projection coordinates in terms of x, y, L and 'θ' as $x_p = x + L\cos\theta$

$$y_p = y + L\sin\theta$$

L depends on angle 'α' and z-coordinate of point to be projected $\tan\alpha = \frac{z}{L}$

$$\text{so, } L = z/\tan\alpha = \frac{zL_1}{1} \quad (\because L_1 \text{ is inverse of } \tan\alpha)$$

So, oblique projection equations are $x_p = x + z(L_1\cos\theta)$

$$y_p = y + z(L_1\sin\theta)$$

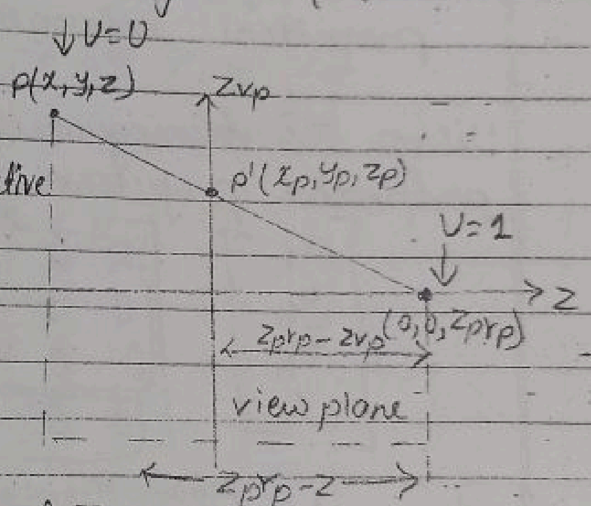
Transformatrix for producing any parallel projection onto $x_v y_v$ plane

$$M_{parallel} = \begin{bmatrix} 1 & 0 & L_1\cos\theta & 0 \\ 0 & 1 & L_1\sin\theta & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Orthographic projection is obtained when $L_1 = 0$ (for $\alpha = 90$)

Oblique projection is obtained for non-zero values for L_1.

Perspective Projection:-



To obtain the perspective projection of 3D object transforms points along projection lines that meet at projection reference point set at Z_{prp} along Z_v-axis and view-plane is at Z_{vp}.

Equations describing perspective projection ~~plane~~ along the perspective line in parametric form as

$$x' = x - xU \qquad y' = y - yU \qquad z' = z - (z - z_{prp})U.$$

Parameter 'U' takes values from 0 to 1 and coordinates (x', y', z') represents any point along projection line.

When $U=0$ we are at position $P(x,y,z)$ at the other end of line $U=1$ with projection reference coordinates $(0, 0, Z_{prp})$ on the view plane $z' = Z_{vp}$

Solving eq^n for parameter 'U' along projection line $U = \frac{Z_{vp} - Z}{Z_{prp} - Z}$

Substituting value of 'U';

$$X_p = \frac{X(Z_{prp} - Z_{vp})}{(Z_{prp} - Z)} = X\left(\frac{d_p}{Z_{prp} - Z}\right)$$

$$Y_p = Y\left(\frac{Z_{prp} - Z_{vp}}{Z_{prp} - Z}\right) = Y\left(\frac{d_p}{Z_{prp} - Z}\right)$$

where $d_p = Z_{prp} - Z_{vp}$ is distance of view plane from projection reference point.

Using 3D homogenous coordinate representation, perspective projection transformation matrix;

translate rotate overall

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{Z_{vp}}{d_p} & Z_{vp}(Z_{prp}/d_p) \\ 0 & 0 & -1/d_p & Z_{prp}/d_p \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

perspective parallel

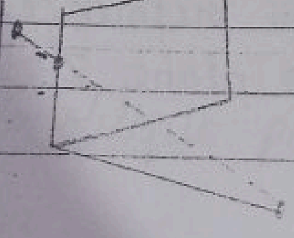
homogeneous beside '1'

here, homogenous factor is $h = \frac{Z_{prp} - Z}{d_p}$

Projection coordinates on viewplane $X_p = \frac{\cancel{x}\, x_h}{h}$

$$y_p = \frac{y_h}{h}$$

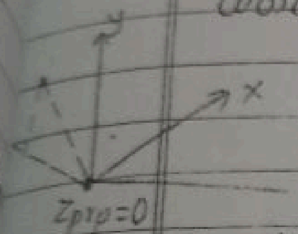
if viewplane is taken to be UV-plane $Z_{vp} = 0$ and projection coordinate will be;



$$x_p = x\left(\frac{Z_{prp}}{Z_{prp} - Z}\right) = x\left(\frac{1}{1 - \frac{Z}{Z_{prp}}}\right)$$

$$y_p = y\left(\frac{z_{prp} - \cancel{z}}{z_{prp} - z}\right) = y\left(\frac{1}{1 - \frac{z}{z_{prp}}}\right)$$

if projection reference is taken to be at viewing coordinate origin $z_{prp} = 0$



$$x_p = x\left(\frac{z_{vp}}{z}\right) = x\left(\frac{1}{z/z_{vp}}\right)$$

$$y_p = y\left(\frac{z_{vp}}{z}\right) = y\left(\frac{1}{z/z_{vp}}\right).$$

Standard perspective projection;

$$x' = x \cdot \frac{z_{prp} - z_{vp}}{z_{prp} - z} \qquad y' = y\,\frac{z_{prp} - z_{vp}}{z_{prp} - z}$$