Chapter 3 Two –Dimensional Transformations

The orientation, size, and shape of the output primitives are accomplished with geometric transformations that alter the coordinate descriptions of objects.

The basic geometric transformations are translation, rotation, and scaling. Other transformations that are often applied to objects include reflection and shear. In these all cases we consider the reference point is origin so if we have to do these transformations about any point then we have to shift these point to the origin first and then perform required operation and then again shift to that position.

Basic Transformations:

1. Translation
2. Rotation
3. Scaling
4. Reflection
5. Shearing

Homogenous Form:

We may have to perform more than one transformation in same object like scaling the object, then rotate the same object, and finally translation. For this, first co-ordinate positions are scaled, then this scaled co-ordinates are rotated and finally translated. A more efficient approach would be to combine the transformation so that final positions are obtained directly from initial co-ordinates thereby eliminating the calculation of intermediate co-ordinates. This allows us to represent all geometric transformation as matrix multiplication.

Expressing position in homogeneous coordinates allows us to represent all geometric transformations equation as matrix multiplications.

We represent each Cartesian co-ordinate position (x, y) with homogenous triple co-ordinate (x_h, y_h, h)

(x, y) -------------------- (x_h, y_h,h)

$x = x_h/h$ $y = y_h/h$

where h is any non zero value

For convenient h=1

(x,y)-------------------------(x, y, 1).

Chapter 3 Two –Dimensional Transformations

Translation:

A translation is applied to an object by repositioning it along a straight-line path from one coordinate location to another. We translate a two-dimensional point by adding translation distances, t_x, and t_y, to the original coordinate position (x, y) to move the point to a new position (x ' , y').

Every point on the object is translated by the same amount

$$x' = x + t_x, \qquad y' = y + t_y$$

$$P = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \qquad p' = \begin{bmatrix} x_1' \\ y_1' \end{bmatrix}, \qquad T = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

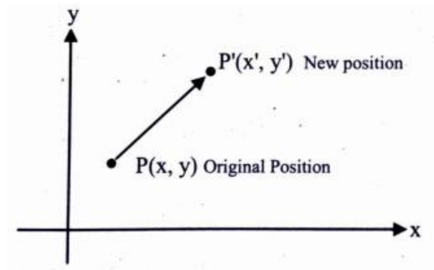
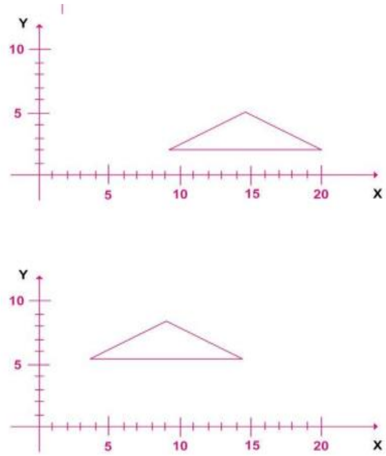


Figure 3.1 Translation of a point

In homogeneous representation if position P = (x, y) is translated to new position P’= (x’, y’) then:



$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow P' = T(t_x, t_y).P$$

Figure 3.2 Translation of a object

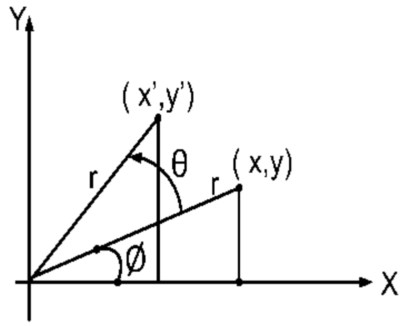
Rotation:

A two-dimensional rotation is applied to an object by repositioning it along a circular path in the xy plane. To generate a rotation, we specify a rotation angle θ and the position (x_r, y_r) of the rotation point (or pivot point) about which the object is to be rotated.

- + Value for ‘θ’ define counter-clockwise rotation about a point
- -Value for ‘θ’ defines clockwise rotation about a point

REQUIREMENT:
Rotation angle θ
&
pivot point

- Clockwise rotation (Negative)
- Anticlockwise rotation (positive)

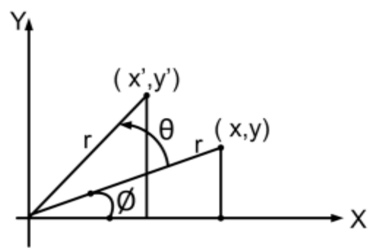


At origin

Coordinates of point (x,y) in polar form

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

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



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

$$P' = R.P$$

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

ROTATION MATRX

If Co-ordinates represented as row vector, Then:

$$P'^T = (R.P)^T$$
$$= P^T.R^T$$

In homogeneous co-ordinate

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow P' = R(\theta).P$$

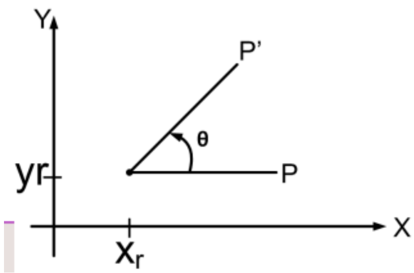
Clockwise direction:

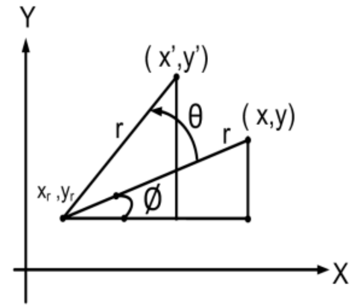
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow P' = R(\theta).P$$

Rotation of a point (x, y) about any point (x_r, y_r) / Fixed Point Rotation:

Steps

1. Translate object so as to coincide pivot to origin
2. Rotate object about the origin
3. Translate object back so as to return pivot to original position





Composite Transformations

$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta)+y_r\sin\theta \\ \sin\theta & \cos\theta & y_r(1-\cos\theta)-x_r\sin\theta \\ 0 & 0 & 1 \end{bmatrix}$$

$$T(x_r, y_r).R(\theta).T(-x_r,-y_r) = R(x_r, y_r, \theta)$$

$$T(-x_r,-y_r) = T^{-1}(x_r, y_r)$$

Q1. Rotate the triangle (5, 5), (7, 3), (3, 3) in counter clockwise (CCW) by 90 degree.

Answer:

$P' = R.\ P$

$$= \begin{pmatrix} \text{Cos90} & \text{-Sin90} & 0 \\ \text{Sin90} & \text{Cos90} & 0 \\ 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} 5 & 7 & 3 \\ 5 & 3 & 3 \\ 1 & 1 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} 5 & 7 & 3 \\ 5 & 3 & 3 \\ 1 & 1 & 1 \end{pmatrix}$$

$$= \begin{bmatrix} -5 & -3 & -3 \\ 5 & 7 & 3 \\ 1 & 1 & 1 \end{bmatrix}$$

Q2. Rotate the triangle (5, 5), (7, 3), (3, 3) about fixed point (5, 4) in counter clockwise (CCW) by 90 degree.

Solution:

Here, the required steps are:

1. Translate the fixed point to origin.
2. Rotate about the origin by specified angle θ.
3. Reverse the translation as performed earlier.

Thus, the composite matrix is given by

Com = T(xf, yf) . R_θ .T(-xf, -yf)

$$= \begin{pmatrix} 1 & 0 & 5 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \text{Cos}90 & -\text{Sin}90 & 0 \\ \text{Sin}90 & \text{Cos}90 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -5 \\ 0 & 1 & -4 \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 5 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -5 \\ 0 & 1 & -4 \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 5 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -1 & 4 \\ 1 & 0 & -5 \\ 0 & 0 & 1 \end{pmatrix}$$

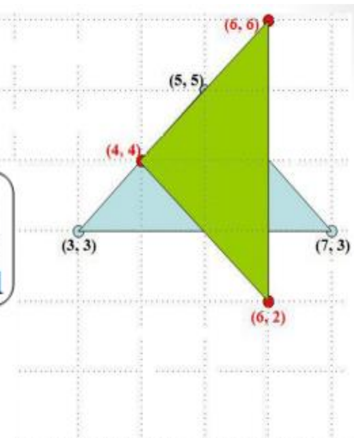
$$= \begin{pmatrix} 0 & -1 & 9 \\ 1 & 0 & -1 \\ 0 & 0 & 1 \end{pmatrix}$$

Now, the required co-ordinate can be calculated as:

P' = Com . P

$$= \begin{pmatrix} 0 & -1 & 9 \\ 1 & 0 & -1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 5 & 7 & 3 \\ 5 & 3 & 3 \\ 1 & 1 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 4 & 6 & 6 \\ 4 & 6 & 2 \\ 1 & 1 & 1 \end{pmatrix}$$



Hence, the new required coordinate points are **(4, 4), (6, 6) & (6, 2).**

Scaling:

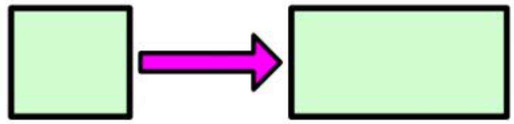
A scaling transformation alters the size of an object. This operation can be carried out for polygons by multiplying the coordinate values (x, y) of each vertex by scaling factors s_x and s_y to produce the transformed coordinates (x', y').

- s_x scales object in 'x' direction
- s_y scales object in 'y' direction

$$x' = x.s_x, \qquad y' = y.s_y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}}_{\text{2 x 2 Scaling Matrix}} . \begin{bmatrix} x \\ y \end{bmatrix}$$

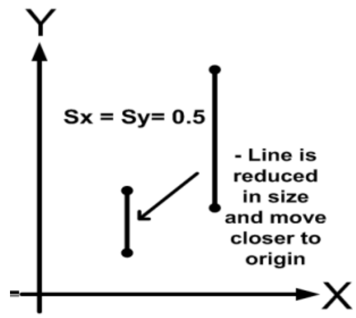
$$P' = S.P$$



Square to Rectangle (Sx = 2 , Sy =1)
(Sx, Sy)→ Scaling factors
Sx=Sy→ Uniform Scaling
Sx≠ Sy → Differential Scaling

In homogeneous co-ordinate

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow P' = S(s_x, s_y).P$$

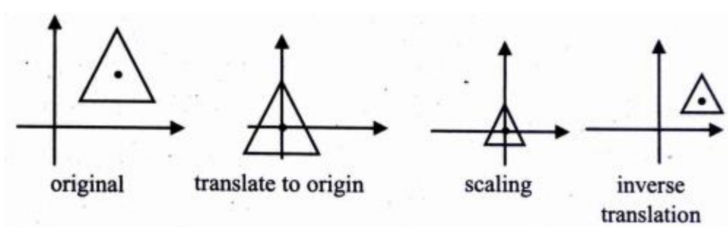


Fixed point scaling:

The location of the scaled object can be controlled by choosing a position called fixed point that is to remain unchanged after the scaling transformation. Fixed point (x_f, y_f) can be chosen as one of the vertices, centroid of the object, or any other position.

Steps:

- Translate object so that the fixed point coincides with the co-ordinate origin.
- Scale the object with respect to the co-ordinate origin.
- Use the inverse translation of steps 1 to return the object to its original position.



$$\mathrm{T}(x_f, y_f) \cdot \mathrm{S}(s_x, s_y) \cdot \mathrm{T}(-x_f, -y_f) = \mathrm{S}(x_f, y_f, s_x, s_y)$$

$$\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & x_f(1 - s_x) \\ 0 & s_y & y_f(1 - s_y) \\ 0 & 0 & 1 \end{bmatrix}$$

Reflection:

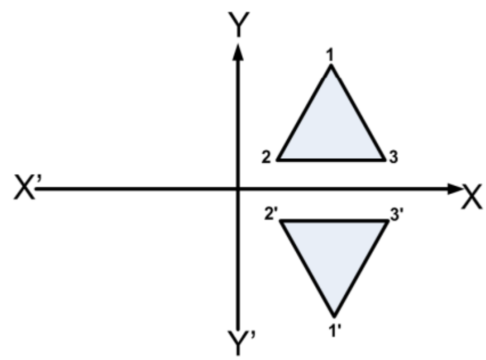
A reflection is a transformation that produces a mirror image of an object. The mirror image for a 2D reflection is generated relative to an axis of reflection by rotating the object 180" about the reflection axis. We can choose an axis of reflection in the xy-plane or perpendicular to the xy plane. When the reflection axis is a line in the xy plane, the rotation path about this axis is in a plane perpendicular to the xy-plane. For reflection axes that are perpendicular to the xy-plane, the rotation path is in the xy plane.

i. Reflection about x axis or about line y = 0

Keeps **X** value same but flips **Y** value of coordinate points

x' = x
y' = -y

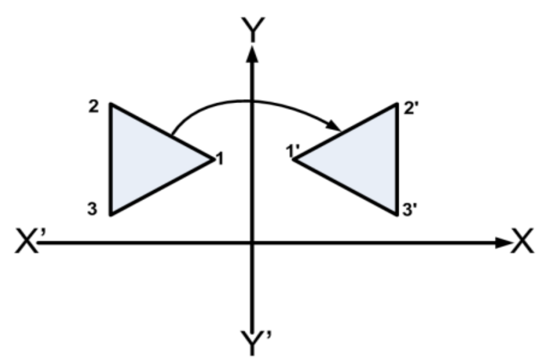
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



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

Figure 3.3 Reflection of object about x-axis.

ii. Reflection about y axis or about line x = 0



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

Figure 3.4 Reflection of object about y-axis.

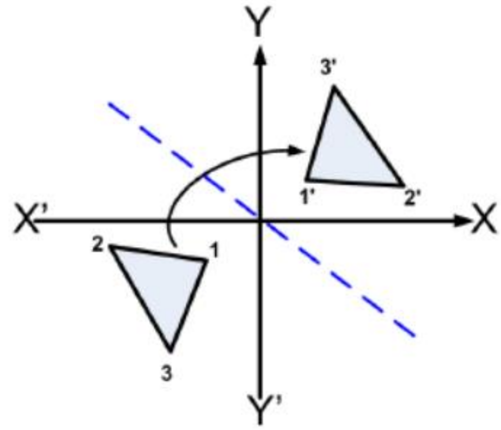
iii. Reflection about origin

Flip both ‘x’ and ‘y’ coordinates of a point

x’ = -x

y’ = -y

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



Homogeneous co-ordinate

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

iv. Reflection about line y = x

x’ = y

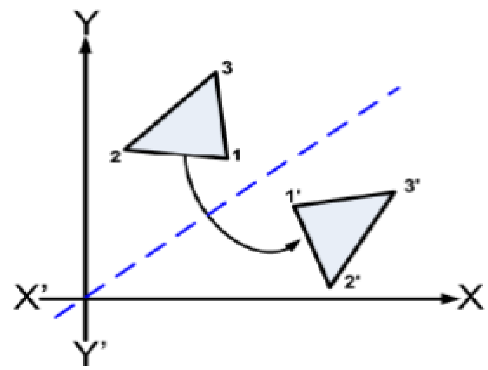
y’ = x

Thus, reflection against x=y-axis (i.e. θ =45)

$$R_{x=y} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Hence

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



Homogeneous co-ordinate

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

Equivalent to:

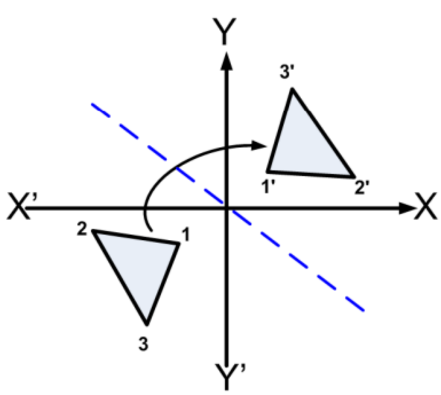
- Reflection about x-axis
- Rotate anticlockwise **90°**

OR

- Clockwise rotation 45 °
- Reflection with x-axis
- anticlockwise rotation 45 °

v. Reflection about line y = - x

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

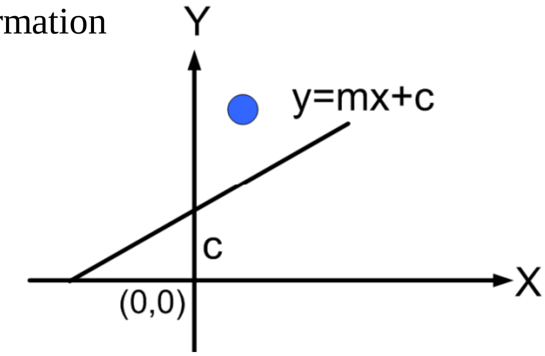


Equivalent to:

- Clockwise rotation 45 °
- Reflection with y-axis
- anticlockwise rotation 45 °

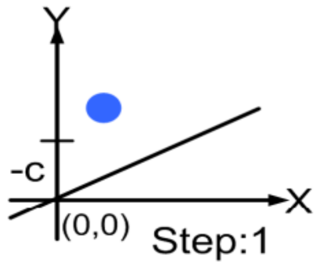
vi. Reflection about y=m*x + c

Combination of translate-rotate-reflect transformation



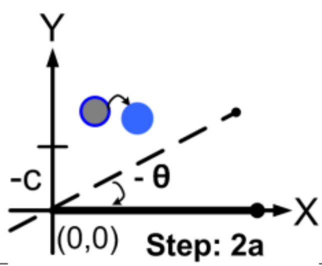
1. First translate the line so that it passes through the origin

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -c \\ 0 & 0 & 1 \end{bmatrix}$$
translatio n

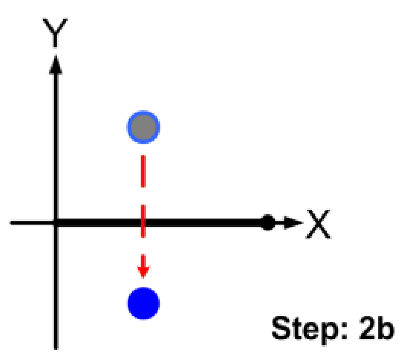


2. Rotate the line onto one of the coordinate axes(say x-axis) and reflect about that axis (x-axis)

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



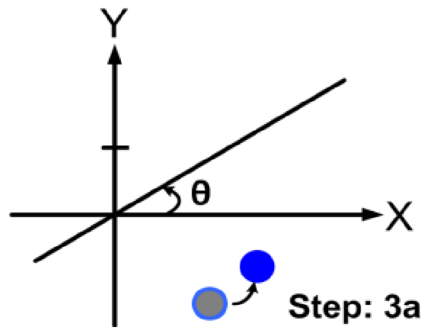
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
reflection



3. Finally, restore the line to its original position with the inverse rotation and translation transformation.

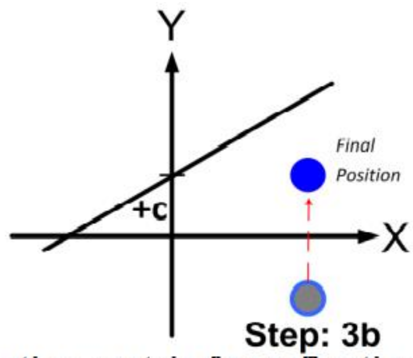
$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

rotation



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

translation



- Thus the composite transformation matrix for reflection about y=m*x+c is

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

Substituting value of tanθ , sinθ & cosθ you will get the reflection matrix

$$\begin{bmatrix} \frac{1-m^2}{1+m^2} & \frac{2m}{1+m^2} & \frac{-2cm}{1+m^2} \\ \frac{2m}{1+m^2} & \frac{m^2-1}{1+m^2} & \frac{2c}{1+m^2} \\ 0 & 0 & 1 \end{bmatrix}$$

Q1. Reflect an object (2, 3), (4, 3), (4, 5) about line y = x +1.

Solution

Here,

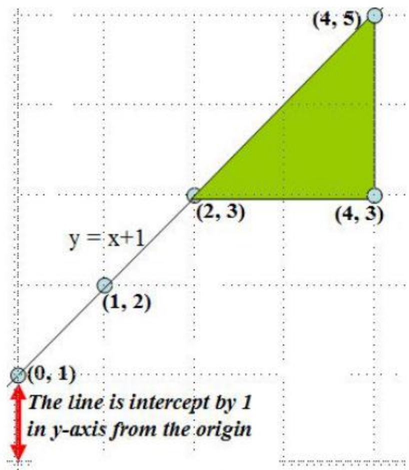
The given line is *y = x +1.*

Thus,

When x = 0, y=1

When x = 1, y=2

When x = 2, y=3



Also,

The slope of the line (m) = 1

Thus, the rotation angle (θ) = $Tan^{-1}(m) = Tan^{-1}(1) = 45^0$

Here, the required steps are:

- Translate the line to origin by decreasing the y-intercept with one.
- Rotate the line by angle 45^0 in clockwise direction so that the given line must overlap x-axis.
- Reflect the object about the x-axis.
- Reverse rotate the line by angle -45^0 in counter-clockwise direction.
- Reverse translate the line to original position by adding the y-intercept with one.

Thus, the composite matrix is given by:

$$CM=T'_{(0,C)} \cdot R'_{\theta} \cdot R_{refl} \cdot R_{\theta} \cdot T_{(0,-C)}$$

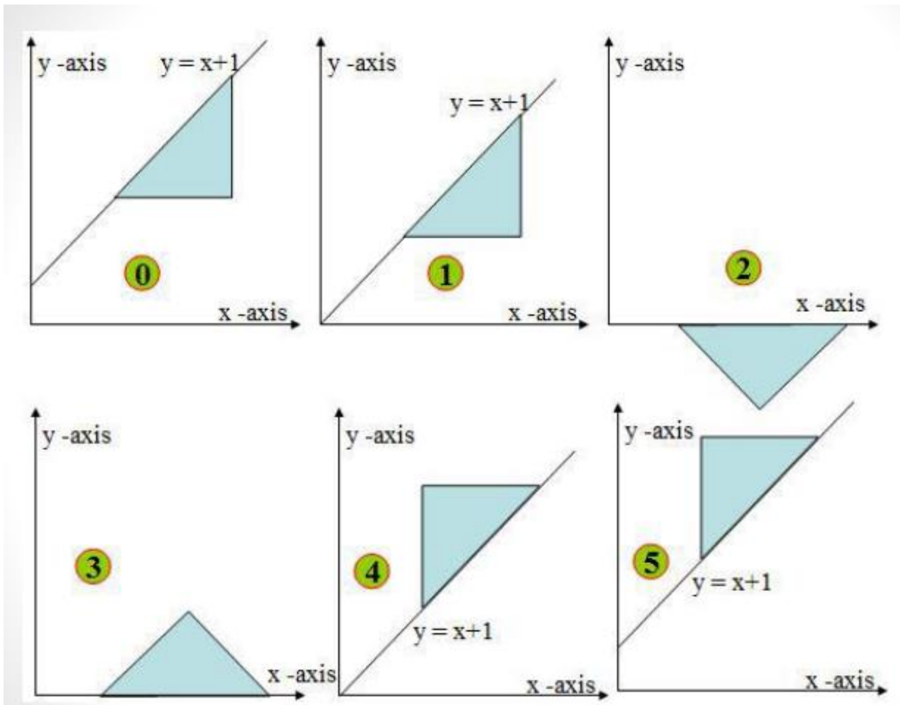
$$= \overset{\textit{Addition y-intercept}}{\begin{pmatrix}1 & 0 & 0\\0 & 1 & 1\\0 & 0 & 1\end{pmatrix}} \overset{\textit{CCW Rotation}}{\begin{pmatrix}Cos45 & -Sin45 & 0\\Sin45 & Cos45 & 0\\0 & 0 & 1\end{pmatrix}} \overset{\textit{Reflection about x-axis}}{\begin{pmatrix}1 & 0 & 0\\0 & -1 & 0\\0 & 0 & 1\end{pmatrix}} \overset{\textit{CW Rotation}}{\begin{pmatrix}Cos45 & Sin45 & 0\\-Sin45 & Cos45 & 0\\0 & 0 & 1\end{pmatrix}} \overset{\textit{Reduce y-intercept}}{\begin{pmatrix}1 & 0 & 0\\0 & 1 & -1\\0 & 0 & 1\end{pmatrix}}$$

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

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

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

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & -1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & -1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$



Now, the required co-ordinate can be calculated as:

P' = Com x P

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

$$= \begin{pmatrix} 2 & 2 & 4 \\ 3 & 5 & 5 \\ 1 & 1 & 1 \end{pmatrix}$$

Hence, the final coordinates are (2, 3), (2, 5) & (4, 5).

Q2. A mirror is placed such that it passes through (0, 10), (10, 0). Fin the mirror image of an object (6, 7), (7, 6), (6, 9).

Solution

Here,

The given mirror or line is passing through the points *(0, 10) & (10, 0).*

Now, the slope of the line

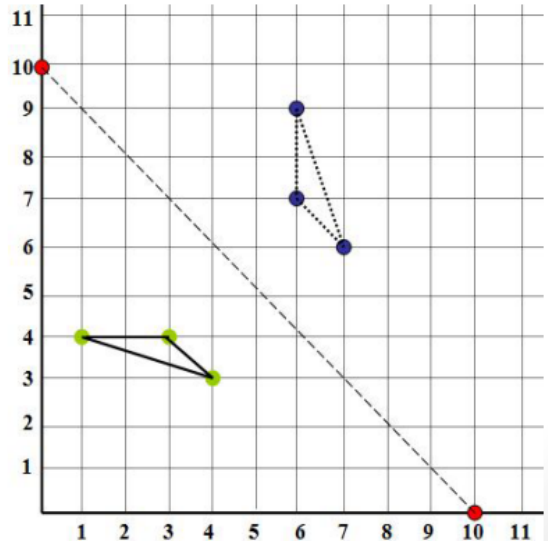
(m) = (y2-y1) / (x2-x1)

= (0 - 10) / (10 - 0) = -1

Thus, the rotation angle (**θ**)

= $\text{Tan}^{-1}(m) = \text{Tan}^{-1}(-1)$

= -45^0



The composite matrix is given by:

Com

$$= T_{(0,\,10)\text{ or }(10,\,0)} \cdot R_{\theta\text{ in CW}} \cdot R_{fx} \cdot R_{\theta\text{ in CCW}} \cdot T_{(0,\,-10)\text{ or }(-10,\,0)}$$

Addition x-intercept · *CW Rotation* · *Reflection about x-axis* · *CCW Rotation* · *Reduce x-intercept*

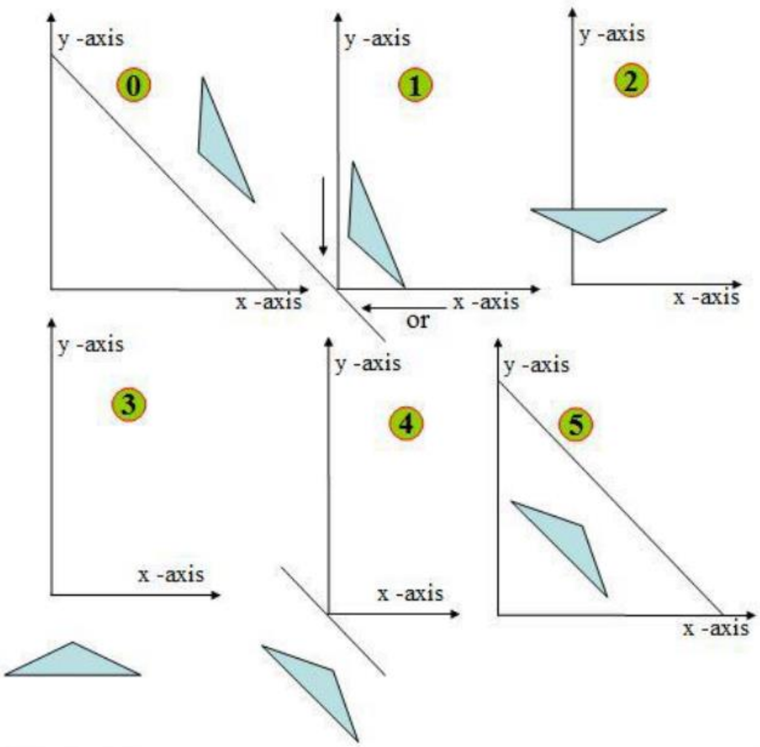
$$= \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \text{Cos45} & \text{Sin45} & 0 \\ -\text{Sin45} & \text{Cos45} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \text{Cos45} & -\text{Sin45} & 0 \\ \text{Sin45} & \text{Cos45} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -10 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -10 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & -1/\sqrt{2} & -10/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} & -10/\sqrt{2} \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & -1/\sqrt{2} & -10/\sqrt{2} \\ -1/\sqrt{2} & -1/\sqrt{2} & 10/\sqrt{2} \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -1 & 0 \\ -1 & 0 & 10 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & -1 & 10 \\ -1 & 0 & 10 \\ 0 & 0 & 1 \end{pmatrix}$$



Now, the required co-ordinate can be calculated as:

P' = Com . P

$$= \begin{pmatrix} 0 & -1 & 10 \\ -1 & 0 & 10 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 6 & 7 & 6 \\ 7 & 6 & 9 \\ 1 & 1 & 1 \end{pmatrix}$$

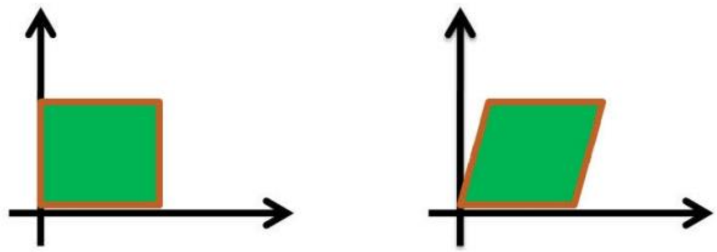
$$= \begin{pmatrix} 3 & 4 & 1 \\ 4 & 3 & 4 \\ 1 & 1 & 1 \end{pmatrix}$$

Hence, the final coordinates are (3, 4), (4, 3) & (1, 4).

Shearing:

It distorts the shape of object in either 'x' or 'y' or both direction. In case of single directional shearing (e.g. in 'x' direction can be viewed as an object made up of very thin layer and slid over each other with the base remaining where it is). Shearing is a non-rigid-body transformation that moves objects with deformation.

Shearing factor (Sh_x, Sh_y)



x- direction shear relative to x-axis

x'=x+y*Shx

y'=y

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

y- direction shear relative to y-axis

x'=x

y'=y +x *Shy

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

x- direction shear relative to other reference line y=y_{ref}

x'=x +Shx(y-y_{ref})

y'=y

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & Shx & -Shx \cdot y_{ref} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

y- direction shear relative to other reference line x=x_{ref}

x'=x

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

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ Shy & 1 & -Shy \cdot X_{ref} \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Composite Transformations:

Translation:

For Successive Translation
vectors (tx1,ty1) and (tx2,ty2)

$$P' = T(t_{x2}, t_{y2}).\{T(t_{x1}, t_{y1}).P\}$$
$$= \{T(t_{x2}, t_{y2}).T(t_{x1}, t_{y1})\}.P$$

Successive Translations are additive

$$\begin{bmatrix} 1 & 0 & t_{x2} \\ 0 & 1 & t_{y2} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_{x1} \\ 0 & 1 & t_{y1} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_{x1} + t_{x2} \\ 0 & 1 & t_{y1} + t_{y2} \\ 0 & 0 & 1 \end{bmatrix}$$

$$T(t_{x2}, t_{y2}).T(t_{x1}, t_{y1}) = T(t_{x1} + t_{x2}, t_{y1} + t_{y2})$$

Rotation:

For Successive Rotations θ_1
and θ_2

$$P' = R(\theta_2).\{R(\theta_1).P\}$$
$$= \{R(\theta_2).R(\theta_1)\}.P$$

Successive Rotations are additive.

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

Scaling:

Successive Scaling are multiplicative

$$S(s_{x2}, s_{y2}).S(s_{x1}, s_{y1}) = S(s_{x1}.s_{x2}, s_{y1}.s_{y2})$$

$$\begin{bmatrix} s_{x2} & 0 & 0 \\ 0 & s_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} s_{x1} & 0 & 0 \\ 0 & s_{y1} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_{x1}.s_{x2} & 0 & 0 \\ 0 & S_{y1}.s_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Fixed Point Scaling

$$\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & x_f(1-s_x) \\ 0 & s_y & y_f(1-s_y) \\ 0 & 0 & 1 \end{bmatrix}$$

$$T(x_f, y_f).S(s_x, s_y).T(-x_f, -y_f) = S(x_f, y_f, s_x, s_y)$$

Two-dimensional viewing:

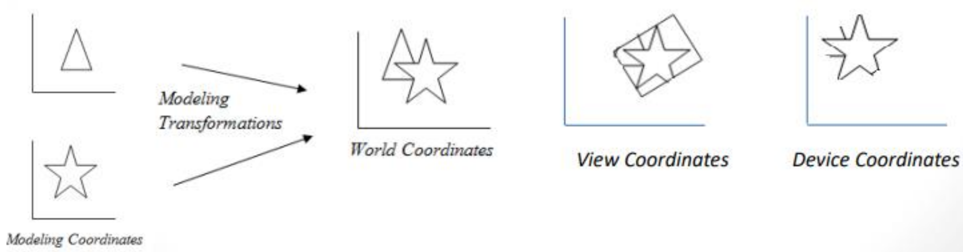
Two-dimensional viewing is the mechanism for displaying views of a picture on an output device. Much like what we see in real life through a small window on the wall or the viewfinder of a camera, a computer-generated image often depicts a partial view of a large scene. For a 2-D picture, a view is selected by specifying a subarea of the total picture area.

Modeling Coordinates:

Modeling coordinates are used to construct shape of individual parts (objects or structures) of a 2D scene. For example, generating a circle at the origin with a “radius” of 2 units. Here, origin (0, 0), and radius 2 units are modeling coordinates.

Modeling coordinates define object shape. Can be floating-point, integers and can represent units like km, m, miles, feet etc.

Chapter 3 Two –Dimensional Transformations



World Coordinates:

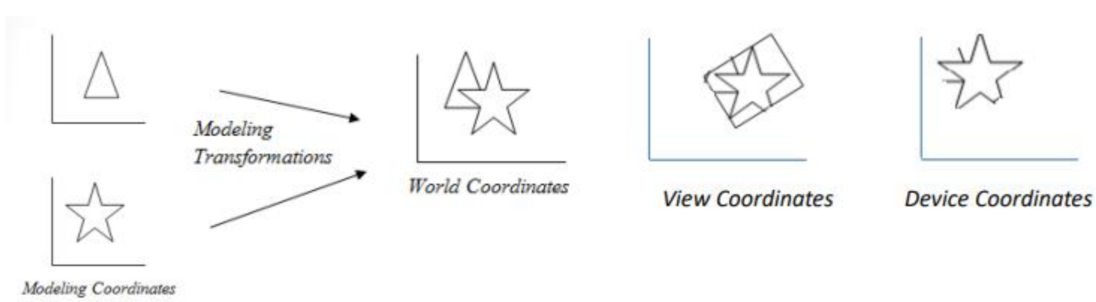
- World coordinates are used to organize the individual parts into a scene.
- World coordinates units define overall scene to be modeled.
- World coordinates represent relative positions of objects. Can be floating-point, integers and can represent units like km, m, miles etc.

Viewing Coordinates:

- Viewing coordinates are used to define particular view of the user. Viewer's position and view angle i.e. rotated/translated.
- Viewing coordinates specify the portion of the output device that is to be used to present the view.
- Normalized viewing coordinates are viewing coordinates between 0 and 1 in each direction. They are used to make the viewing process independent of the output device (paper, mobile).

Device Coordinates or Screen Coordinates:

- The display coordinate system is called device coordinate system. Device coordinates are specific to output device.
- Device coordinates are integers within the range (0, 0) to (xmax, ymax) for a particular output device.

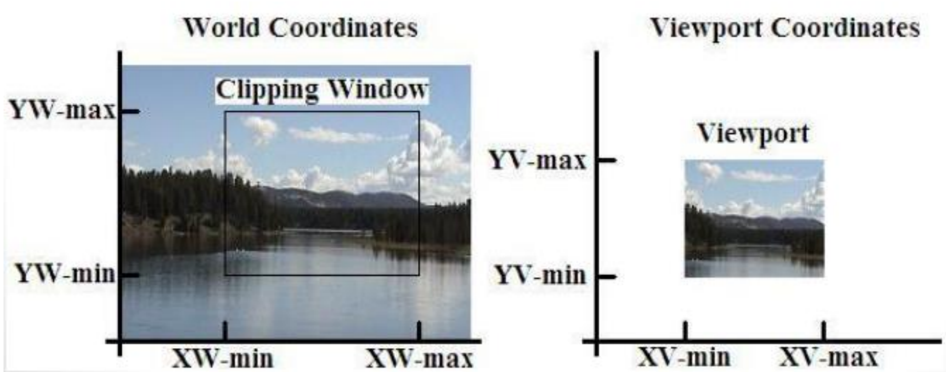


Window:

- A world-coordinate area selected for display is called a window or clipping window. That is, window is the section of the 2D scene that is selected for viewing.
- The window defines what is to be viewed.

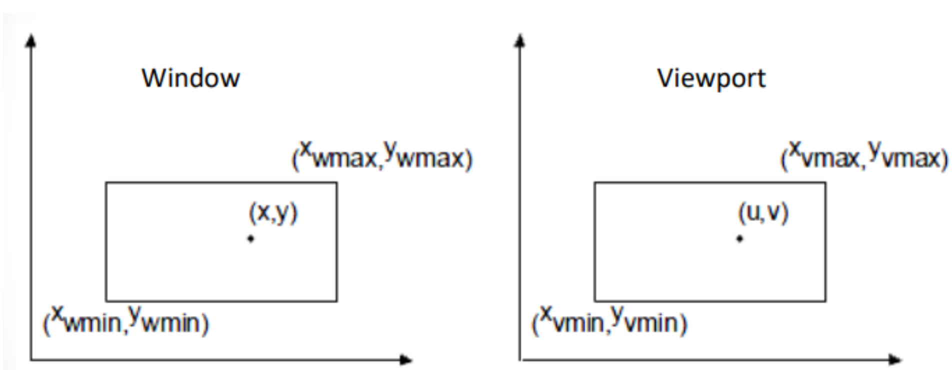
Viewport:

- An area on a display device to which a window is mapped is called a viewport.
- The viewport indicates where on an output device selected part will be displayed.



Window to Viewport Transformation:

A window is specified by four world coordinates: X_{wmin}, X_{wmax}, Y_{wmin} and Y_{wmax} Similarly, a viewport is described by four normalized device coordinates: X_{vmin}, X_{vmax}, Y_{vmin} and Y_{vmax}.



1. Translate the window to the origin. That is, apply T ($-X_{wmin}$, $-Y_{wmin}$)

2. Scale it to the size of the viewport. That is, apply S (S_x, S_y)

3. Translate scaled window to the position of the viewport. That is, apply T (X_{vmin}, Y_{vmin}).

Therefore, net transformation, $T_{WV} = T(X_{vmin}, Y_{vmin}).\ S(S_x, S_y).T(-X_{wmin}, -Y_{wmin})$

It maintains the same relative placement of objects in normalized space as in viewing coordinates.

Let (x, y) be the world coordinate point that is mapped onto the viewport point (u, v), then we must have

$$= \quad \frac{u - x_{v\min}}{x_{v\max} - x_{v\min}} = \frac{x - x_{w\min}}{x_{w\max} - x_{w\min}}$$

$$= \quad u = x_{vmin} + \frac{x_{v\max} - x_{v\min}}{x_{w\max} - x_{w\min}} . (x - x_{wmin})$$

$$= \quad \frac{v - y_{v\min}}{y_{v\max} - y_{v\min}} = \frac{y - y_{w\min}}{y_{w\max} - y_{w\min}}$$

$$= \quad v = y_{vmin} + \frac{y_{v\max} - y_{v\min}}{y_{w\max} - y_{w\min}} . (y - y_{wmin})$$

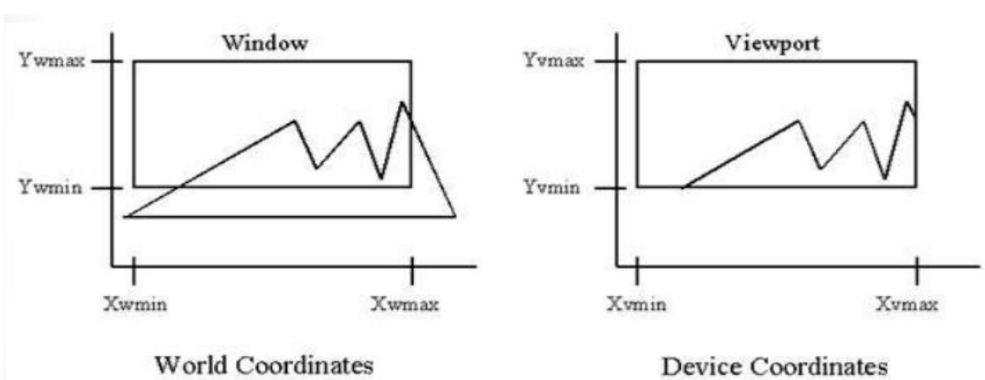
But, we know that

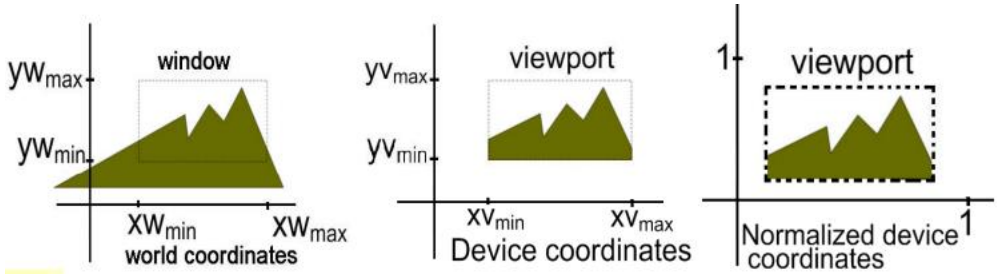
$$u = x_{vmin} + s_x(x - x_{wmin})$$
$$v = y_{vmin} + s_y(y - y_{wmin})$$

Where

$$s_x = \frac{x_{v\max} - x_{w\min}}{x_{w\max} - x_{w\min}}$$

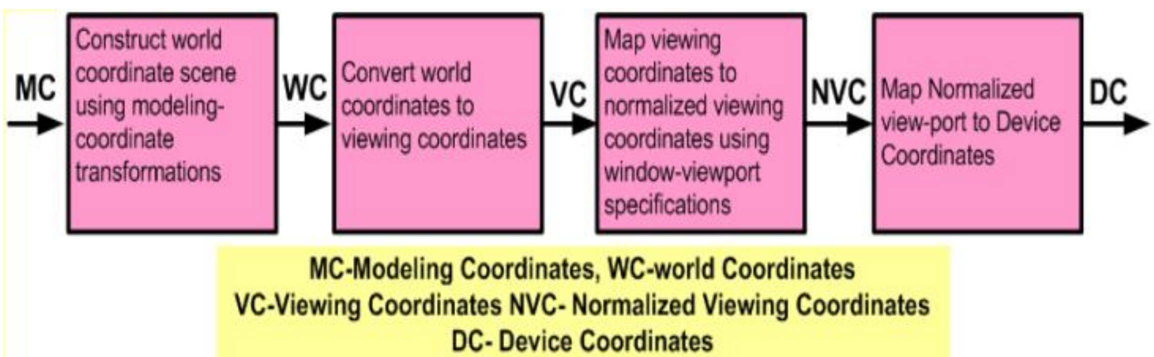
$$s_y = \frac{y_{v\max} - y_{v\min}}{y_{w\max} - y_{w\min}}$$





Window and viewport are often rectangular in standard positions, because it simplifies the transformation process and clipping process. Other shapes such as polygons, circles take longer time to process. By changing the position of the viewport, we can view objects at different positions on the display area of an output device. Also by varying the size of viewports, we can change size of displayed objects. Zooming effects can be obtained by successively mapping different-sized windows on a fixed-sized viewport.

Two-Dimensional Viewing Pipeline:



The mapping of a part of a world co-ordinate scene to a device coordinate is referred to as viewing transformation.

Sometimes, the 2D viewing transformation is simply referred to as the window–to–viewport transformation or the windowing transformation.

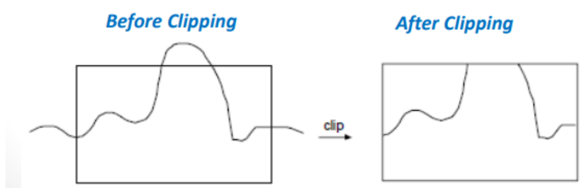
The transformation that maps the window into the viewport is applied to all of the output primitives (lines, rectangles, circles) in world coordinates.

Clipping:

The process of identifying those portions of a picture that are either inside or outside of the specified region of space is called clipping.

Two possible ways to apply clipping in the viewing transformation:

- Apply clipping in the world coordinate system: ignore objects that lie outside of the window.
- Apply clipping in the device coordinate system: ignore objects that lie outside of the viewport.



- Excludes unwanted graphics from the screen.
- Improves efficiency, as the computation dedicated to objects that appear off screen can be significantly reduced.

Applications:

- In drawing and painting, it is used to select picture parts for copying, erasing, moving.
- Identifying visible surfaces in 3-dimensional views
- Extracting part of a defined scene for viewing
- Creating objects using solid modeling procedures
- Displaying a multiwindow environment

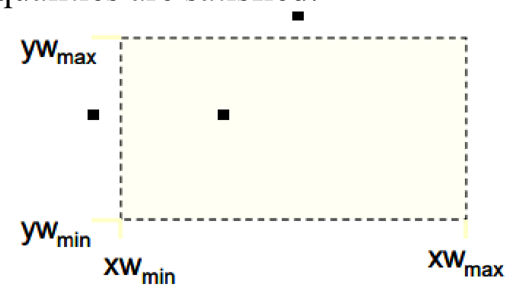
Some Primitive Types of Clipping:

- Point Clipping
- Line Clipping (Straight line segment)
- Area Clipping (Polygon)
- Curve Clipping
- Text clipping

Point Clipping:

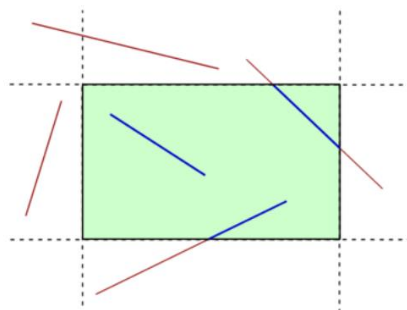
Let W denote a clip window with coordinates (XWmin, YWmin), (XWmin, YWmax), (XWmax, YWmin), (XWmax, YWmax), then a vertex (x, y) is displayed only if following "point clipping" inequalities are satisfied:

$$xw_{min} \leq x \leq xw_{max}$$
$$yw_{min} \leq y \leq yw_{max}$$



Line Clipping:

Lines that do not intersect the clipping window are either completely inside the window or completely outside the window. On the other hand, a line that intersects the clipping window is divided by the intersection point(s) into segments that are either inside or outside the window.



For a line segment with endpoints (x_1,y_1) and (x_2,y_2) and one or both endpoints outside the clipping rectangle, the parametric representation

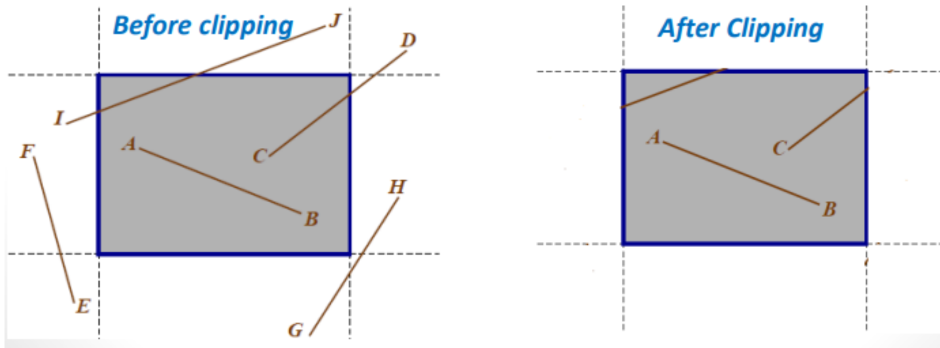
$$x=x_1 + u(x_2- x_1)$$
$$y=y_1 + u(y_2- y_1) \qquad 0\leq u \leq 1$$

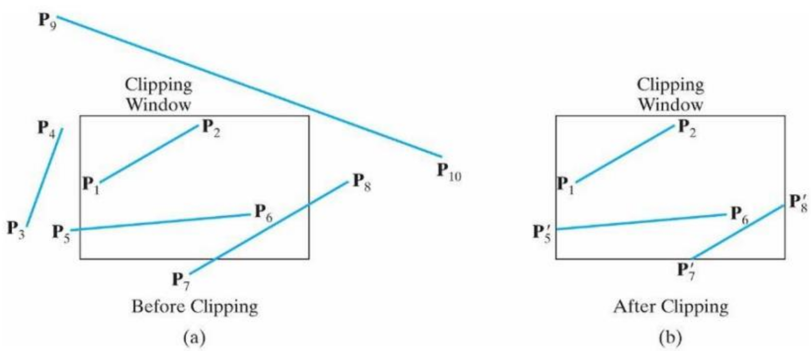
- ➢ If value of u for an intersection with a rectangle boundary edge is outside the range 0→1, the line does not enter the interior of the window at that boundary
- ➢ If value of u is within the range 0→1, the line segment does indeed enter into the clipping area

For any particular line segment:

a. Both endpoints are inside the region (line AB).
    - No clipping necessary.
b. One endpoint is inside and one is outside of the clipping window (line CD).
    - Clip at intersection point.
c. Both endpoints are outside the region:
    - No intersection (lines EF, GH)
    - Discard the line segment.

- Line intersects the region (line IJ)
- Clip line at both intersection points.

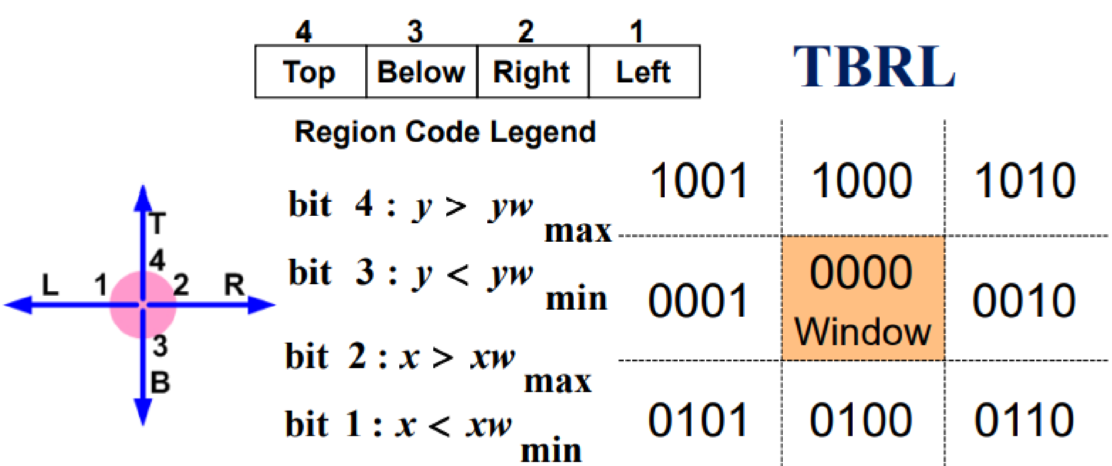




Cohen-Sutherland Line Clipping Algorithm

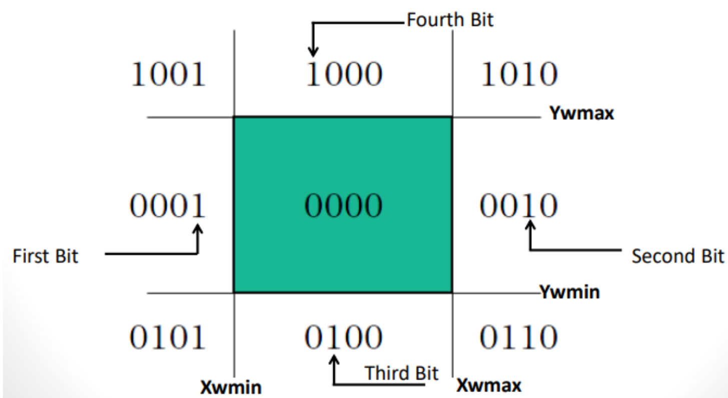
One of the oldest and most popular line-clipping algorithms. In this method, World space is divided into nine regions based on the window boundaries. All regions have their associated region codes.

Every line endpoint is assigned a four digit binary code (region code or out code). Each bit in the code is set to either a 1(true) or a 0(false). Assign a bit pattern to each region as shown:



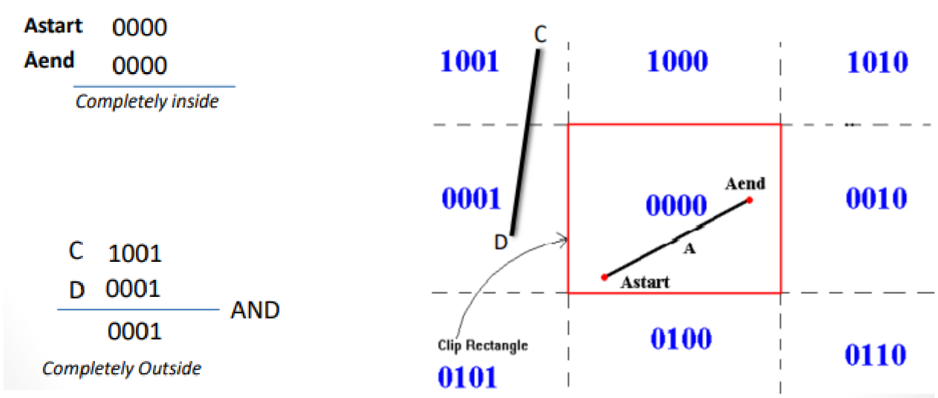
Step 1: Establish Region code for all line end point

First bit is 1, if x < Xwmin	(Point lies to left of window) , else set it to 0
Second bit is 1, if x > Xwmax	(Point lies to right of window) , else set it to 0
Third bit is 1, if y < Ywmin	(Point lies to below window), else set it to 0
Fourth bit is 1, if y > Ywmax	(Point lies to above window) , else set it to 0



Step 2: Determine which lines are completely inside window and which are not

a. If both end points of line has region codes ‘0000’ line is completely inside window.
b. If logical AND operation of region codes of two end points is NOT ‘0000’. The line is completely outside (some bit position have 1’s)



Step 3: If both tests fail then line is partially visible so we need to find the intersection with boundaries of window.

a) If 1st bit is 1 then line intersect with **Left** boundary and

- $Y_i = Y_1 + m\,(X - X_1)$ *where* X= Xwmin

b) If 2[nd] bit is 1 then line intersect with **Right** boundary and

- $Y_i = Y_1 + m\,(X - X_1)$ *where* X= Xwmax

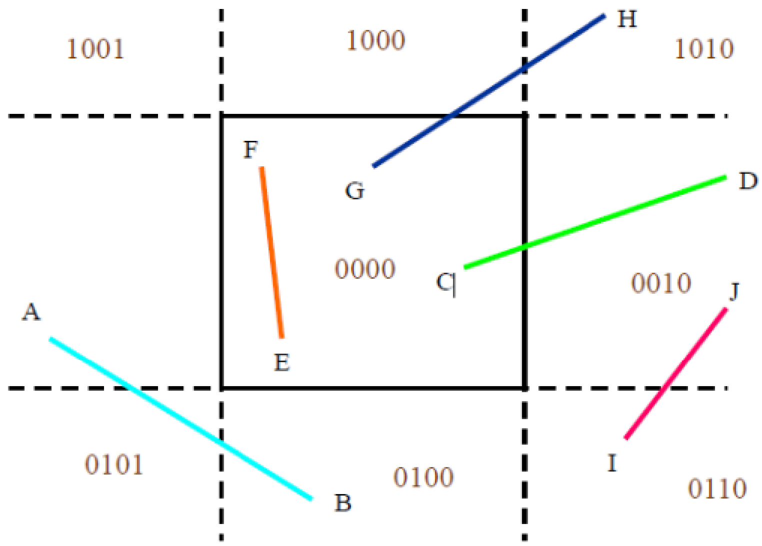
c) If 3[rd] bit is 1 then line intersect with **Bottom** boundary and

- $X_i = X_1 + (1/m)\,(Y - Y_1)$ *where* Y= Ywmin

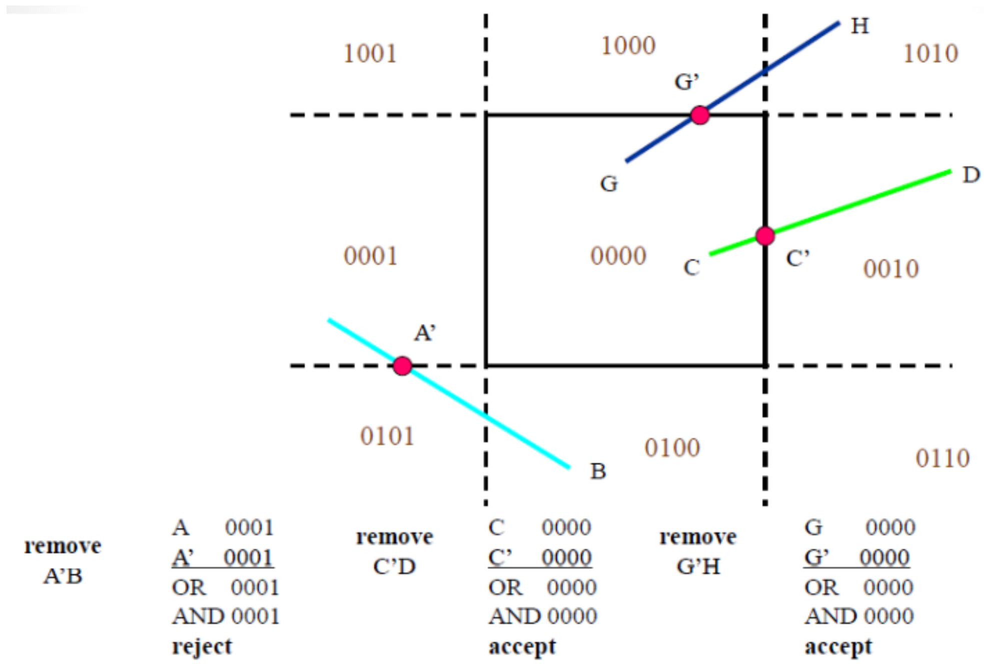
d) If 4[th] bit is 1 then line intersect with **Top** boundary and

- $X_i = X_1 + (1/m)\,(Y - Y_1)$ *where* Y= Ywmax

Here, (Xi , Yi) are (X ,Y) intercepts for that the step 4 repeat step 1 through step 3 until line is completely accepted or rejected



A 0001	C 0000	E 0000	G 0000	I 0110
B 0100	D 0010	F 0000	H 1010	J 0010
OR 0101	OR 0010	OR 0000	OR 1010	OR 0110
AND 0000	AND 0000	AND 0000	AND 0000	AND 0010
clip	**clip**	**accept**	**clip**	**reject**

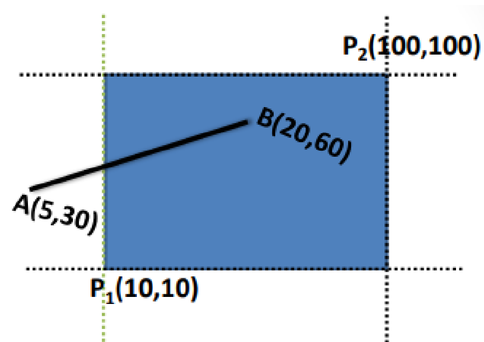


Q. Clip a line with end point A (5, 30), B (20, 60) against a clip window with lower most left corner at P1 (10, 10) and upper right corner at P2 (100, 100)

Answer:

Step 1: Establish the region codes for end point A,B

A(5,30)	B(20,60)
5<10 : (true)= 1	20<10: (false) =0
5>100 : (false) =0	20>100: (false)=0
30<10 : (false) =0	60<10 : (false)=0
30>100 : (false) =0	60>100: (false)=0
A(5,30) = 0001	***B(20,60)=0000***



Step 2: Visibility check

A: 0001
AND B: 0000
0000

Partial Visibility

Step 3: Intersection point with boundary

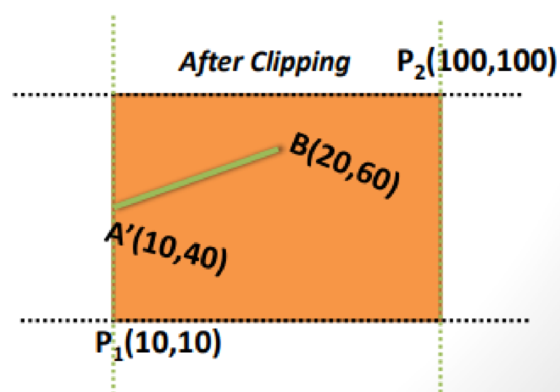
A: 0001 , Cond^n of 1$^{st\ bit}$ is 1

$Y_i = Y_1 + m (X - X_1)$ *where*, X= 10

= 30 + 2 (10-5)

= 40

(X,Y)=(10,40)



Q. Given a clipping window P(30, 0), S(0, 20) use the Cohen Sutherland algorithm to determine the visible portion of the line A(10, 30) and B(40, 0).

Q. Clip a line with end point A(1, 60) , B(20, 120) against a clip window with lower most left corner at P1(10,10) and upper right corner at P2(100, 100)

Liang-Barsky Line Clipping Algorithm:

Based on parametric equation of a line:

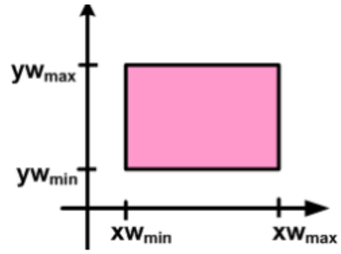
$$x = x_1 + u.\triangle x$$

$$y = y_1 + u.\triangle y$$

Similarly, the clipping window is represented by:

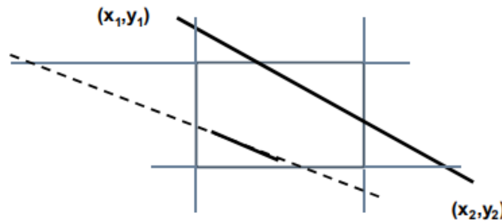
$$xw_{min} \le x_1 + u.\triangle x \le xw_{max} \qquad 0 \le u \le 1$$

$$yw_{min} \le y_1 + u.\triangle y \le yw_{max}$$



... or,

$$u.\ p_k \le q_k \qquad k = 1, 2, 3, 4$$



where:

k = 1 (is the line inside left boundary ?) $p_1 = -\Delta x$, $q_1 = x_1 - xw_{min}$

k = 2 (is the line inside right boundary ?) $p_2 = \Delta x$, $q_2 = xw_{max} - x_1$

k = 3 (is the line inside bottom boundary ?) $p_3 = -\Delta y$, $q_3 = y_1 - yw_{min}$

k = 4 (is the line inside top boundary ?) $p_4 = \Delta y$, $q_4 = yw_{max} - y_1$

$P_k<0$ → infinite extension of the line proceeds from outside to inside the infinitely extended boundary

$P_k>0$ → infinite extension of the line proceeds from inside to outside of the infinitely extended boundary

- **Trivial rejection :**
  Reject line with $p_k = 0$ for some k and one $q_k < 0$ for these k.

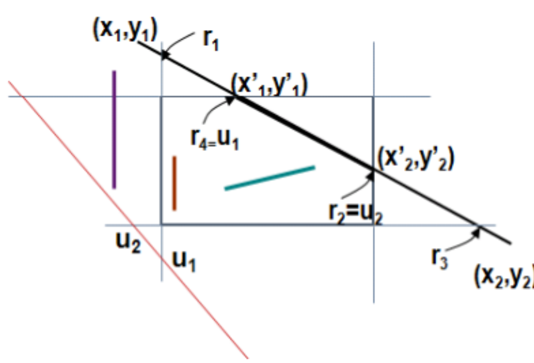
- clipped line will be

$$x_1' = x_1 + u_1.\triangle x \qquad u_1 \ge 0$$

$$y_1' = y_1 + u_1.\triangle y$$

$$x_2' = x_1 + u_2.\triangle x \qquad u_2 \le 1$$

$$y_2' = y_1 + u_2.\triangle y$$



- Calculate

$$u_k = \frac{q_k}{p_k}$$

$\mathbf{u_1}$ --(For intersection with the boundaries to which line enters the boundary)
→**maximum value between 0 and u** (for $p_k < 0$), where starting value of $\mathbf{u_1=0}$

$\mathbf{u_2}$ --(For intersection with the boundaries to which line leaves the boundary)
→**minimum value between u and 1** (for $p_k > 0$), where starting value of $\mathbf{u_2=1}$

Algorithm:

Step1) Accept line end points A(x1,y1) ,B(x2,y2) and window coordinates Xwmin, Ywmin, Xwmax,Ywmax as input.

Step2) Calculate p_k and q_k for k=1,2,3,4 as

$$p1 = -\Delta x \qquad q1 = x1 - Xw\min$$
$$p2 = \Delta x \qquad q2 = Xw\max - x1$$
$$p3 = -\Delta y \qquad q3 = y1 - Yw\min$$
$$p4 = \Delta y \qquad q4 = Yw\max - y1$$

Step3) if pk=0 then line is parallel to k^{th} boundary.

if qk<0 then line is outside the boundary discard the line and stop.

if qk$\geq$0 then line is inside the parallel boundary, calculate intersection points.

if pk$\neq$0 then go to step 4.

Step4) Calculate $rk = \dfrac{qk}{pk}$ for k=1,2,3,4

Step5) Determine u1 for pk<0 by selecting relative rk as u1={rk,0}max

and u2 for pk>0 by selecting rk as u2={rk,1}min

Step6) if u1>u2 then line is totally outside discard and stop.

Step7) if u1<u2 then calculate end points of clipped line.

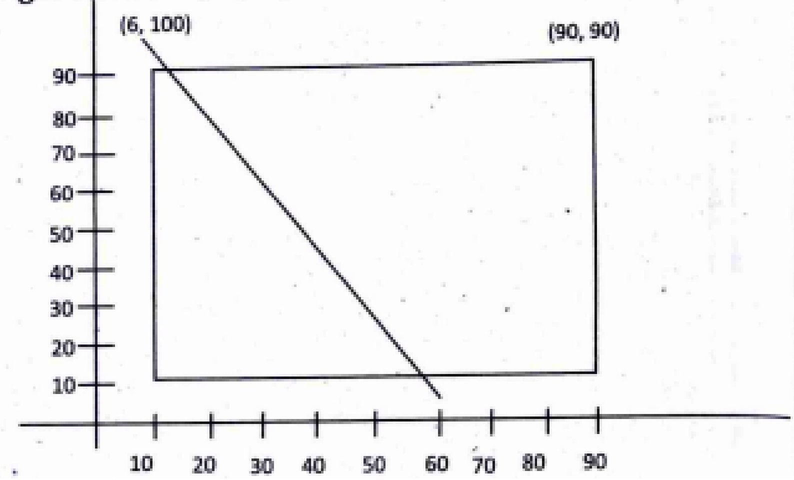
$$x' = x1 + u1\Delta x \qquad y' = y1 + u1\Delta y \qquad I'(x', y')$$
$$x'' = x1 + u2\Delta x \qquad y'' = y1 + u2\Delta y \qquad I''(x'', y'')$$

Step8) Display line segment from

$$I'I''$$

Q.

Use Liang Barsky line clipping algorithm to clip a line starting from (6, 100) and ending at (60, 5) against the window having its lower left corner at (10, 10) and upper right corner at (90, 90). *[2075 Ashwin]*



Solution:

Clipping window $(x_{min}, y_{min}) = (10, 10)$

$(x_{max}, y_{max}) = (90, 90)$

$(x_1, y_1) = (6, 100)$ and $(x_2, y_2) = (60, 5)$

$\Delta x = x_2 - x_1 = 60 - 6 = 54$

$\Delta y = y_2 - y_1 = 5 - 100 = -95$

K	p_k	q_k	$r_k = q_k/p_k$
0	$-\Delta x = -54 (p_k<0)$	$x_1 - x_{min} = 6-10 = -4$	$0.074(u_1)$
1	$\Delta x = 54$	$x_{max} - x_1 = 90-6 = 84$	$1.55(u_2)$
2	$-\Delta y = 95$	$y_1 - y_{min} = 100-10 = 90$	$0.947(u_2)$
3	$\Delta y = -95$	$y_{max} - y_1 = 90-100 = -10$	$0.105(u_1)$

$u_1 = \max(0, r_k) = 0.105$

$u_2 = \min(0, r_k) = 0.947$

$x_1' = x_1 + u_1\Delta x = 6 + 0.105 \times 54 = 11.67 \approx 12$

$y_1' = y_1 + u_1\Delta y = 100 + 0.105 \times (-94) = 90.025 \approx 90$

$x_2' = x_1 + u_2\Delta x = 6 + 0.947 \times 54 = 57.118 \approx 57$

$y_2' = y_1 + u_2\Delta y = 100 + 0.947 \times (-94) = 10.035 \approx 10$

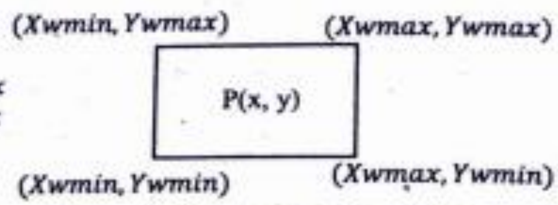
Required points are (12, 90) and (57, 10)

Q. Write down the condition for point clipping. Find the clipped region in window of diagonal vertex (10,10) and (100,100) for line P_1 (5,120) and P_2 (80,7) using Liang-Barsky line clipping method.

Solution:

Assuming that the clip window is rectangle in standard position, we save a point p(x, y) for display, if the following inequalities are satisfied.

$$Xwmin \le X \le Xwmax$$
$$Ywmin \le Y \le Ywmax$$



The edges of the clip window can be either we window boundary or viewport boundaries.

If any one of these four inequalities is not satisfied, the point is clipped i.e., not saved for display.

x_{wmin} = 10

y_{wmin} = 10

x_{wmax} = 100

y_{wmax} = 100

(x_1, y_1) = (5, 120)

(x_2, y_2) = (80, 7)

Now, we have to find out the value of u_1 and u_2 by calculating p_k, q_k, r_k from k = 1 to 4.

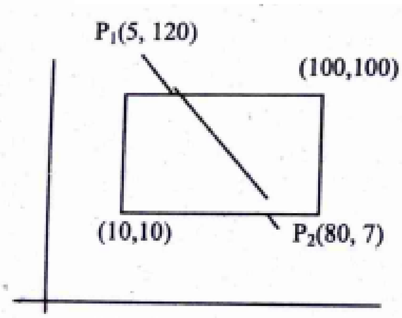
Then, the clipped line will be

$x_1' = x_1 + u_1\Delta x$

$y_1' = y_1 + u_1\Delta y$

$x_2' = x_1 + u_2\Delta x$

$y_2' = y_1 + u_2\Delta y$



K	p_k	q_k	$r_k = \frac{q_k}{p_k}$
1	$-\Delta x = -(80-5) = -75$ i.e., $p_k<0$	$x_1 - x_{wmin}$ $(5-10) = -5$	$r_1 = \frac{-5}{-75} = \frac{1}{15}$ (u_1)
2	Δx $= -(80-5)$ $= 75$ i.e., $p_k>0$	$x_{wmax} - x_1$ $= 100 - 5$ $= 95$	$r_2 = \frac{95}{75}$ (u_2)
3	Δy $= -(7-120)$ $= 113$ i.e., $p_k>0$	$y_1 - y_{wmm}$ $= (120-10)$ $= 110$	$r_3 = \frac{110}{113}$ (u_2)
	Δy $= -113$ i.e., $p_k<0$	$y_{wmax} - y_1$ $= 100 - 120$ $= -20$	$r_4 = \frac{-20}{-113} = \frac{20}{113}$ (u_1)

Now,

$$u_1 = \max(0, r_k)$$

$$= \max(0, \frac{1}{15}, \frac{20}{113}) = \frac{20}{113}$$

$$u_2 = \min(1, r_k)$$

$$= \min(1, \frac{95}{75}, \frac{110}{113}) = \frac{110}{113}$$

$$x_1' = x_1 + u_1\Delta x$$

$$= 5 + \frac{20}{113} \times 75 = 18.27$$

$$y_1' = y_1 + u_1\Delta y$$

$$= 120 + \frac{20}{113} \times (-113) = 100$$

$$x_2' = x_1 + u_2\Delta x$$

$$= 5 + \frac{110}{113} \times 75 = 78$$

$$y_2' = y_1 + u_2\Delta y$$

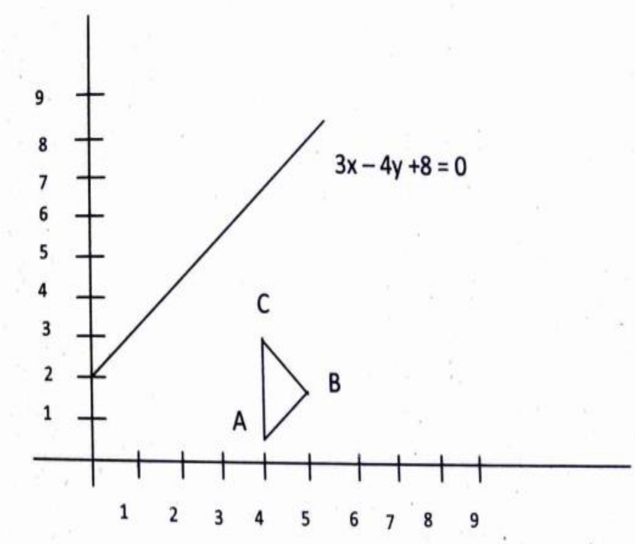
$$= 120 + \frac{110}{113} \times (-113) = 10$$

$$P_1'(x_1', y_1') = P_1'(18{,}27{,}100)$$

$$P_2'(x_2', y_2') = P_2'(78, 10)$$

Q. Reflect the triangle ABC about the line 3x – 4y + 8 = 0 the position vector of coordinate ABC as A(4, 1), B(5, 2) and C(4,3).

Solution:



The arbitrary line about which the triangle ABC has to be reflected is $3x - 4y + 8 = 0$

i.e., $y = \frac{3}{4}x + 2$

$m = \frac{3}{4}$

$c = 2$

$\theta = \tan^{-1}(m) = \tan^{-1}\frac{3}{4} = 36.8698^{\circ}$

C.M. $= T^{-1}R^{-1}_{\theta}R_{fx}R_{\theta}T$

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

$$R_{\theta} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos(36.87) & \sin(36.87) & 0 \\ -\sin(36.87) & \cos(36.87) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

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

$$= \begin{bmatrix} \cos(-36.87) & \sin(-36.87) & 0 \\ -\sin(-36.87) & \cos(-36.87) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$T^{-1} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\text{C.M.} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.8 & -0.6 & 0 \\ 0.6 & 0.8 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
$$\begin{bmatrix} 0.8 & 0.6 & 0 \\ -0.6 & 0.8 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \text{C.M.} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Q. Clip the line P1P2 with P1(-5,3) and P2(15,9) with clip window having diagonal coordinate (0,0) and (10,10) using Liang-Barskey line clipping method.

Q. State the conditions of point clipping. Perform clipping operation for the following using Liang Barskey line clipping algotithm:

Clipping window: (Xmin, Ymin) = (2,5) and (Xmin, Ymin) = (35,50)

Line: (x1, y1) = (-2,2) and (x2,y2) = (45,40)