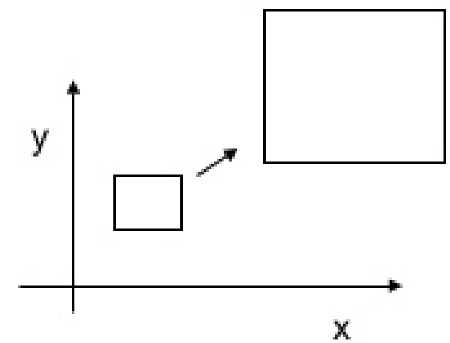
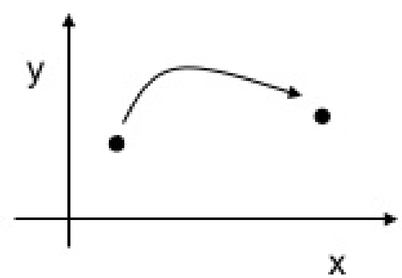
Chapter 3

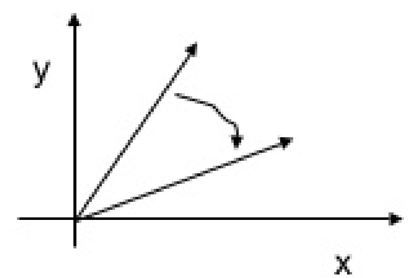
Two Dimensional Transformation

Basic Transformation

- Transformation means changing the object by changing position, orientation or size of original object by applying certain rules.
- The basic transformations are
  - Translation/Shifting
  - Scaling
  - Rotation







Translation

- Translation means repositioning an object along a straight line path from one coordinate location to another
- We add translational distance t_x ,t_y to original coordinate position (x,y) to move the point to a new position (x',y')

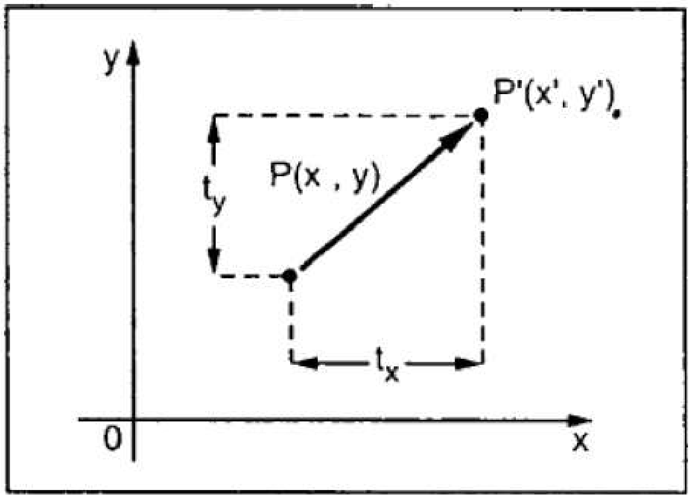
$$x' = x + t_x$$

$$y' = y + t_y$$

where the pair (tx , ty) is called the *translation vector.*

- We can write equation as a single matrix equation by using column vectors to represent coordinate points and translation vectors. i.e.

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



- So, we can write

$$\mathbf{P' = P + T}$$

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

Scaling

- Scaling Transformation alters the size of an object i.e we can magnify and reduce the size of an object
- In case of polygons scaling is done by multiplying coordinate values (x, y) of each vertex by scaling factors s_x, s_y to produce the final transformed coordinates (x', y').
- s_x scales object in 'x' direction and s_y scales object in 'y' direction
- We can represent this in equation as

$$x' = x \cdot s_x \quad \text{and} \quad y' = y \cdot s_y$$

- We can also represent in matrix for as

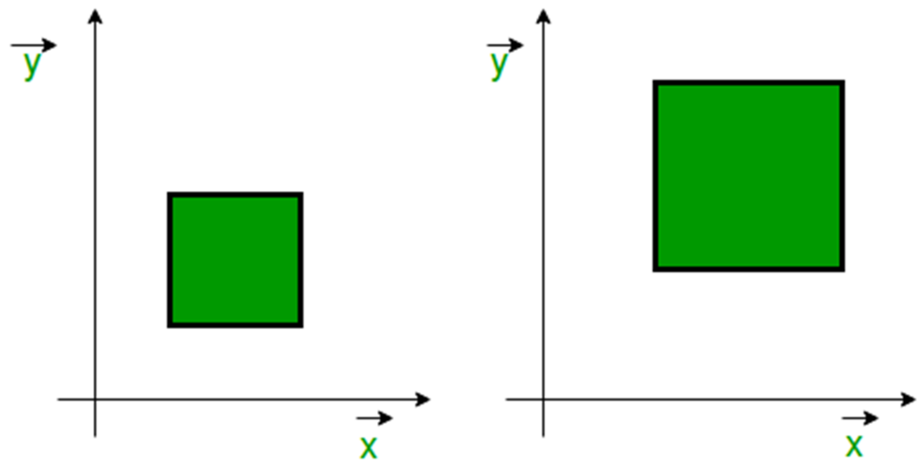
$$\mathbf{P} = \begin{bmatrix} x \\ y \end{bmatrix} \qquad \mathbf{T} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \qquad \mathbf{P'} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

or

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

or

$$\mathbf{P'} = \mathbf{S} \cdot \mathbf{P}$$



- Values greater than 1 for s_x, s_y produce *enlargement*
- Values smaller than 1 for s_x, s_y *reduce* size of object
- $s_x = s_y = 1$ leaves the size of the object *unchanged*
- If $s_x = s_y$ then a *Uniform Scaling* is produced else *Differential Scaling* is produced

Rotation

- Rotation repositions an object along a circular path in xy plane
- To generate a rotation, we specify a rotation angle θ and the position (x_r, y_r) of rotation point about which the object is to be rotated.
- + value for 'θ' define *counter-clockwise* rotation about a point
- - value for 'θ' define *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 the point (x,y) is rotated through angle 'θ' in a counter clock wise direction
- Express the transformed coordinates in terms of 'Φ' and 'θ' as

$$x' = r\cos(\Phi + \theta) = r\cos\Phi.\cos\theta - r\sin\Phi.\sin\theta \quad ....(i)$$

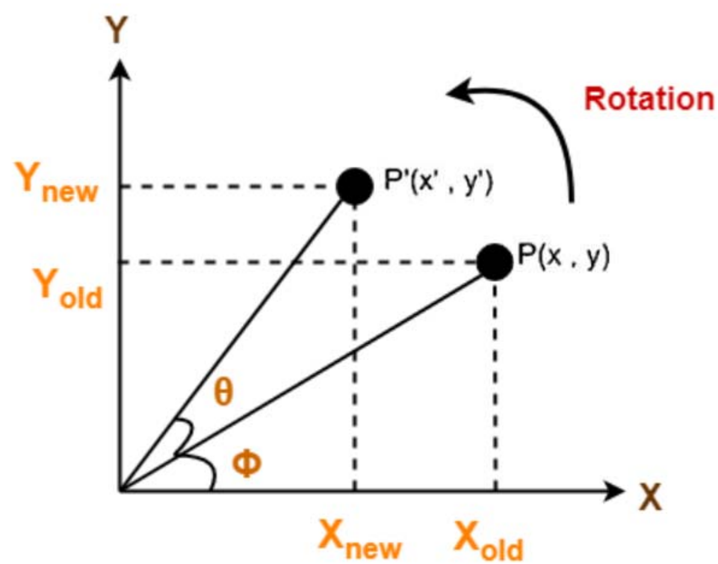
$$y' = r\sin(\Phi + \theta) = r\cos\Phi.\sin\theta + r\sin\Phi.\cos\theta \quad ....(ii)$$

- We know that original coordinates of point in polar coordinates are

$$x = r\cos\Phi \text{ and } y = r\sin\Phi$$

- substituting these values in (i) and (ii)
- we get,

$$x' = x\cos\theta - y\sin\theta \qquad \text{and} \qquad y' = x\sin\theta + y\cos\theta$$



Rotation

- So, using column vector representation for coordinate points the matrix form would be

$$P' = R \cdot P$$

where

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

So,

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

If rotation is in clockwise direction, we take negative angle

So,

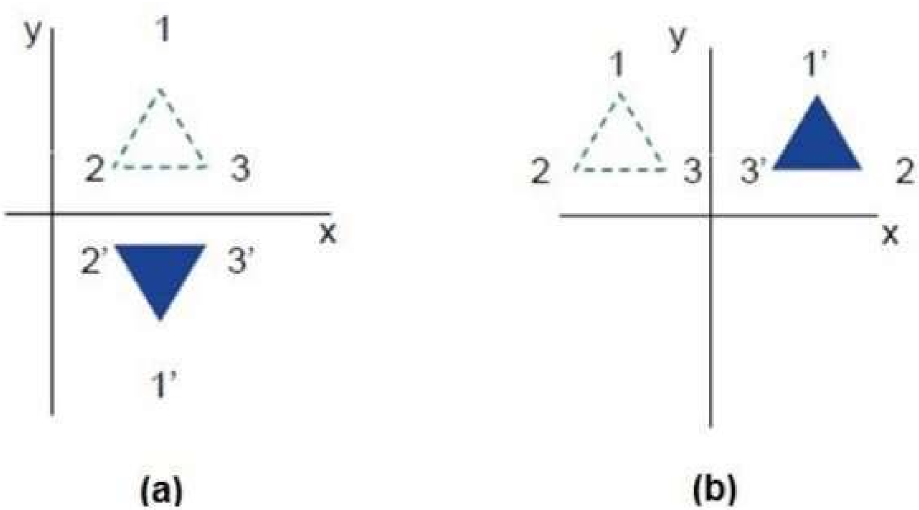
$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

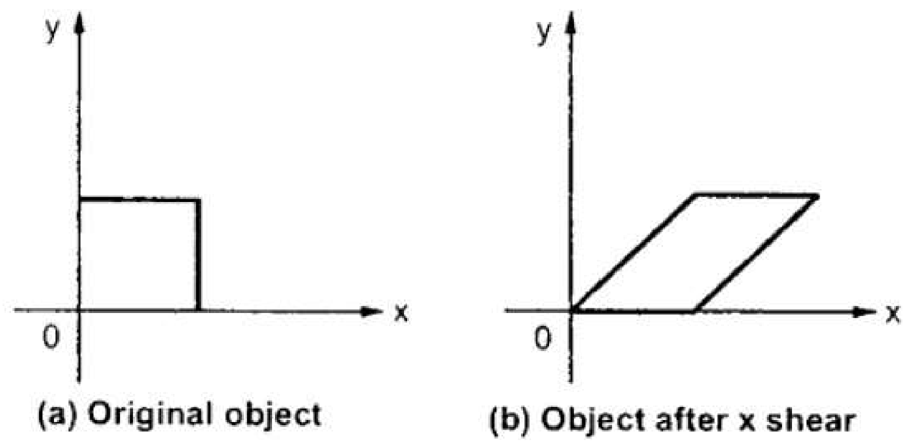
and

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

Other Transformation

- Besides basic transformations, we have other two transformation. i.e
  - Reflection
  - Shearing





Reflection

- Reflection is a transformation that produces a mirror image of an object
- Mirror image for 2D reflection is generated relative to an axis of reflection by rotating the object 180 degree about the reflection axis.

Reflection about x-axis or about line y=0

- Keeps 'x' value same but flips y value of coordinate points

So $x' = x$ and $y' = -y$

i.e.

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

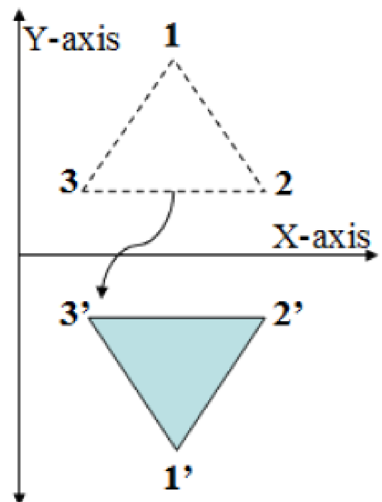


Fig. Reflection of an object about the x axis.

Reflection

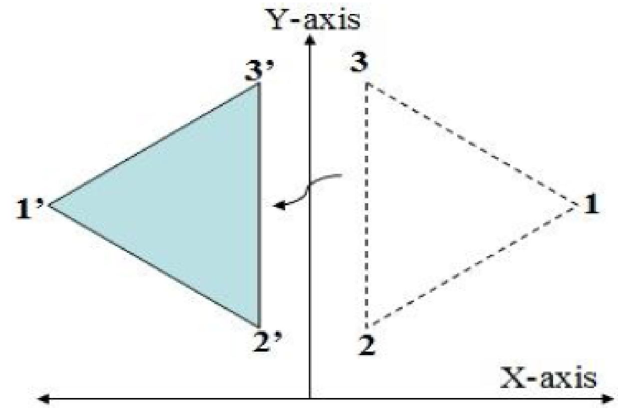
Reflection about y-axis or about line x=0

- Keeps y value same but flips x value of coordinate points

So $x' = -x$ and $y' = y$

i.e.

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



Reflection of an object about the Y axis (X=0).

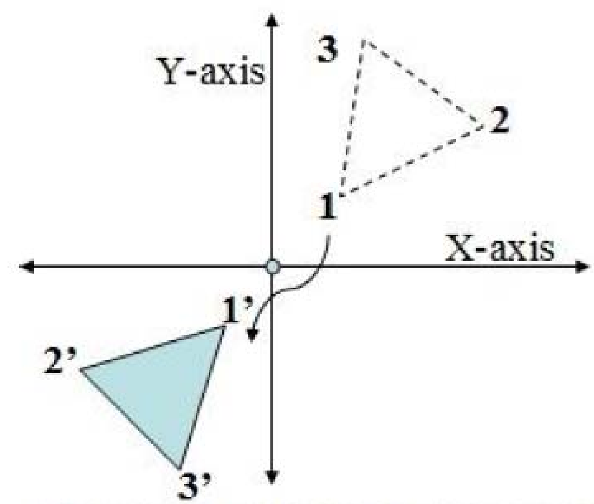
Reflection about origin

- Flip both x and y vale

So $x' = -x$ and $y' = -y$

i.e.

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



Reflection of an object about origin

Reflection

Reflection about y=x

Steps required:

i. Rotate about origin in clockwise direction by 45 degree

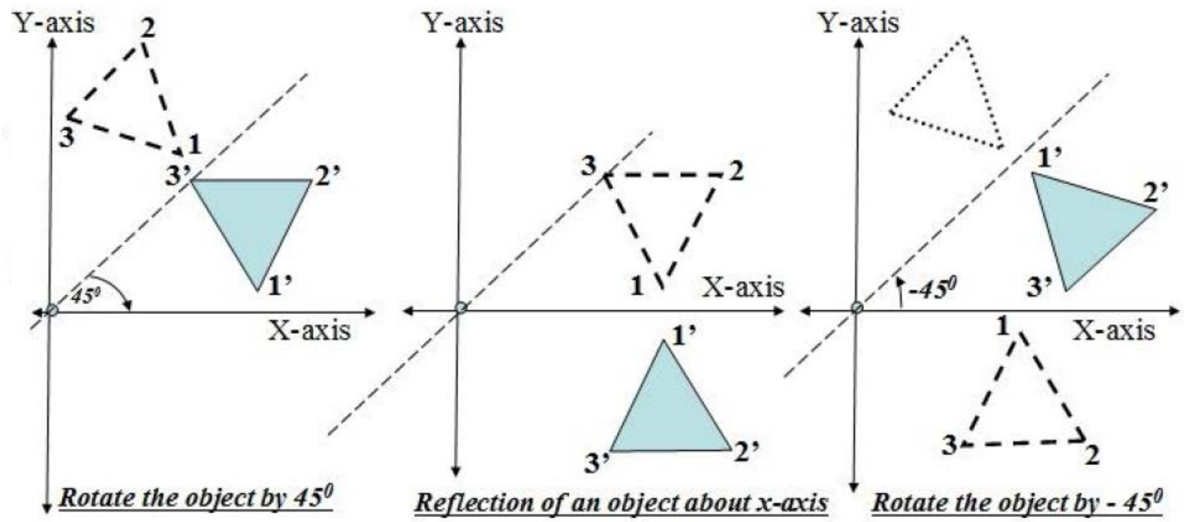
$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

ii. Take reflection against x-axis

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

iii. Rotate in anti-clockwise direction by same angle

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



Reflection

Reflection about y=x

We have $\theta = 45$

Solving all these steps, we get final result

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

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

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

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

Reflection

Reflection about y=-x

Steps required:

i. Rotate about origin in clockwise direction by 45 degree

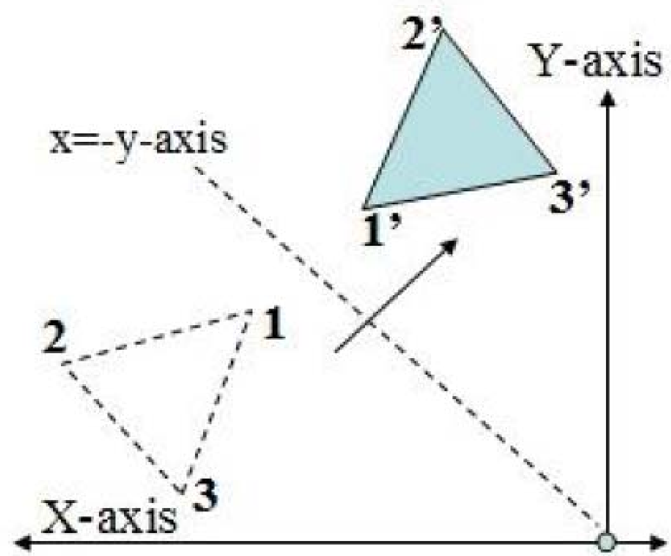
$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

ii. Take reflection against y-axis

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

iii. Rotate in anti-clockwise direction by same angle

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



Reflection of an object about x= -y in clockwise direction

Reflection

Reflection about y=-x

We have θ = 45

Solving all these steps, we get final result

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

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

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

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

Shearing

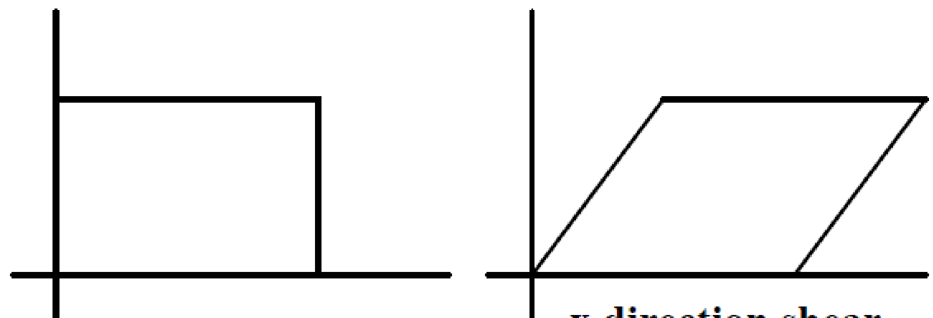
- Shearing distorts the shape of an object in either x or y or both direction
- In case of single directional shearing (e.g. in ‘x’ direction can be viewed as an object made up of very thin layer and slid over each other with the *base* remaining where it is).

in ‘x’ direction,

$x' = x + s_{hx} \cdot y$

$y' = y$

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



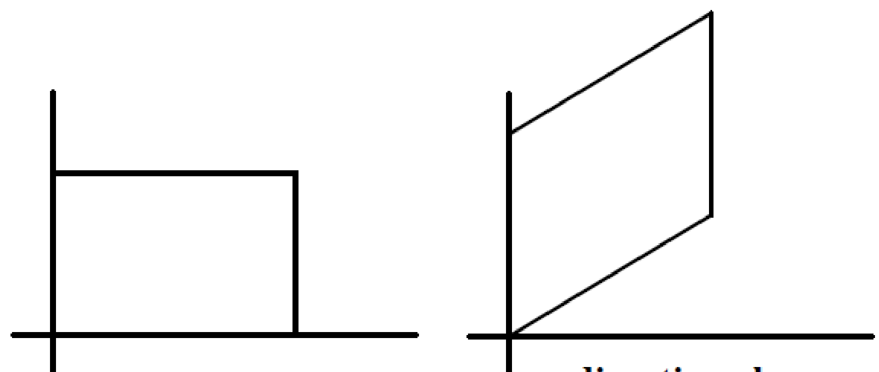
x direction shear

in ‘y’ direction,

$x' = x$

$y' = y + s_{hy} \cdot x$

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



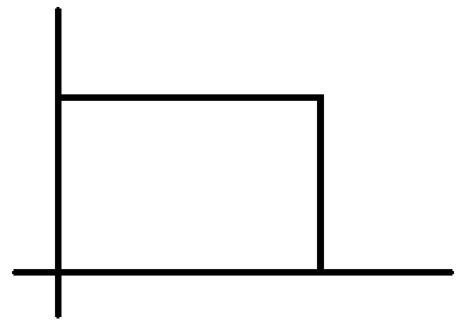
y direction shear

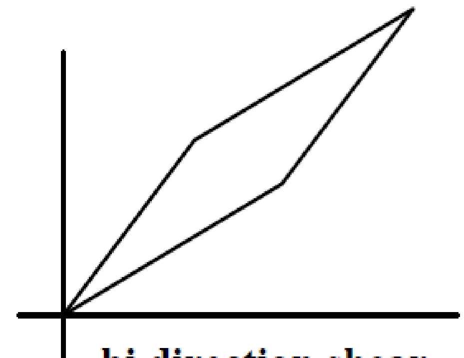
Shearing

in both directions,

$$x' = x + s_{hx} \cdot y$$
$$y' = y + s_{hy} \cdot x$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & s_{hx} \\ s_{hy} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$





bi direction shear

Rotation of a point about an arbitrary pivot position can be seen in the figure.

Here,

$$x' = x_r + (x-x_r)\cos\theta \; - \; (y- y_r)\sin\theta$$
$$y' = y_r + (x- x_r)\sin\theta \; + (y- y_r)\cos\theta$$

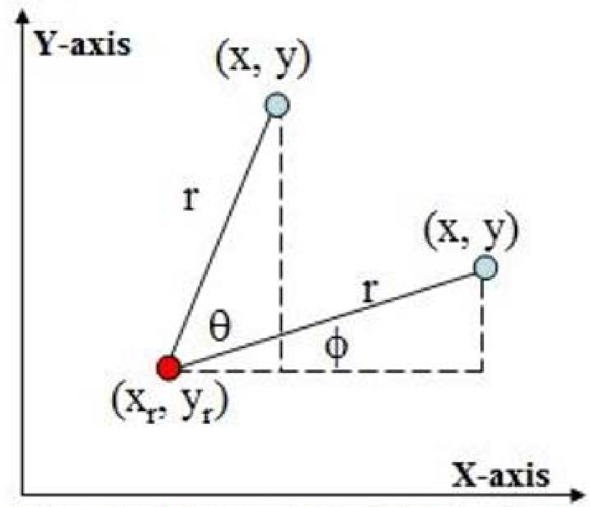
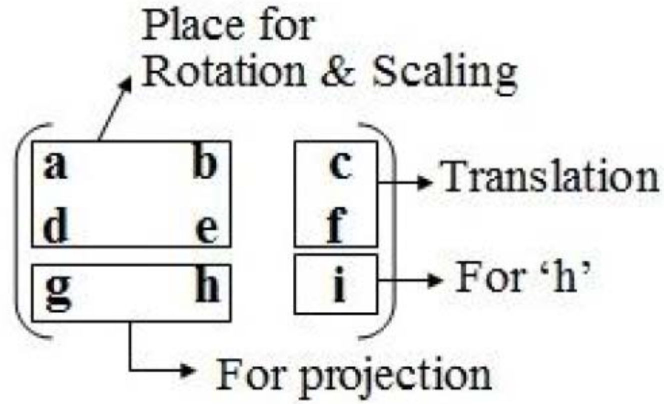


Fig. Rotating a point from position (x, y) to position (x', y') through an angle θ about rotation point (x_r, y_r).

Note: - This can also be achieved by translating the arbitrary point into the origin and then apply the rotation and finally perform the reverse translation.

Homogeneous Coordinate



- The matrix representations for translation, scaling and rotation are respectively:
  1. Translation: **P' = T + P** (*Addition*)
  2. Scaling: **P' = S . P** (*Multiplication*)
  3. Rotation: **P' = R . P** (*Multiplication*)

- Since, the composite transformation include many sequence of translation, rotation etc and hence the many naturally different addition & multiplication sequence have to perform by the graphics allocation.

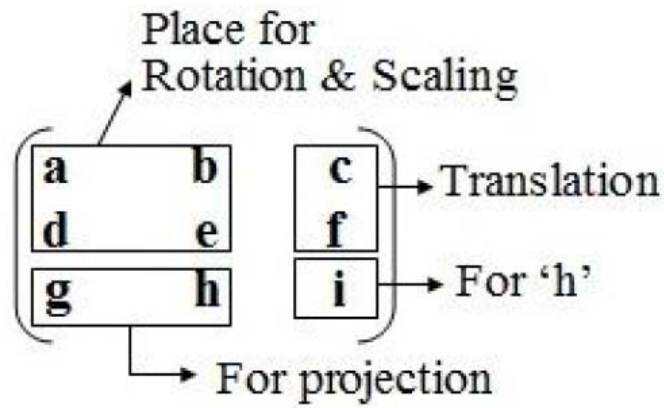
- Hence, the applications will take more time for rendering. Thus, we need to treat all three transformations in a consistent way so they can be combined easily & compute with one mathematical operation.

- If points are expressed in homogenous coordinates, all geometrical transformation equations can be represented as matrix multiplications.

- Here, in case of homogenous coordinates we add a third coordinate '*h*' to a point (x, y) so that each point is represented by (*h*x, *h*y, *h*). The '*h*' is normally set to 1. If the value of 'h' is more the one value then all the co-ordinate values are scaled by this value.

- Coordinates of a point are represented as three element column vectors, transformation operations are written as 3 x 3 matrices.

Homogeneous Coordinate



- **For Translation**

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

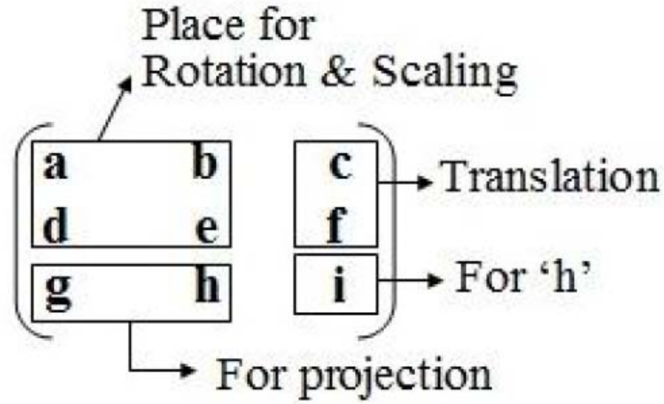
- **For rotation**

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

- **For Scaling**

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

Homogeneous Coordinate



- **For Reflection about X-axis**

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

- **For Reflection about Y-axis**

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

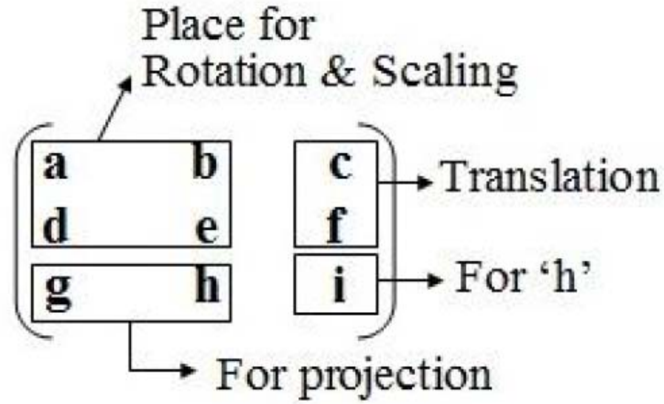
Homogeneous Coordinate

- **For Reflection about y=x**

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

- **For Reflection about y=-x**

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



Composite Transformation

- With the matrix representation of transformation equations it is possible to setup a matrix for any sequence of transformations as a composite transformation matrix by calculating the matrix product of individual transformation.
- Forming products of transformation matrices is often referred to as a **concatenation,** or **composition,** of matrices.
- For column matrix representation of coordinate positions we form composite transformation by multiplying matrices in order from **right to left.**

i. Two Successive Translation are Additive

Let two successive translation vectors (t_{x1} , t_{y1}) and (t_{x2} , t_{y2}) are applied to a coordinate position P then

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

Here the composite transformation matrix for this sequence of translation is

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

or, $$T(t_{x2}, t_{y2}) \cdot T(t_{x1}, t_{y1}) = T(t_{x1} + t_{x2}, t_{y1} + t_{y2})$$

Composite Transformation

ii. Two successive Scaling operations are Multiplicative

Let (s_{x1} , s_{y1}) and (s_{x2} , s_{y2}) be two successive vectors applied to a coordinate position P then the composite scaling matrix thus produced is

or, $P' = \{S(s_{x2}, s_{y2}) \cdot S(s_{x1}, s_{y1})\} \cdot P$

Here the composite transformation matrix for this sequence of translation is

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

or, $$S(s_{x2}, s_{y2}) \cdot S(s_{x1}, s_{y1}) = S(s_{x1} \cdot s_{x2}, s_{y1} \cdot s_{y2})$$

Composite Transformation

<u>iii. Two successive Rotation operations are Additive</u>

Let $R(\theta_1)$ and $R(\theta_2)$ be two successive rotations applied to a coordinate position P then the composite scaling matrix thus produced is

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

Here the composite transformation matrix for this sequence of translation is

or,
$$\begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

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

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

Composite Transformation

Reflection about y=x

Steps required:

i. Rotate about origin in clockwise direction by 45 degree $\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

ii. Take reflection against x-axis $\begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$

iii. Rotate in anti-clockwise direction by same angle $\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Representing these steps in composite matrix

$$\underbrace{\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Step 3}} \underbrace{\begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}}_{\text{Step 2}} \underbrace{\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Step 1}}$$

Composite Transformation

$$\underbrace{\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Step 3}} \underbrace{\begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}}_{\text{Step 2}} \underbrace{\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Step 1}}$$

Composite Transformation

Reflection about y=mx+c

Steps required:

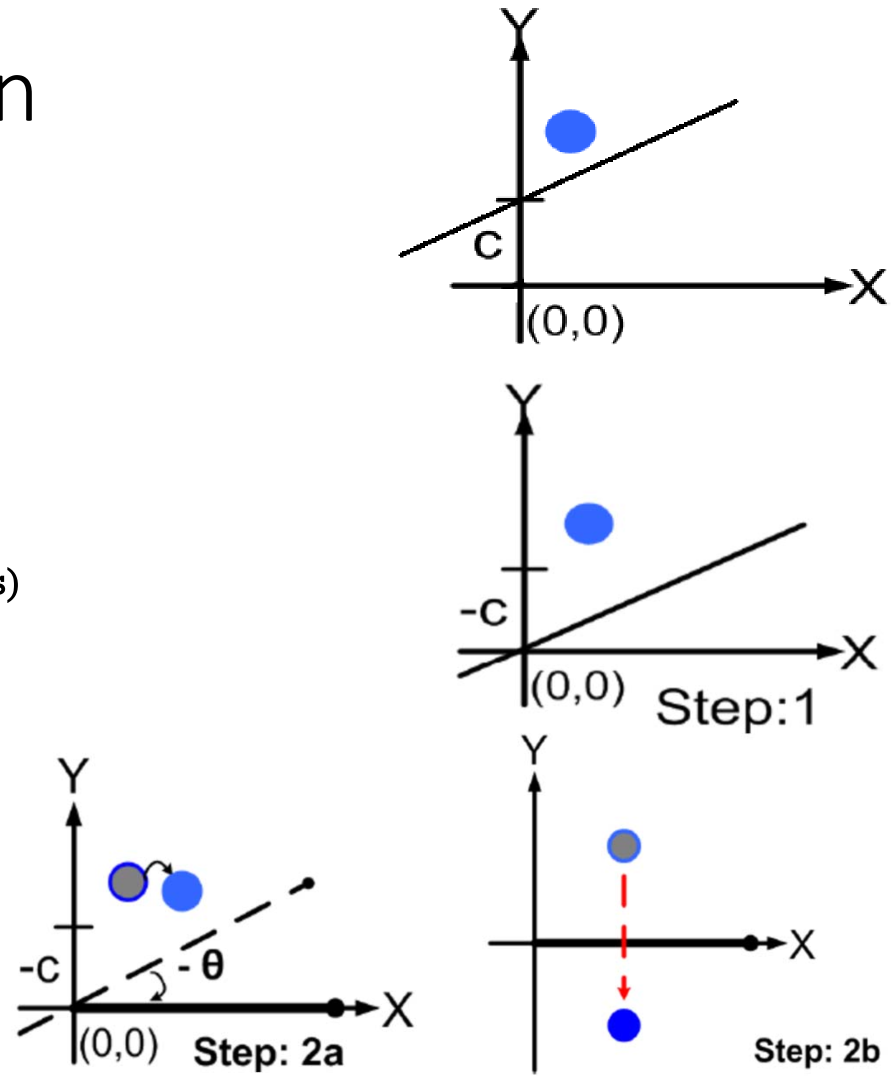
i. First translate the line so that it passes through the origin

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

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

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

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



Composite Transformation

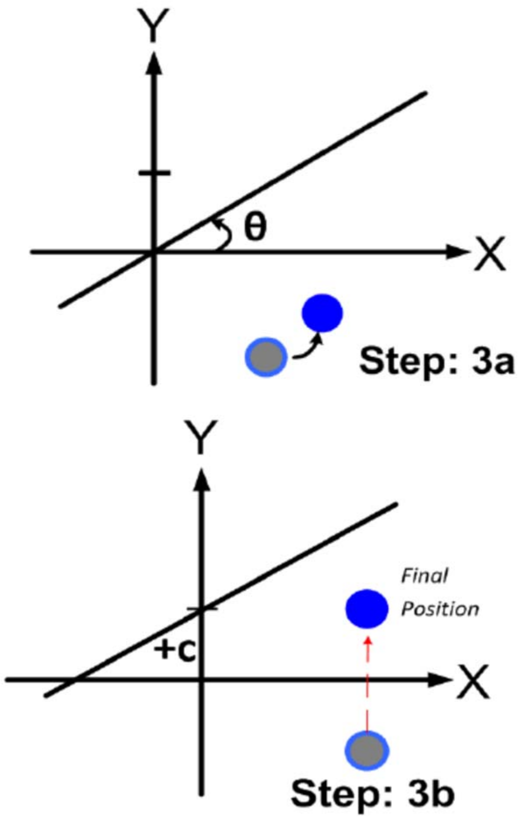
Reflection about y=mx+c

Steps required:

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

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

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & c \\ 0 & 0 & 1 \end{bmatrix} --- \textit{translation}$$



Composite Transformation

Reflection about y=mx+c

So, the multiplication sequence will be

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

Solving all these we will get

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

Since we have

$\tan\theta = m$

$\sec^2\theta = 1+\tan^2\theta$

$\sec\theta = \sqrt{1+\tan^2\theta}$

$\cos\theta = \frac{1}{\sec\theta} = \frac{1}{\sqrt{1+m^2}}$

$\tan\theta = \frac{\sin\theta}{\cos\theta}$

$\sin\theta = \tan\theta . \cos\theta = \frac{m}{\sqrt{1+m^2}}$

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

Solution:

$$C.M. = T_{(5,4)}.R_{90}T_{(-5,-4)}$$

$$= \begin{bmatrix} 1 & 0 & 5 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 90^0 & -\sin 90^0 & 0 \\ \sin 90^0 & \cos 90^0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -5 \\ 0 & 1 & -4 \\ 0 & 0 & 1 \end{bmatrix}$$

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

Now $P' = C.M.* P$

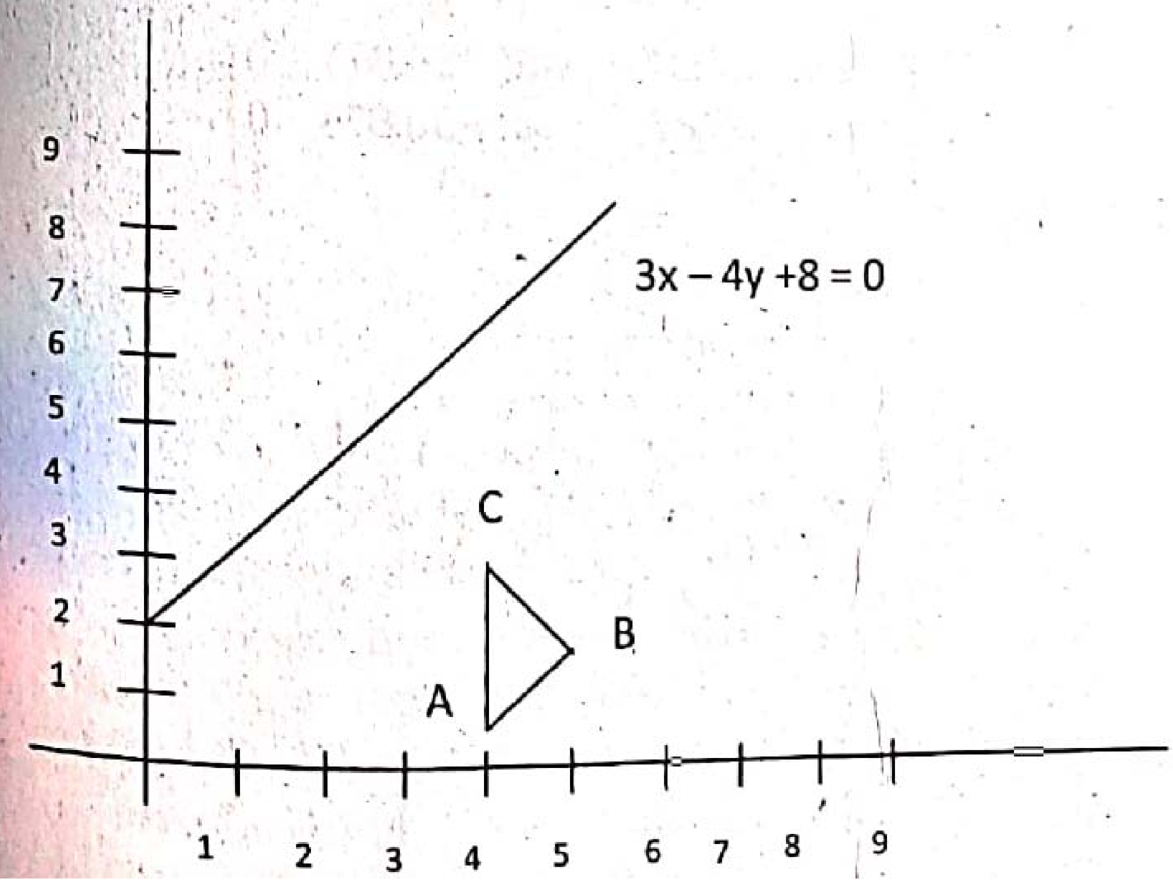
$$= \begin{bmatrix} 0 & -1 & 9 \\ 1 & 0 & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 5 & 7 & 3 \\ 5 & 3 & 3 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -5+9 & -3+9 & -3+9 \\ 8-1 & 7-1 & 3-1 \\ 1 & 1 & 1 \end{bmatrix}$$

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

$\therefore$ New co-ordinates are (4,4), (6,6) (6,2)

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.; $y = \frac{3}{4}x + 2$

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

$c = 2$

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

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

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

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

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

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

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

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

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

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

Two Dimensional Viewing

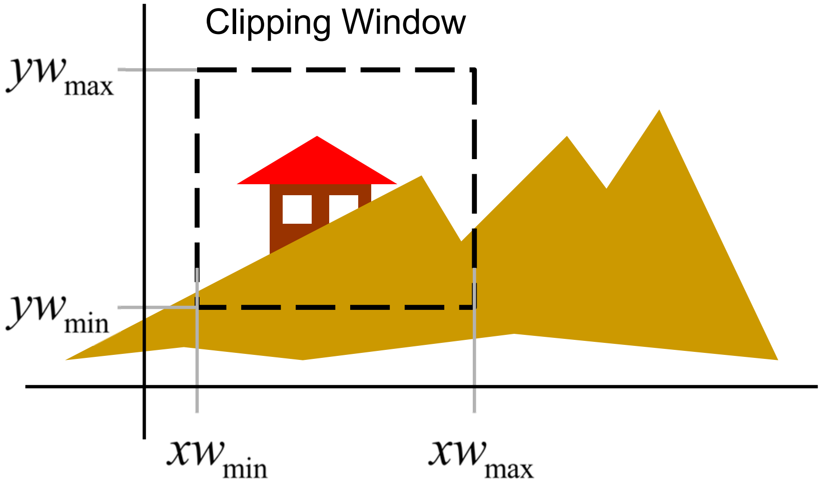
- **Window**
  - a world coordinate area selected for display
  - define what is to be viewed
- **View-port**
  - An area on a display device to which a window is mapped
  - Define where it is to be displayed
- **Windows and view-port**
  - Rectangle in standard positions, with rectangle edges parallel to coordinate axes
  - Other geometric takes longer to proceed
- **Viewing transformation**
  - the mapping of a world coordinate scene to device coordinates
- **Transformation pipeline**
  - Takes the object coordinates through several intermediate coordinates systems before finishing with device coordinates

Two Dimensional Viewing

Window in world coordinates.

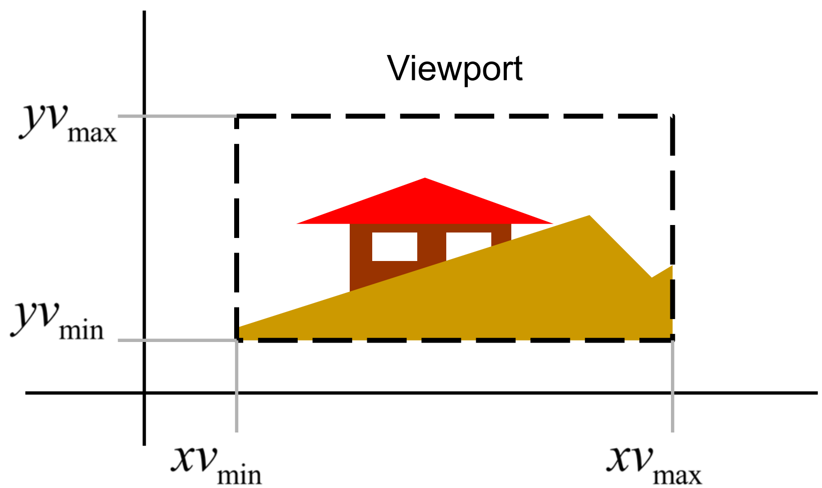
Viewport in
Device coordinates

Two Dimensional Viewing



World Coordinates

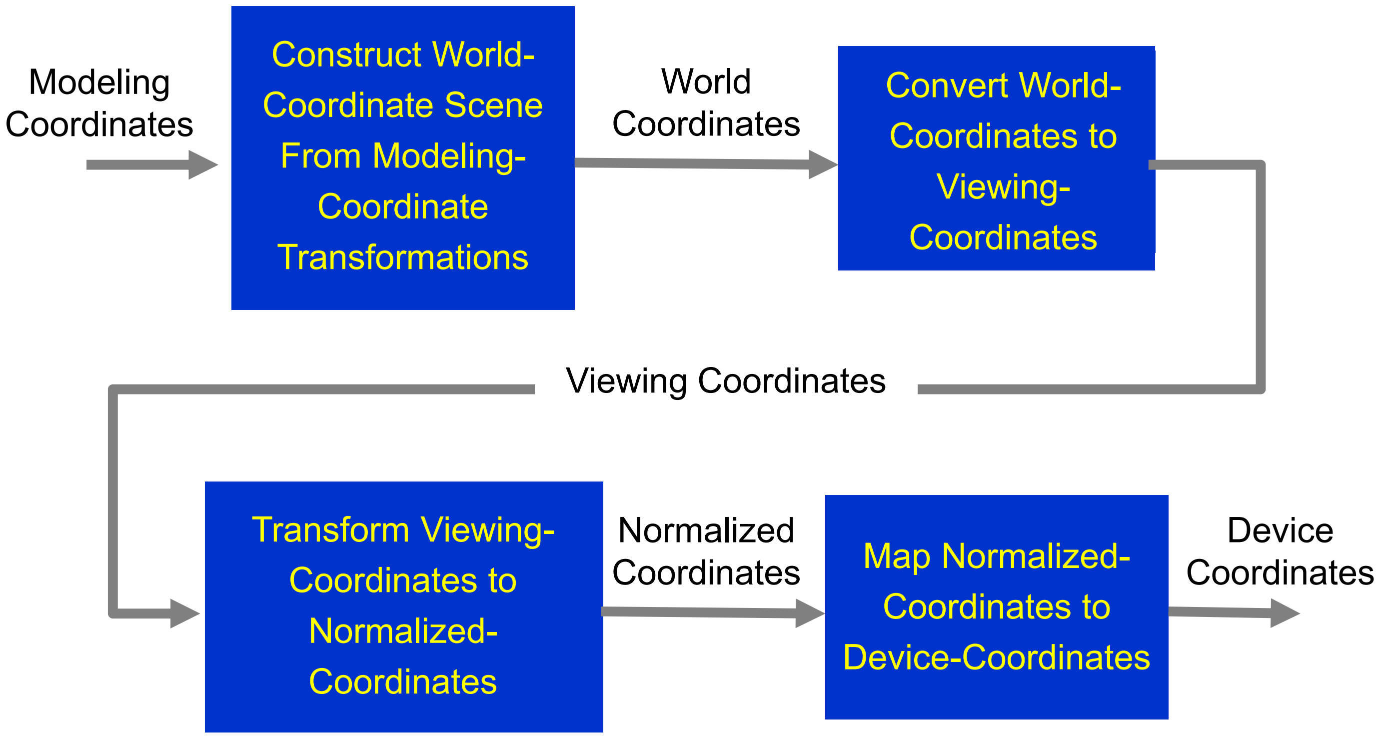
The clipping window is mapped into a viewport.



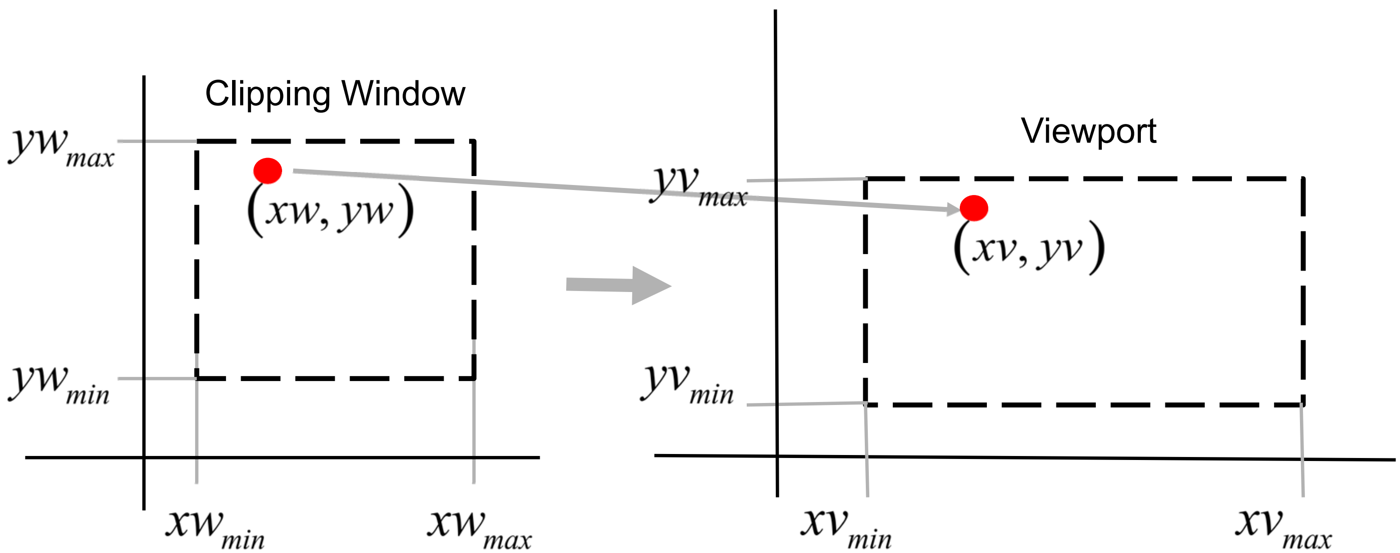
Viewing world has its own coordinates, which may be a non-uniform scaling of world coordinates.

Viewport Coordinates

Two Dimensional Viewing Transformation Pipeline



Window to Viewport Co-ordinate Transformation



Maintain relative size and position between clipping window and viewport.

$$\frac{xv - xv_{\min}}{xv_{\max} - xv_{\min}} = \frac{xw - xw_{\min}}{xw_{\max} - xw_{\min}} \qquad \frac{yv - yv_{\min}}{yv_{\max} - yv_{\min}} = \frac{yw - yw_{\min}}{yw_{\max} - yw_{\min}}$$

Window to Viewport Co-ordinate Transformation

$$\frac{xv - xv_{min}}{xv_{max} - xv_{min}} = \frac{xw - xw_{min}}{xw_{max} - xw_{min}} \quad and \quad \frac{yv - yv_{min}}{yv_{max} - yv_{min}} = \frac{yw - yw_{min}}{yw_{max} - yw_{min}}$$

Solving we get,

$$xv = xv_{min} + (xw - xw_{min})sx$$

$$yv = yv_{min} + (yw - yw_{min})sy$$

where scaling factors

$$sx = \frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}}$$

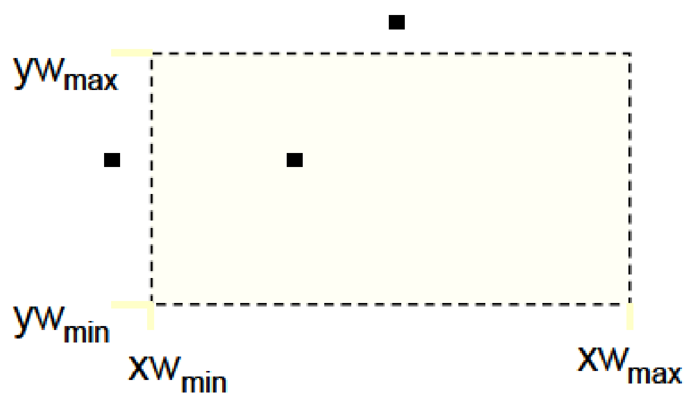
$$sy = \frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}}$$

Clipping

- Any procedure that identifies those portions of a picture that are either inside or outside of a specified region of a space
- A **clip window** can be polygon or curved boundaries
- World- coordinate clipping removes the primitives outside the window from further consideration; thus eliminating the processing necessary to transform these primitives to device space.
- **Clipping type-** point ,line, area, curve and text clipping

Point Clipping

Assume clipping window is Rectangle, point P=(x,y) is saved for display if following inequalities are satisfied



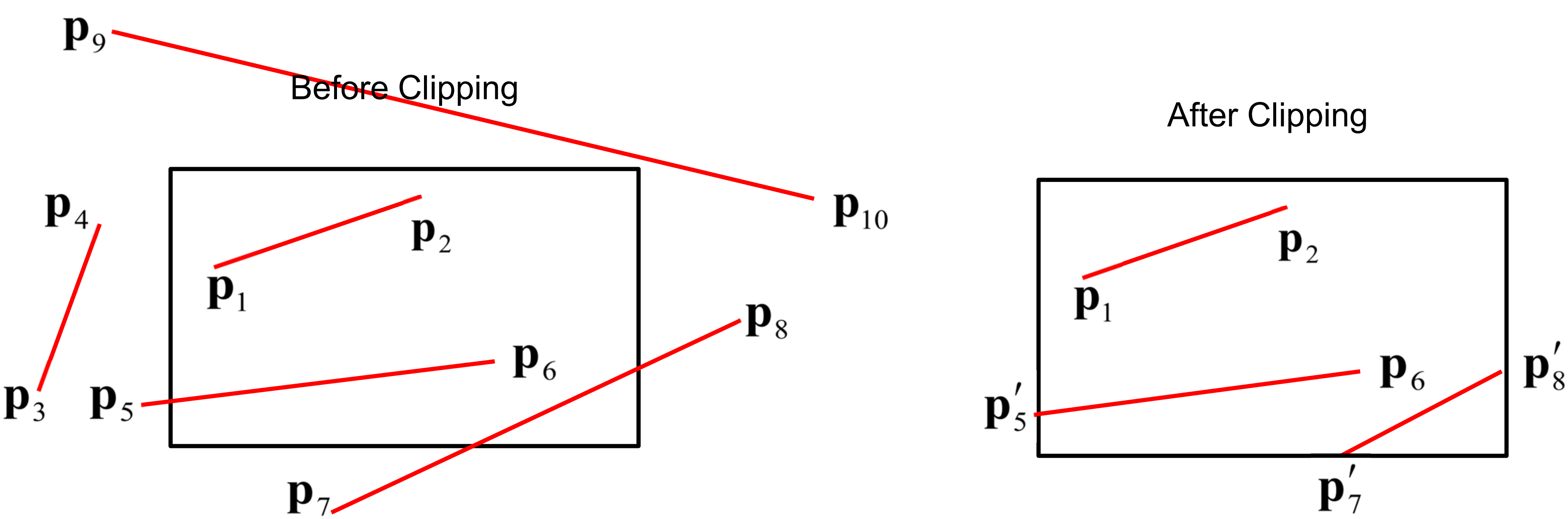
$xw_{min} \leq x \leq xw_{max}$

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

$(xw_{min}, xw_{max}, yw_{min}, yw_{max}) \rightarrow$ can be either world coordinate window boundaries or viewport boundaries

If all the four inequalities are satisfied for a point with co-ordinate (x,y), the point is accepted; i.e not clipped

Line Clipping



Cohen Sutherland Line Clipping Algorithm

- This is an efficient method of accepting or rejecting lines that do not intersect the window edges.
- Divide 2D space into 3x3 = 9- regions.
- Middle region is the **clipping window.**
- Each region is assigned **a** 4-bit code.
- Region codes indicate the position of the regions with respect to the window

4	3	2	1
Top	Below	Right	Left

Region Code Legend

bit 4 : $y > yw_{max}$

bit 3 : $y < yw_{min}$

bit 2 : $x > xw_{max}$

bit 1 : $x < xw_{min}$

TBRL

1001	1000	1010
0001	0000 Window	0010
0101	0100	0110

Cohen Sutherland Line Clipping Algorithm

1. Assign a region code for each endpoints.
2. If both endpoints have a **region code 0000** →**trivially accept** these line.
3. Else, perform the logical AND operation for both region codes.

   3.1 **if** the result is **NOT** 0000 → **trivially reject** the line.

   3.2 **else** (i.e. **result = 0000**, need clipping)

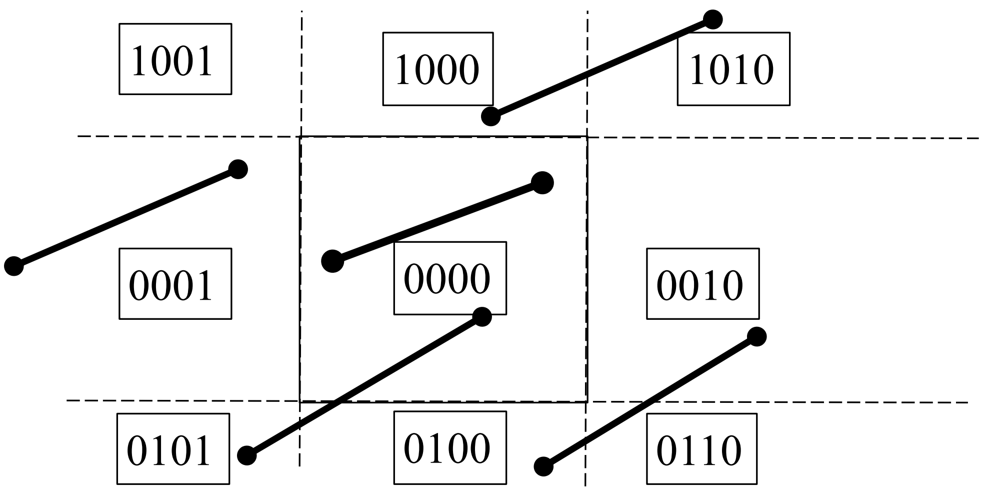
      3.2.1. Choose an endpoint of the line that is outside the window.

      3.2.2. Find the intersection point at the window boundary (based on region code).

      3.2.3. Replace endpoint with the intersection point and update the region code.

      3.2.4. Repeat step 2 until we find a clipped line either trivially accepted or trivially rejected.
4. Repeat step 1 for other lines.

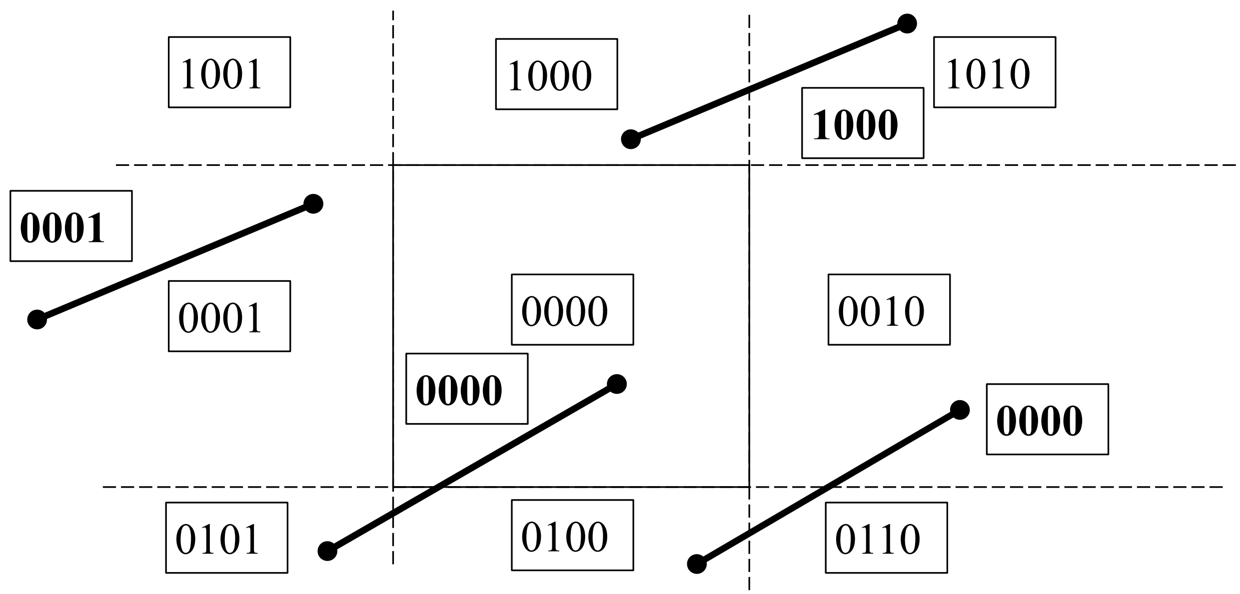
Cohen-Sutherland Algorithm



Both endpoint codes 0000, trivial acceptance, else:

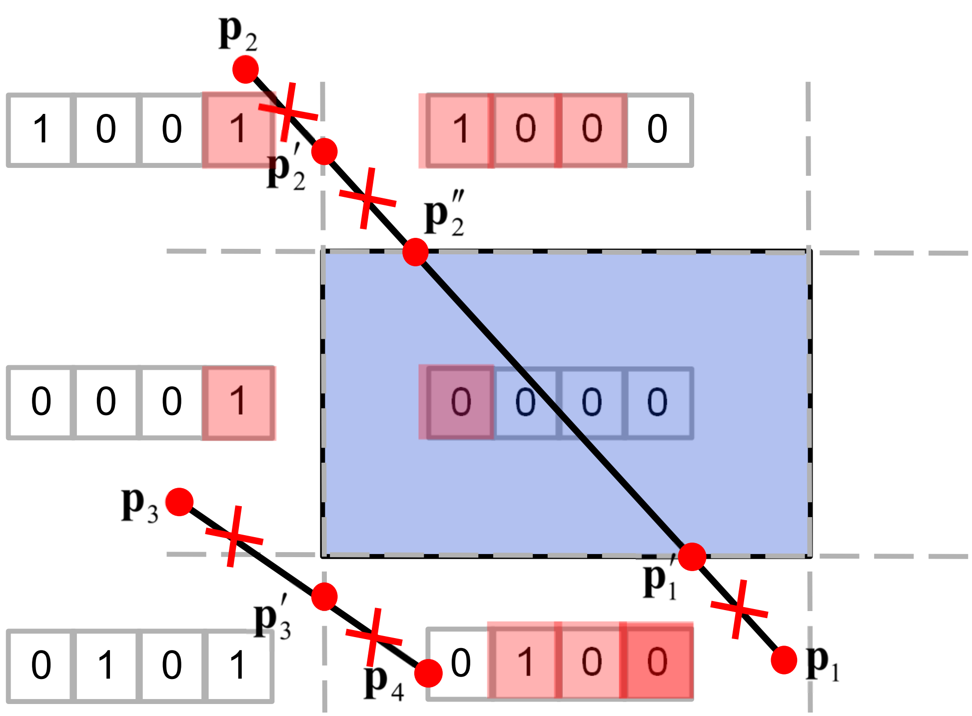
Do logical AND of Outcodes (reject if non-zero)

Cohen-Sutherland Algorithm



Logical AND between codes for 2 endpoints,
Reject line if non-zero – trivial rejection.

Cohen-Sutherland Algorithm



Cohen-Sutherland Algorithm

Intersection calculations:

Intersection with vertical boundary

$y = y_1 + m(x-x_1)$

Where

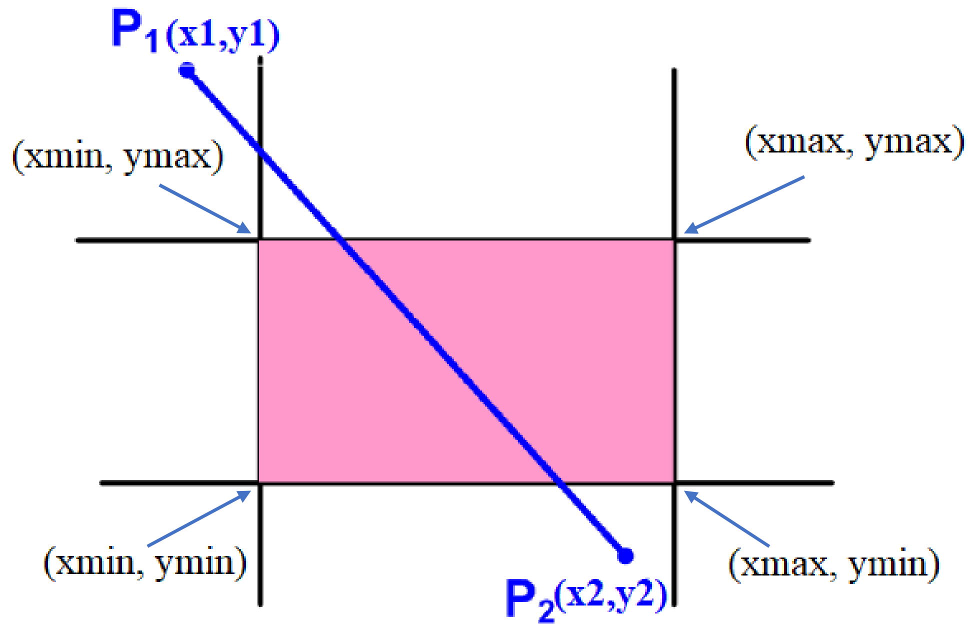
$x = xw_{min}$ or xw_{max}

Intersection with horizontal boundary

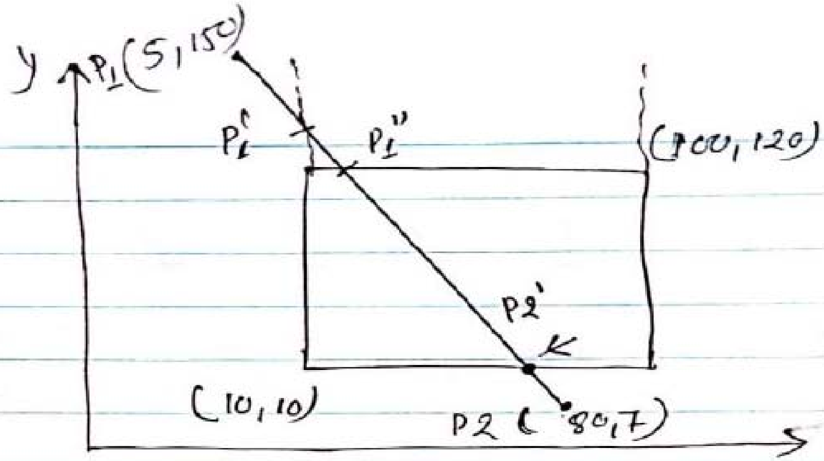
$x = x_1 + (y-y_1)/m$

Where

$y = yw_{min}$ or yw_{max}



⊛ Example-1 :



4	3	2	1
Top	Below	Right	Left

Region Code Legend

bit 4 : $y > yw_{max}$

bit 3 : $y < yw_{min}$

bit 2 : $x > xw_{max}$

bit 1 : $x < xw_{min}$

• Clip P1P2 against given window.

Soln Region Code for P1

$\boxed{1001}$

$x - xw_{min} < 0 \Rightarrow \boxed{bit1 = 1}$

$\overset{120}{\cancel{yw_{max}}} - \overset{150}{\cancel{y}} < 0 \Rightarrow \boxed{bit4 = 1}$

Region code for P2,

$\boxed{0100}$

$x - xw_{min} > 0 \Rightarrow bit1 = 0$

$xw_{max} - x > 0 \Rightarrow bit2 = 0$

$y - ywin < 0 \Rightarrow \boxed{bit3 = 1}$

$yw_{max} - y > 0 \Rightarrow bit4 = 0$

Logical AND of two Region Codes

$$\begin{array}{r} 1001 \\ 0100 \\ \hline 0000 \end{array}$$

Choose P2, to calculate the intersection with window boundary.

$$P2 = (x', yw_{min})$$

$$x' = x_1 + \frac{yw_{min} - \cancel{y}_2}{m}$$

$$m = \frac{7-150}{80-5} \Rightarrow \frac{-143}{75} = -1.91$$

$$x' = 80 + \frac{10-7}{-1.91} \Rightarrow 80 + \frac{3}{1.91}$$

$$\Rightarrow 80 - 1.57$$

$$\Rightarrow 78.43$$

$$x' \simeq 78$$

$$\boxed{P_2' = (78, 10)}$$

→ Region code (0000)

- Logical ORing ~~of~~ of two Region code

$$\begin{array}{r} 1001 \\ 0000 \\ \hline \boxed{1001} \end{array} \rightarrow \text{Need to clip.}$$

choose p1, to calculate the intersection with window Vertical boundary, $P_1' = (\underline{10, y'})$

$$y' = y_1 + m(\cancel{x} - x_1)$$

$$= 150 + -1.91(10-5)$$

$$= 150 - 9.55$$

$$= 140.45$$

$$y' \simeq 141$$

$$\underbrace{P_1' = (10, 141)}$$

↓ Region code

$$\boxed{1000}$$

Logical ORing of two region codes,

$$\begin{array}{r} 0000 \\ 1000 \\ \hline \boxed{1000} \end{array} \rightarrow \text{Need to clip again.}$$

Calculate the intersection with horizontal boundary,

$$P_1'' = (?, y_{wmax})$$

$$x'' = x_1 + \frac{y - y_1}{m}$$

$$= 5 + \frac{120 - 150}{-1.91}$$

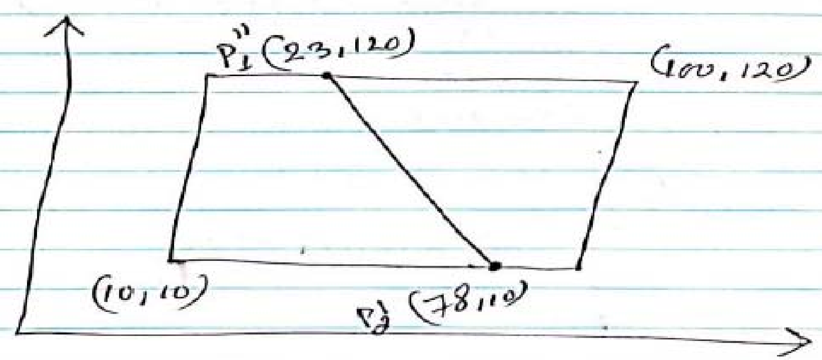
$$= 5 + 15.71$$

$$= 22.71$$

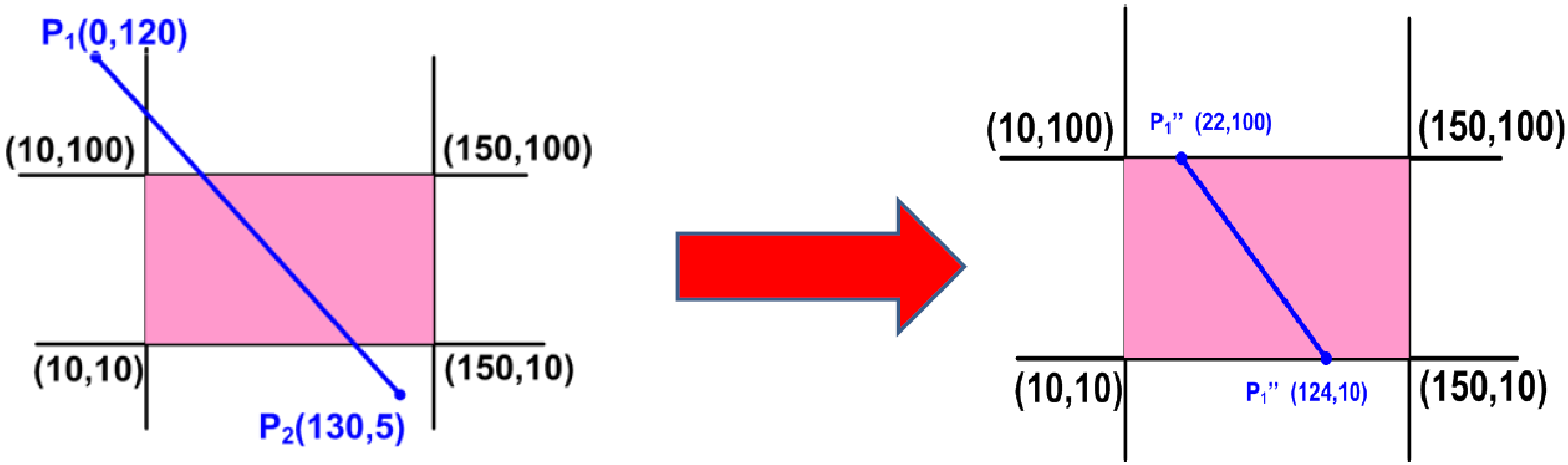
$$x'' \simeq 23$$

$$\boxed{\begin{array}{l} P_1'' = (23, 120) \\ P_2' = 78, 10 \end{array}}$$

Both Region 0000 Accepted & clipped.



Obtain the endpoints of line P_1P_2 after cohen-sutherland clipping



Liang Barsky Line Clipping Algorithm

- Uses parametric equation of a line and inequalities describing the range of the clipping window to determine the intersections between the line and the clip window.
- Efficient algorithm than Cohen Sutherland Algorithm, as not computing the coordinate values at irrelevant vertices
- Parametric equation of a line

$$x = x_1 + t\,(x_2 - x_1) \;\ldots\ldots\ldots\ldots..(1)$$

$$y = y_1 + t\,(y_2 - y_1) \;\ldots\ldots\ldots\ldots..(2)$$

- A point is inside the clipping window if

$$xw_{min} \le x \le xw_{max} \quad \text{i.e. } xw_{min} \le x_1 + t\Delta x \le xw_{max} \text{ where } \Delta x = x_2 - x_1$$

And,

$$yw_{min} \le y \le yw_{max} \quad \text{i.e. } yw_{min} \le y_1 + t\Delta y \le yw_{max} \text{ where } \Delta y = y_2 - y_1$$

Liang Barsky Line Clipping Algorithm

- Each of these inequalities can be written as

$- t\Delta x \leq x1 - xw_{min}$ (multiplying both sides by – in $t\Delta x \geq xw_{min} - x_1$)

$t\Delta x \leq xw_{max} - x_1$

$- t\Delta y \leq y1 - yw_{min}$ (multiplying both sides by – in $t\Delta y \geq yw_{min} - y_1$)

$t\Delta y \leq yw_{max} - y_1$

- These 4 inequalities can be expressed as

$tp_k \leq q_k$ for $k = 1, 2, 3, 4$

where,

$p_1 = -\Delta x$ and $q_1 = x1 - xw_{min}$ (left boundary)

$p_2 = \Delta x$ and $q_2 = xw_{max} - x_1$ (right boundary)

$p_3 = -\Delta y$ and $q_3 = y1 - yw_{min}$ (bottom boundary)

$p_4 = \Delta y$ and $q_4 = yw_{max} - y_1$ (top boundary)

Liang Barsky Line Clipping Algorithm

Algorithm

1. If $p_k = 0$ for some k, then the line is parallel to the clipping boundary, now test q_k

   If $q_k < 0$ for these k, then the line is outside

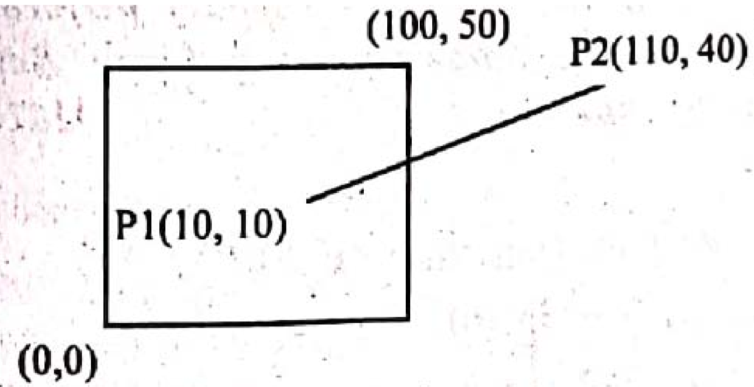
   If $q_k \geq 0$ for these k, then some portion of the line is inside

1. For all $p_k < 0$ (i.e. line proceeds from outside to inside boundary) calculate $t_1 = \max(0, r_k)$ to determine intersection point with clipping boundary and obtain new point for that line at t_1 ($r_k = q_k / p_k$)
2. For all $p_k > 0$ (i.e. line proceeds from inside to outside boundary) calculate $t_2 = \min(1, r_k)$ to determine intersection point with clipping boundary and obtain new point for that line at t_1 ($r_k = q_k / p_k$)
3. If t1 > t2, then discard the line
4. Else new points are calculated as

$$x_{1new} = x_1 + t_1 \Delta x \quad \text{and} \quad y_{1new} = y_1 + t_1 \Delta y$$

$$x_{2new} = x_1 + t_1 \Delta x \quad \text{and} \quad y_{2new} = y_1 + t_1 \Delta y$$

Example of Liang Barsky Line Clipping Algorithm



K	p_k	q_k	r_k
1	$-\Delta x$ $= -(110-10)$ $= -100$ i. e., $p_k<0$	$x_1 - x_{wmin}$ $= 10-0$ $= 10$	r1 $= 10/-(100)$ $= -1/10$ = candidate for u_1
2	Δx $= (110-10)$ $= 100$ i. e., $p_k>0$	$x_{wmax} - x_1$ $= 100-10$ $= 90$	r_2 $= 90/100$ $= 0.9$ = candidate for u_2
3	$-\Delta y$ $= -(40-10)$ $= -30$ i. e.Pk<0	$y_1 - y_{wmin}$ $= 10-0$ $= 10$	r_3 $= 10/-30$ $= -1/3$ = candidate for u_1
4	Δy $= (40-10)$ $= 30$ i. e.Pk>0	$y_{wmax} - y_1$ $= 50-10$ $= 40$	r_4 $= 40/30$ $= 4/3$ = candidate for u_2

We take $u_1 = 0$ and $u_2 = 0.9$

Clipped line

$x_1' = 10+0\times100 = 10$ $\quad x_2' = x1+u_2\times100 = 10+0.9\times100 = 100$

$y_1' = 10+0\times30 = 10$ $\quad y_2' = y1+u_2\times100 = 10+0.9\times30 = 37$

Example of Liang Barsky Line Clipping Algorithm

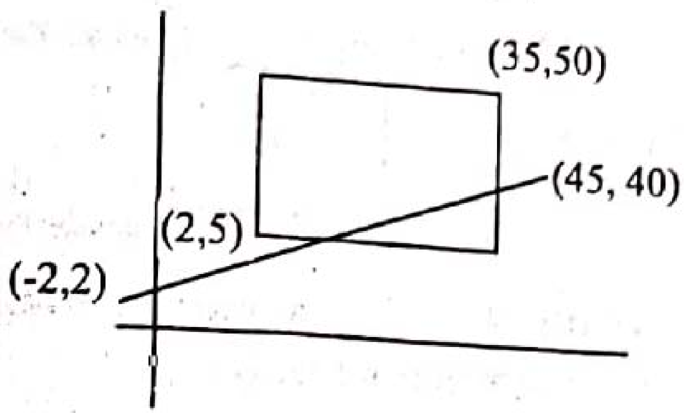
Clipping window: $(x_{min}, y_{min}) = (2,5)$.

And $(x_{max}, y_{max}) = (35,50)$

Line $(x_1 y_1) = (-2,2)$ and $(x_2, y_2) = (45,40)$

$\Delta x = x_2 - x_1 = 45 - (-2) = 47$

$\Delta y = y_2 - y_1 = 40 - 2 = 38$



k	p_k	q_k	$R_k = \frac{q_k}{p_k}$
0	$\Delta x = -47$, $p_k < 0$	$x_1 - x_{min} = -4$	$0.0851(u_1)$
1	$\Delta x = 47$	$x_{max} - x_1 = 37$	$0.787(u_2)$
2	$-\Delta y = -38$	$y_1 - y_{min} = -3$	$0.0789(u_1)$
3	$\Delta y = 38$	$y_{max} - y_1 = 48$	$1.263(u_2)$

$u_1 = \max(0, r_k)$

$= \max(0, 0.0851, 0.0789)$

$= 0.0851$

$u_2 = \min(1, r_k) = \min(1, 0.787, 1.263) = 0.787$

$x_1' = x_1 + u_1 \Delta x = -2 \times 0.051 \times 47 = 1.997 \approx 2$

$y_1' = y_1 + u_1 \Delta y = -2 + 0.0851 \times 38 = 5$

$x_2' = x_1 + u_2 \Delta x = -2 + 0.787 \times 47 = 35$

$y_2' = y_1 + u_2 \Delta y = 2 + 0.787 \times 38 = 32$

Required points are (2,5) and (35, 32)