Chapter 4
Two Dimensional Transformation and Viewing

Introduction

- The process of changing the sizes, orientations and positions of objects by performing some mathematical operation is called **transformation.**
- When a transformation takes place on a 2D plane, it is called 2D transformation.
- Transformation are important because in real world scenario, objects move and are viewed from different angle so graphics system apply transformation to change the way an object appear, its orientation and its size, thus **avoid the problem of redrawing it.**
- Some of the **basic transformation that can be applied to object are translation, rotation, scaling.**
- **Other transformation includes reflection and shear.**

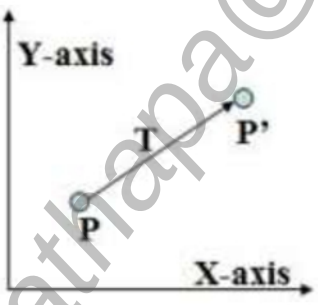
Basic Transformation Techniques

1. 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 p(x,y) by adding a **translation distance** t_x and t_y, to the original coordinate position (x,y) to move the point to new position (x', y') as shown in the figure below

$$x' = x + t_x \qquad y' = y + t_y \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots I$$

where, translation distance pair (t_x, t_y) is called **translation vector or shift vector.**



- We can express the translation equation I, as a single matrix equation by using column vectors to represent coordinates position and the translation vector.

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

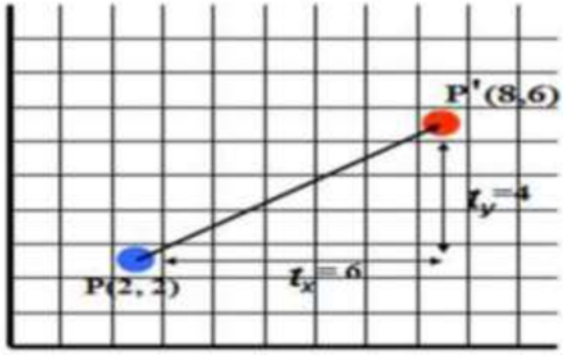
i.e. p' = p + T, where T is transformation Matrix.

- Translation is **Rigid body transformation** that moves objects without deformation i.e. every point on the object is translated by the same amount.

Note:

- *Rigid body transformations are the ones which preserve the shape and size of the object i.e. magnitude and the angle also. Pure rotations and pure reflections are also rigid body transformation. Uniform scaling is not a rigid body transformation as it changes the magnitude.*

For Example:
Given a point P with original position (2,2). Then after performing translation operation with t_x = 6 and t_y = 4, we get new transformed coordinate P' with coordinate (8,6).

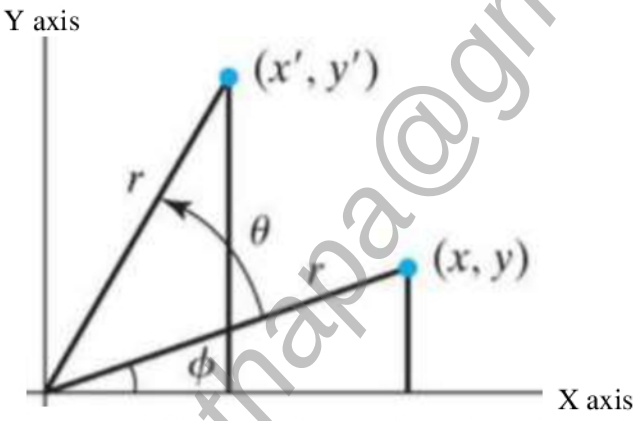


2. Rotation

- A two dimensional rotation is applied to an object by repositioning it along a circular path in the XY plane.
- To apply rotation, we need to specify two things
  - i. Rotation angle(θ): **positive θ means anticlockwise rotation, negative θ means clockwise rotation**
  - ii. Pivot point i.e. Rotation Point about which object is to be rotated.

Case 1: Counter Clockwise I.e. Anticlockwise Rotation of a point about origin pivot point

- In the figure below, a point at position (x, y) is rotated to position (x', y') through an angle θ about rotation point origin i.e. (0,0). The original angular displacement of the point form the x-axis is ϕ and r is the constant distance of the point from the origin.



- So using, standard trigonometric identities, the transformed coordinates in term of angle θ and ϕ can be expressed as below

For Point P(x, Y)

$\cos\phi = x/r$ i.e. $x = r\cos\phi$

$\sin\phi = y/r$ i.e. $y = r\sin\phi$

For the transformed point, P'(x', y'), new angle = (ϕ + θ)

Hence,

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

or, $x' = r * \cos(\phi+\theta)$

or, $x' = r * \cos\phi\cos\theta - r * \sin\phi\sin\theta$

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

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

or, $y' = r * \sin(\phi+\theta)$

or, $y' = r * \sin(\phi)\cos\theta + r * \cos\phi\sin\theta$

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

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

- With the column vector representation, we can write the rotation equation for counter clockwise rotation in matrix form as

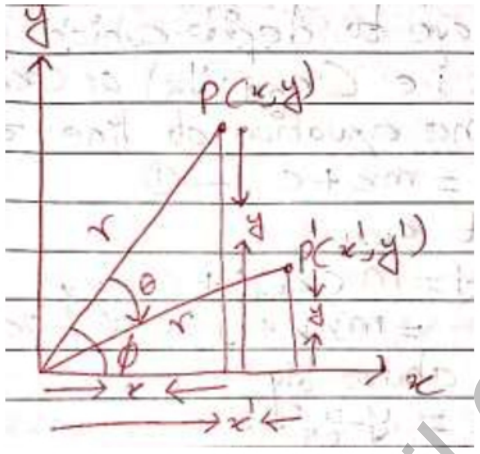
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

i.e. P' = R. P where, R is the rotation matrix.

Note: if θ = -θ, then the above rotation matrix performs clockwise rotation.

Case 2: Clockwise Rotation of a point about origin pivot position

- Suppose, we need to perform clockwise rotation of point P(x,y) to P'(x', y') about angle θ as shown in figure below.



- So using, standard trigonometric identities, the transformed coordinates in term of angle θ and ϕ can be expressed as below

For Point P(x, Y)

$\cos\phi = x/r$ i.e. $x = r\cos\phi$

$\sin\phi = y/r$ i.e. $y = r\sin\phi$

For the transformed point, P'(x', y'), new angle = (ϕ - θ)

Hence,

$\cos(\phi - \theta) = x'/r$

or, $x' = r * \cos(\phi - \theta)$

or, $x' = r\cos\phi\cos\theta + r\sin\phi\sin\theta$

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

$\sin(\phi - \theta) = y'/r$

or, $y' = r * \sin(\phi - \theta)$

or, $y' = r * \sin(\phi)\cos\theta - r\cos\phi\sin\theta$

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

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

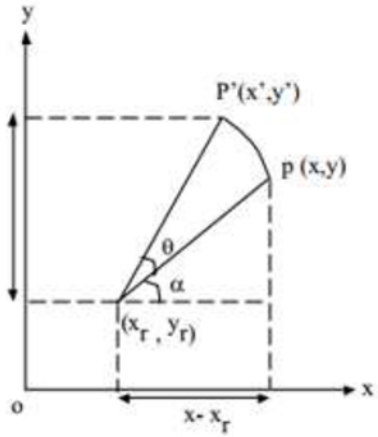
- With the column vector representation, we can write the rotation equation for clockwise rotation in matrix form as

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

i.e. P' =R. P where, R is the rotation matrix.

Case 3: Counter Clockwise rotation of a point about an arbitrary pivot position

- Rotation of a point about an arbitrary pivot position is shown in the figure below.



- For performing anticlockwise rotation of any point P (x, y) about any arbitrary point (X_r, Y_r), we have to perform following **successive transformations**
  i. First we shift /translate the given point such that the arbitrary point is translated to the origin using translation vector ($-X_r$, $-Y_r$)
  i.e.
  $$X^l = X - X_r \text{ and } Y^l = Y - Y_r$$
  ii. The, perform rotation operation about origin
  i.e.
  $$X^l = X\cos\theta - Y\sin\theta \qquad \text{i.e. } X^l = (X - X_r)\cos\theta - (Y - Y_r)\sin\theta$$
  $$Y^l = X\sin\theta + Y\cos\theta \qquad \text{i.e. } Y^l = (X - X_r)\sin\theta + (Y - Y_r)\cos\theta$$
  iii. Finally shift the transformed point such that the arbitrary point is translated back to its original position using translation vector (X_r, Y_r)
  i.e.
  $$X^l = X + X_r \qquad \text{i.e. } X^l = (X - X_r)\cos\theta - (Y - Y_r)\sin\theta + X_r$$
  $$Y^l = Y + Yr \qquad \text{i.e. } Y^l = (X - X_r)\sin\theta + (Y - Y_r)\cos\theta + Y_r$$

- Hence, the transformation equation for rotation(anticlockwise) of a point about any specified rotation point (x_r, y_r) can be generalized as
$$x' = x_r + (x - x_r)\cos\theta - (y - y_r)\sin\theta$$
$$y' = y_r + (x - x_r)\sin\theta + (y - y_r)\cos\theta$$
- Similar to translation, rotations are also **rigid body transformations** that move objects without deformation i.e. every point on an object is rotated through same angle.

Case 4: Clockwise rotation of a point about an arbitrary pivot position
$$X^l = X_r + (X - X_r)\cos\theta + (Y - Y_r)\sin\theta$$
$$Y^l = Y_r - (X - X_r)\sin\theta + (Y - Y_r)\cos\theta$$

Assignment: Derive the equation for clockwise rotation of a point about and arbitrary pivot point.

Note:

sin(a+b) =sina cosb + cosa sinb
sin(a-b) =sina cosb - cosa sinb
cos (a+b) = cosa cosb - sina sinb
cos (a-b) = cosa cosb + sina sinb

3. Scaling

- This transformation changes the size of an object such that we can magnify or reduce its size.
- 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.

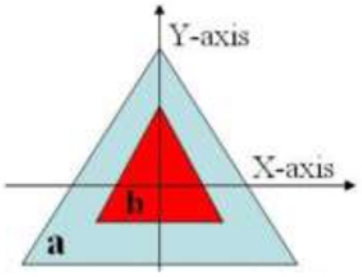


Fig: Scaling Large Triangle 'a' to Small Triangle 'b'

- If s_x and s_y are equal **then scaling is uniform** such that it maintains relative object proportion otherwise scaling is **called differential scaling.**

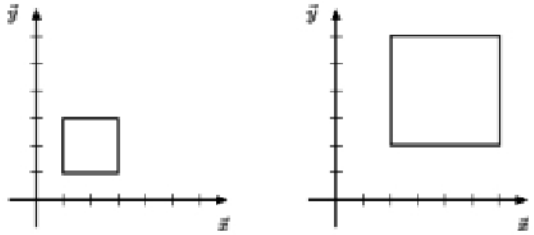


Fig: Uniform Scaling

- Any positive number can be assigned to the scaling factors s_x and s_y, values less than 1 reduces the size of objects while values greater than 1 produces enlargement. Value 1 for s_x and s_y leaves the size of object unchanged.

Case 1: Scaling About Origin

- If p(x,y) be scaled to a point p'(x', y') by s_x time in X-units and s_y times in y-units then the equation for scaling is given as

$$x' = s_x . x$$
$$y' = s_y . y$$

- With the column vector representation, we can write the scaling equation in matrix form as

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

i.e. P' = S. P, where S is the Scaling Matrix

Case 2: Scaling About arbitrary point

- For performing scaling of any point P(x, Y) by Scaling Factor (Sx, S_y) about any arbitrary point (X_f, Y_f), we have to perform following successive transformations
  - i. First we shift /translate the given point such that the arbitrary point is translated to the origin using translation vector ($-X_f$, $-Y_f$)
    i.e.
    $$X^l = X - X_f \text{ and } Y^l = Y - Y_f$$
  - ii. Perform scaling operation about origin.
    i.e.
    $X^l = S_x . X$ i.e. $X^l = S_x . (X - X_f)$
    $Y^l = S_y . Y$ i.e. $Y^l = S_y . (Y - Y_f)$
  - iii. Finally shift the transformed point such that the arbitrary point is translated back to its original position using translation vector (X_f, Y_f)

    i.e.
    $X^l = X + X_f$ i.e. $X^l = S_x . (X - X_f) + X_f$
    $Y^l = Y + Y_f$ i.e. $Y^l = S_y . (Y - Y_f) + Y_f$

- Hence, if p(x,y) be scaled to a point p'(x', y') by s_x time in X-units and s_y times in y-units about arbitrary point (x_f, y_f) then the equation for scaling is given as

$$x' = x_f + s_x . (x - x_f)$$
$$y' = y_f + s_y . (y - y_f)$$

Other Transformation

1. Reflection

- A reflection is a transformation that produces **a mirror image** of an object.
- The mirror image for a two dimensional reflection is generated relative to an axis of reflection by rotating the object 180^0 about the reflection axis.
- In 2D-transformation, reflection is generated relative to an axis of reflection.
- Following are examples of some common reflections:
  i. Reflection about the line y=0, the X- axis.

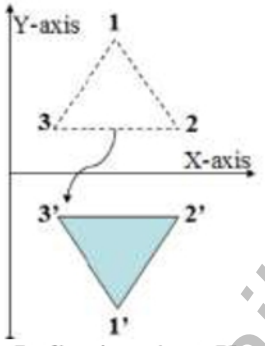


Fig: Reflection about X-axis

- The line representing x-axis is y = 0.
- The reflection of a point P(x, y) on x-axis, changes the y-coordinate sign i.e. Reflection about x-axis, the reflected position of P(x, y) will be P'(x, -y).
- Hence, reflection in x-axis is accomplished with transformation matrix given below

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

ii. Reflection about the line x=0, the Y- axis.

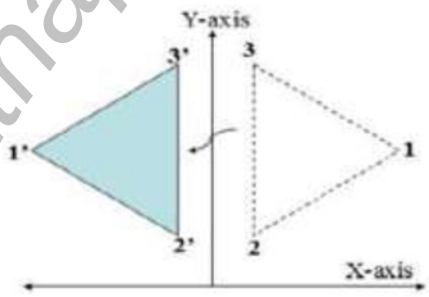


Fig: Reflection about Y-Axis

- The line representing y-axis is x = 0.
- The reflection of a point P(x, y) on y-axis changes the sign of x-coordinate i.e. P(x, y) changes to P'(-x, y).
- Hence reflection of a point on y-axis is accomplished with transformation matrix given below

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

iii. Reflection about origin.

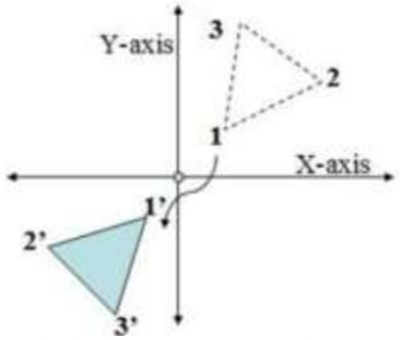


Fig: Reflection about origin

- The reflection on the line perpendicular to xy-plane and passing through origin flips x and y co-ordinates both.
- So sign of x and y co-ordinate value changes.
- The equivalent matrix equation for the point is

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

iv. Reflection of an object w.r.t the straight line y=x.

- If we choose the reflection axis as the diagonal line y= x, then the reflection of a point P(x, y) on y=x axis become P'(y, x).
- The equivalent matrix equation for the point is

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

- We can also derive this matrix by concatenating a sequence of rotation and coordinate axis reflection matrices.
- So when more than one transformations are applied for performing a task then such transformation is **called composite transformation**.

Two Ways

1. Following steps are carried out to reflect the object about y=x axis as shown in figure below.
   i. Rotate about origin in clockwise direction by 45^0. This rotate the line y=x to X-axis.
   ii. Now perform reflection about X-axis.
   iii. Finally Rotate in counterclockwise direction by 45^0.

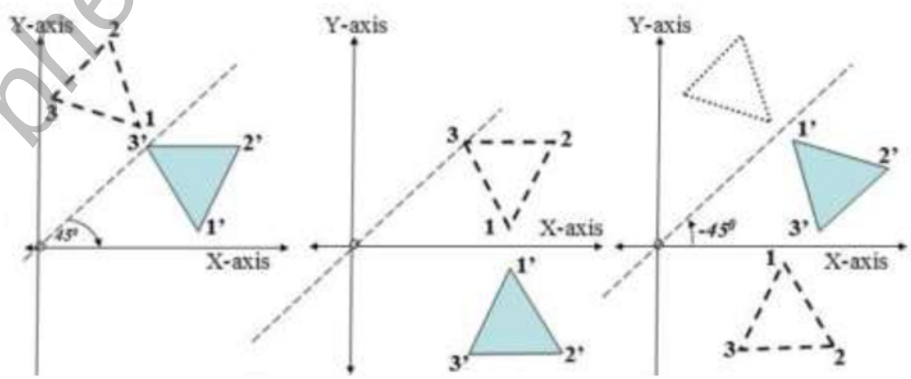


Fig: Steps for Reflecting an object about Y=x axis

So the composite transformation required for reflecting the given object about y=x axis is

$$T = R_{\theta=45^0}\ R_x\ R_{\theta=-45^0}$$

2. Following steps are carried out to reflect the object about y=x axis as shown in figure below.
   i. Rotate about origin in counter clockwise direction by 45^0. This rotate the line y=x to Y-axis.
   ii. Now perform reflection about Y-axis.
   iii. Finally Rotate in clockwise direction by 45^0.

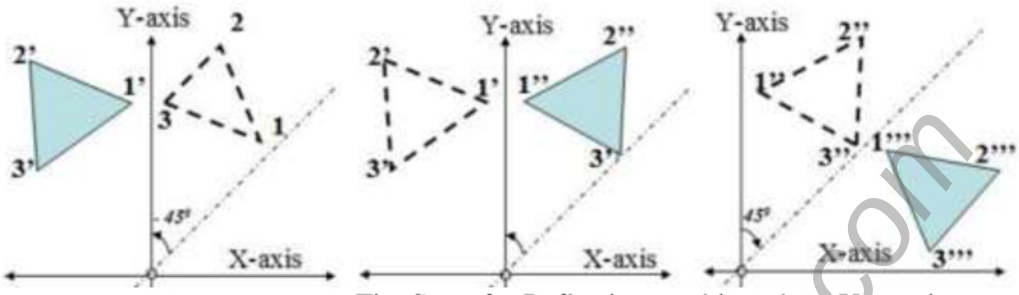


Fig: Steps for Reflecting an object about Y=x axis

So the composite transformation required for reflecting the given object about y=x axis is

$$T = R_{\theta=-45^0}\ R_y\ R_{\theta=45^0}$$

v. Reflection of an object w.r.t the straight line y= -x.

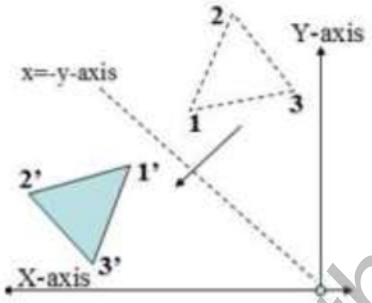


Fig: Reflection of an Object about Y=-X

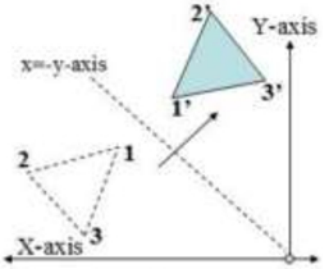


Fig: Reflection of an Object about Y=-X

- If we choose the reflection axis as the diagonal line y= -x, then the reflection of a point P(x, y) on y=x axis become P'(-y, -x).
- The equivalent matrix equation for the point is

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

- We can also derive this matrix by concatenating a sequence of rotation and coordinate axis reflection matrices.

Two Ways

1. First Way

i. Rotate about origin in counter clockwise direction by 45^0. This rotate the line y=-x to X-axis.
ii. Now perform reflection about X-axis.
iii. Finally Rotate in clockwise direction by 45^0.

So the composite transformation required for reflecting the given object about y=x axis is

$$T = R_{\theta=-45^0}\ R_X\ R_{\theta=45^0}$$

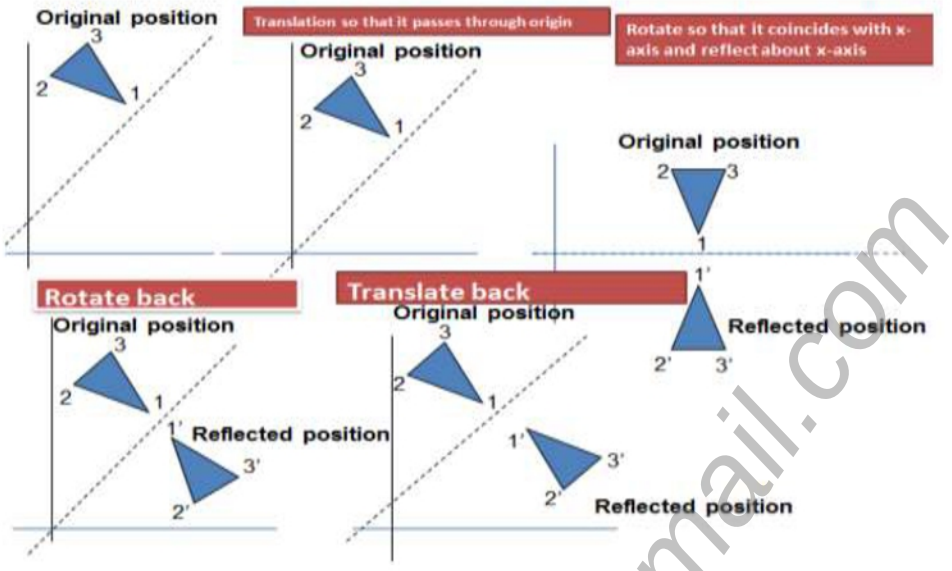
2. Second Way

i. Rotate about origin in clockwise direction by 45^0. This rotate the line y=-x to Y-axis.
ii. Now perform reflection about Y-axis.
iii. Finally Rotate in anticlockwise direction by 45^0.

So the composite transformation required for reflecting the given object about y=x axis is

$$T = R_{\theta=45^0}\ R_y\ R_{\theta=-45^0}$$

ii. Reflection of an object w.r.t. arbitrary axis y=mx + c
In order to reflect an object about any arbitrary axis y = mx + b , it requires series of transformation steps, as below.



Steps

- Translate the object so that reflection axis passes through origin i.e. Apply T(0,-b)
- Rotate (clockwise(-θ) or anticlockwise(θ)) object such that the reflection axis coincides with one of the coordinate axis, in this case, X axis i.e. apply clockwise rotation: $R_{(-\theta)}$ where

$$\theta = \tan^{-1}(m).$$

- Reflect the object about X-Axis i.e. apply R_x
- Rotate back so that reflection axis regains its initial slope i.e. apply $R_{(\theta)}$
- Translate back to return the object to its original position i.e. apply T(0,b)

So the composite transformation required for reflecting the given object about y=x axis is

$$T = T_{(0,b)}\ R_{(\theta)}\ R_X\ R_{(-\theta)}\ T_{(0,-b)}$$

2. Shearing
   - A shear is a transformation that distorts the shape of an object along coordinate axes.
   - It distorts the shape of an object such that the transformed shape appears as if the object were composed of internal layers that had been caused to slide over each other.
   - Shearing is a non-rigid body transformation because it moves object with deformation.

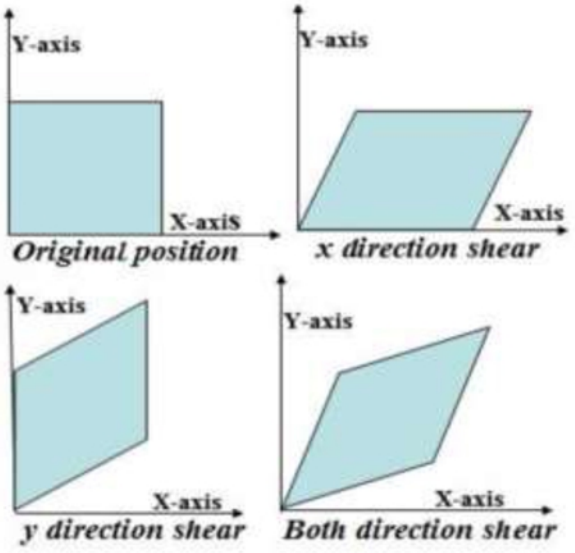


Fig: Different **Shearing Operations that transforms a square to Rhombus (all side equal)**

- Two common shearing transformations are those that shift coordinate x-values and those that shift y-values.
- In X direction shear, it shifts X-coordinate values by an amount proportion to y value but y remains same. The equation for sharing transformation is given as

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

Where, S_{hx} is shearing constant, can be any real number.

The equivalent matrix representation for above equation is given as

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

- In Y direction shear, it shifts Y-coordinate by an amount proportion to x value values but x remains same. The equation for sharing transformation is given as

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

Where, S_{hy} is shearing constant, can be any real number.

The equivalent matrix representation for above equation is given as

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

- In Both direction shear, it shifts both X and Y-coordinate values. The equation for sharing transformation is given as

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

Where, S_{hx} and S_{hy} are shearing constant, can be any real number.

The equivalent matrix representation for above equation is given as

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

Numerical Examples

1. Translate the given points (2,5) by the translating value (3,3).

   Solution:
   Given Point P (2, 5) and translation distance Tx= 3, and Ty= 3
   We have From Translation Matrix

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

   i.e.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 5 \\ 8 \end{bmatrix}$$

   Therefore, P(2,5) is translated to new point P'(5,8).

2. Translate the given square having coordinate A (0,0), B (3,0), and C (3,3) and D (0, 3) by the translating. value 2 in both directions.

Soln:

Given point

A(0,0) C(3,3)

B(3,0) D(0,3)

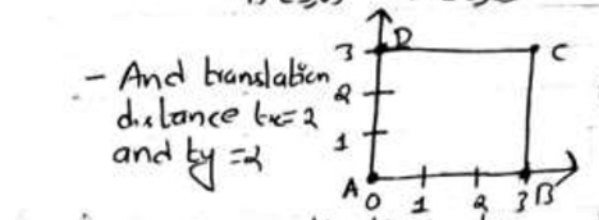


fig: Original image

- And translation distance $t_x = 2$ and $t_y = 2$

Now from reflection matrix we have

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

$$\therefore A' = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

$$B' = \begin{bmatrix} 3 \\ 0 \end{bmatrix} + \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$$

$$C' = \begin{bmatrix} 3 \\ 3 \end{bmatrix} + \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

$$D' = \begin{bmatrix} 0 \\ 3 \end{bmatrix} + \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \end{bmatrix}$$

$A(0,0) \rightarrow A'(2,2)$

$B(3,0) \rightarrow B'(5,2)$

$C(3,3) \rightarrow C'(5,5)$

$D(0,3) \rightarrow D'(2,5)$

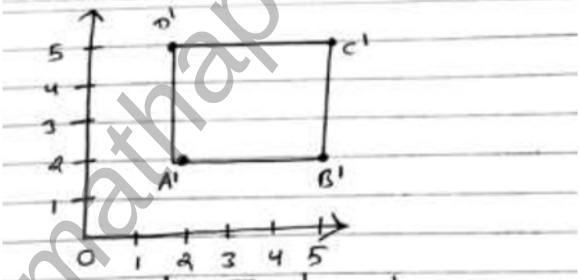


fig: Transformed image

3. Rotate a triangle at A (0, 0), B (6, 0) and C (3, 3) by 90^0 about origin in anticlockwise direction.

Solution:

Given point

A(0,0) B(6,0) C(3,3)

$\theta = 90^\circ$

Direction = Anticlockwise,

Now from the Rotation matrix we have

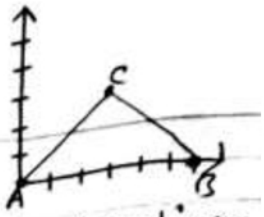


fig: Original image

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

$\therefore A' = \begin{bmatrix} \cos 90 & -\sin 90 \\ \sin 90 & \cos 90 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

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

$\therefore B' = \begin{bmatrix} \cos 90 & -\sin 90 \\ \sin 90 & \cos 90 \end{bmatrix} \begin{bmatrix} 6 \\ 0 \end{bmatrix}$

$= \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 6 \\ 0 \end{bmatrix}$

$= \begin{bmatrix} 0 \\ 6 \end{bmatrix}$

And $C' = \begin{bmatrix} \cos 90 & -\sin 90 \\ \sin 90 & \cos 90 \end{bmatrix} \begin{bmatrix} 3 \\ 3 \end{bmatrix}$

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

$= \begin{bmatrix} -3 \\ 3 \end{bmatrix}$

$\therefore A(0,0) \rightarrow A'(0,0),\ B(6,0) \rightarrow B'(0,6),\ C(3,3) \rightarrow C'(-3,3)$

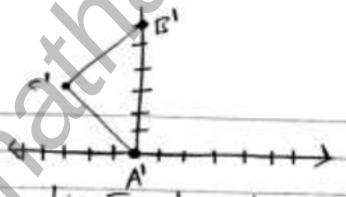


fig: Transformed Image

4. Scale an object (4,4) (3,2) (5,2) about a fixed point (4,3) by Scaling Factor 2 on both directions.

Soln: Given points
A(4,4), B(3,2) and C(5,2)

Scaling factors,
$s_x = s_y = 2$

Pivot point $(x_f, y_f) = (4,3)$

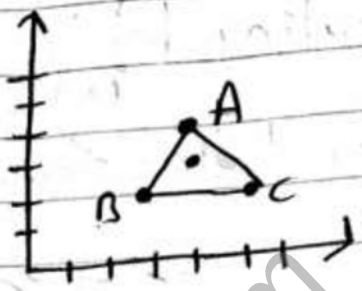


fig: Original image

Now first translate the object such that the pivot point is shifted to origin.

for this translation distance $(t_x\ t_y) = (-4,-3)$

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

$\therefore$ A is translated to $(4-4, 4-3) = (0,1)$
B is translated to $(3-4, 2-3) = (-1,-1)$
C is translated to $(5-4, 2-3) = (1,-1)$

Then Apply scaling about origin.

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

$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$

∴ A is scaled to $(2*0+0*1, 0*0+2*1)$
$= (0,2)$

B is scaled to $(2*-1, 2*-1) = (-2,-2)$

C is scaled to $(2*1, 2*-1) = (2,-2)$

Finally apply inverse translation to shift to its original position.

For this translation distance $= (4,3)$

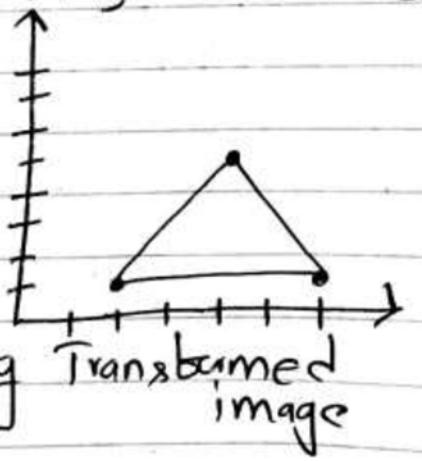
Since $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$

Hence A is translated to $(0+4, 2+3)$
$= (4,5)$

B is translated to $(-2+4, -2+3)$
$= (2,1)$

C is translated to $(2+4, -2+3) = (6,1)$

∴ $A(4,4) \rightarrow A'(4,5)$
$B(3,2) \rightarrow B'(2,1)$
$C(5,2) \rightarrow C'(6,1)$



Note: Can be directly done by using following Transformation equation
$x' = x_b + s_x(x - x_b)$
$y' = y_b + s_y(y - y_b)$

Matrix Representation and Homogeneous Coordinate

- The matrix representations for translation, scaling, and rotation are, respectively

$$P' = T + P$$
$$P' = S.\ P$$
$$P' = R.\ p$$

- Here translation is treated differently (as addition) from scaling and rotation (as multiplications), so that to treat all three transformations in a consistent way, **a homogeneous coordinate system is used**.
- The homogeneous co-ordinate system provides a uniform framework for handling different geometric transformations, simply as multiplication of matrices.
- For a task like animation that requires scaling, rotation, translation, instead of applying each transformation one at a time separately we can combine them so that final coordinates are obtained directly from initial coordinates thus eliminating intermediate steps.
- In homogeneous coordinate representation each 2D point (X, Y) is represented as homogeneous coordinate triple (X_h, Y_h, h), where $X = X_h/h$, $Y = Y_h/h$ (h is 1 usually for 2D case).
- Example: 2D point (2, -4) can be represented as (2, -4, 1) or (4, -8, 2) or (6, -12, 3) etc.
- If points are expressed in homogenous coordinates, all geometrical transformation equations can be **represented as matrix multiplications.**
- Coordinates of a point are represented as three element column vectors; transformation operations are written as 3 x 3 matrices.

Why Homogeneous Coordinate System?

- To represent all the transformation equations as matrix multiplications.
- To perform more than one transformation at a time.
- To reduce unwanted calculations of intermediate steps, to save time and memory and produce a sequence of transformations

1. Translation using Homogeneous Coordinate

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

$$\text{Therefore, } P' = T(t_x, t_y).P$$

And inverse translation can be done as

$$T^{-1}(t_x, t_y) = T(-t_x, -t_y)$$

Note: This clearly shows how the addition operation required for translation is converted to equivalent multiplication operation

2. Rotation about origin in anticlockwise direction Using Homogeneous Coordinate.

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

$$\text{Therefore, } P' = R(\theta).\ P$$

And inverse rotation can be done as

$$R^{-1}(\theta) = R(-\theta)$$

3. **Rotation about origin in clockwise direction Using Homogeneous Coordinate.**

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

Therefore, $P' = R(-\theta).\ P$

And inverse rotation can be done as

$$R^{-1}(-\theta) = R(\theta)$$

4. **Scaling using Homogeneous Coordinate.** *

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

And inverse scaling matric can be obtained by replacing S_{hx} with $1/S_{hx}$ and S_{hy} with $1/S_{hy}$

5. **Reflection using Homogeneous Coordinate**
   i. Reflection about X –axis

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

   ii. Reflection about Y-axis

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

   iii. Reflection about y=x

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

How?

As discussed earlier, the composite matrix for reflection of an object about y= x axis is given as

$$T = R_{\theta=45^0}\ R_x\ R_{\theta=-45^0}$$

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

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

iv. Reflection about y=-x

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

How?

As discussed earlier, the composite matrix for reflection of an object about y= x axis is given as

$$T = R_{\theta=-45^0} \; R_X \; R_{\theta=45^0}$$

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

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

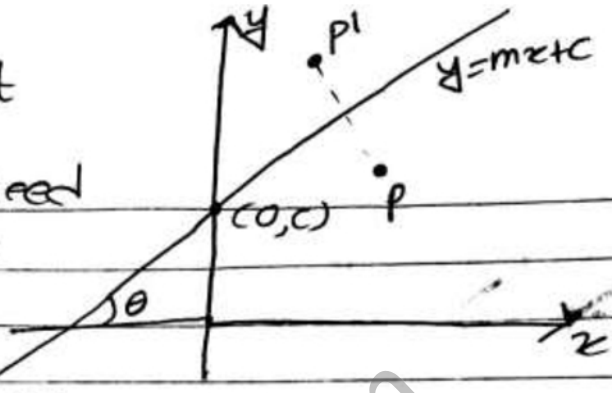
$$= \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & -1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

v. Reflection about any line y=mx +c

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{pmatrix} \frac{-(m^2-1)}{(m^2+1)} & \frac{2m}{(m^2+1)} & \frac{2mc}{(m^2+1)} \\ \frac{2m}{(m^2+1)} & \frac{(m^2-1)}{(m^2+1)} & \frac{2c}{(m^2+1)} \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

How?

In order to reflect an object about any line $y = mx + c$, we need to perform composite transformation as below.



$$T = T_{(0,c)} \cdot R_{\theta} \cdot R_{x} \cdot R_{-\theta} \cdot T_{(0,-c)}$$

Here $T_{(0,-c)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -c \\ 0 & 0 & 1 \end{bmatrix}$

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

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

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

$T_{(0,c)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & c \\ 0 & 0 & 1 \end{bmatrix}$

And slope $m = \tan\theta$

We know $\cos^2\theta = \frac{1}{\tan^2\theta + 1} = \frac{1}{m^2 + 1}$

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

Again we have

$\sin^2\theta + \cos^2\theta = 1$

$\Rightarrow \sin^2\theta = 1 - \cos^2\theta$

$\Rightarrow \sin^2\theta = 1 - \frac{1}{m^2+1} = \frac{m^2+1-1}{m^2+1}$

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

So combining all basic transformation, we get

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

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

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

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

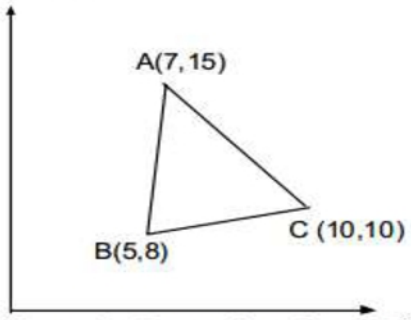
$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1-m^2}{m^2+1} & \frac{2m}{m^2+1} & -\frac{2cm}{m^2+1} \\ \frac{2m}{m^2+1} & \frac{m^2-1}{m^2+1} & \frac{c-cm^2}{m^2+1} \\ 0 & 0 & 1 \end{bmatrix}$$

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

Numerical

1. Rotate triangle ABC by 45^0 clockwise about origin and scale it by (2,3) about origin.
   Solution:

Solution:



Step 1: Rotation by 45˚ (clock wise)

Step-II: Scaling by (2,3)

S(2,3)

Net transformation: $= S(2,3).R(-45)$

$$= \begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45 & Sin45 & 0 \\ -Sin45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

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

Finally, the transformed co-ordinates are

$A' = TA$

$$\begin{bmatrix} 2/\sqrt{2} & 2/\sqrt{2} & 0 \\ -3/\sqrt{2} & 3/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 7 \\ 15 \\ 1 \end{bmatrix} =$$

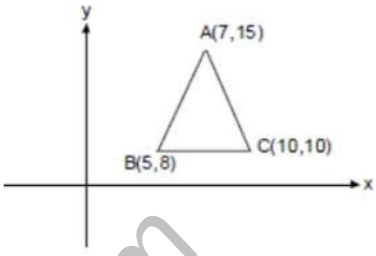
$B' = TB$

$$\begin{bmatrix} 2/\sqrt{2} & 2/\sqrt{2} & 0 \\ -3/\sqrt{2} & 3/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 5 \\ 8 \\ 1 \end{bmatrix} =$$

$C' = TC$

$$\begin{bmatrix} 2/\sqrt{2} & 2/\sqrt{2} & 0 \\ -3/\sqrt{2} & 3/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 10 \\ 10 \\ 1 \end{bmatrix} =$$

2. Rotate the △ABC by 90° anti clockwise about (5,8) and scale it by (2,2) about (10,10)
Solution:



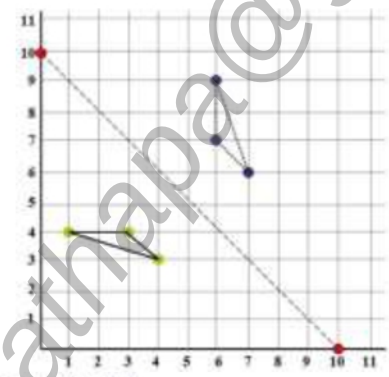
The composite matrix for the above transformation can be defined as

$$T = T_{(10,10)}\, S_{(2,2)}\, T_{(-10,-10)}\, T_{(5,8)}\, R_{(\theta=90^0)}\, T_{(-5,-8)}$$

$$= \begin{bmatrix}1 & 0 & 10\\ 0 & 1 & 10\\ 0 & 01 & \end{bmatrix}\begin{bmatrix}2 & 0 & 0\\ 0 & 2 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & -10\\ 0 & 1 & -10\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 5\\ 0 & 1 & 8\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\cos 90 & -\sin 90 & 0\\ \sin 90 & \cos 90 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & -5\\ 0 & 1 & -8\\ 0 & 0 & 1\end{bmatrix}$$

Complete yourself……

3. A mirror is placed such that it passes through (0,10) (10,0). Find the mirror image of an object (6,7), (7,6), (6,9).
Solution:



Now, the slope of the line (m) = $(y_2-y_1) / (x_2-x_1)$
= (0 - 10) / (10 - 0) = -1
Thus, the rotation angle (θ) = $\text{Tan}^{-1}(m) = \text{Tan}^{-1}(-1) = -45^0$

The composite matrix for the above transformation can be defined as

$$T = T_{(10,0)}\, R_{(\theta=-45)}\, R_X\, R_{(\theta=45)}\, T_{(-10,\,0)}$$

$$= \begin{pmatrix}1 & 0 & 10\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}\cos 45 & \sin 45 & 0\\ -\sin 45 & \cos 45 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}\cos 45 & -\sin 45 & 0\\ \sin 45 & \cos 45 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0 & -10\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}$$

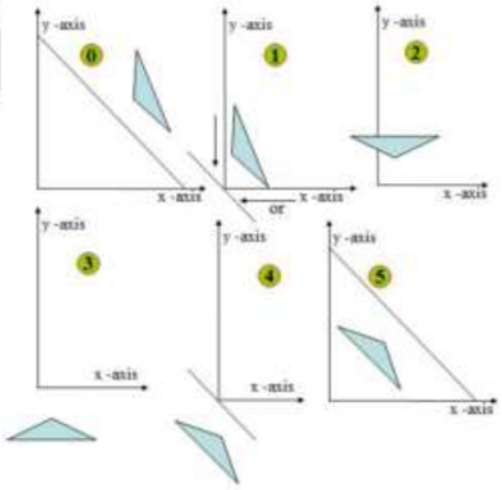
Addition x-intercept · *CW Rotation* · *Reflection about x-axis* · *CCW Rotation* · *Reduce x-intercept*

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

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

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

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



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

$$P' = Com \times P$$

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

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

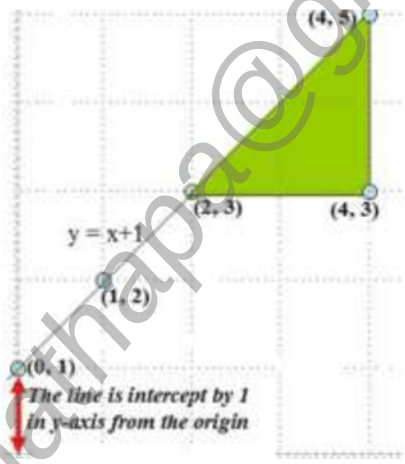
Hence, the final coordinates are (3, 4), (4, 3) & (1, 4).

Note: The following Transformation matrix also could be used.

$$T = T_{(0,10)}\, R_{(\theta=-45)}\, R_X\, R_{(\theta=45)}\, T_{(0,-10)}$$

4. Magnify the triangle with vertices A(0,0), B(1,1) and C(5,2) to twice its size while keeping C(5,2) fixed.
   Steps:
   1) First, translate the triangle by Tx= -5 and Ty=-2
   2) Then Magnify the triangle by twice its size i.e. Scaling Factor S_x and S_y is equal to 2.
   3) Again translate the triangle by Tx= 5 and Ty= 2.

5. Reflect the object (2,3), (4,3), (4,5) about lin y=x+1.
   Solution:



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) = $\text{Tan}^{-1}(m) = \text{Tan}^{-1}(1) = 45^0$

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: Com = $T' R_\theta' R_{fx} R_\theta T$

Thus, composite matrix is given by

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

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

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

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

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

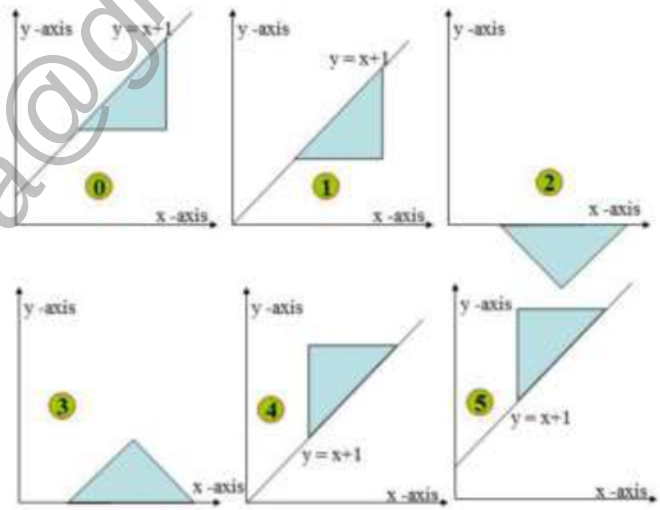
Or, this can also be directly computed as below

Also, the composite matrix can be calculated as:

$$= \begin{pmatrix}\frac{-(m^2-1)}{(m^2+1)} & \frac{2m}{(m^2+1)} & \frac{2mc}{(m^2+1)}\\ \frac{2m}{(m^2+1)} & \frac{(m^2-1)}{(m^2+1)} & \frac{2c}{(m^2+1)}\\ 0 & 0 & 1\end{pmatrix}$$

Where, m =1, c=1

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



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).

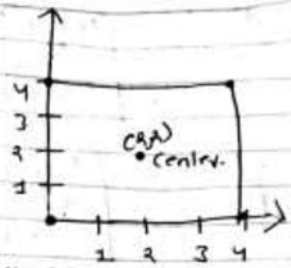
6. Reflect the object (2,3), (4,3), (4,5) about lin y=5x+1.
   Solve yourself

7. Reflect the object (2,3), (4,3), (4,5) about lin y=5x
   Solve yourself

8. Find the transformation matrix that transforms the square ABCD whose center is at (2,2) is reduced to half of its size, with center still remaining at (2,2). The coordinates of the square ABCD are A(0,0), B(0, 4), C(4, 4) and D(4,0). Find the coordinates of the new square.
Soluton

Soln:

Given points of square
A(0,0), B(0,4)
C(4,4), D(4,0)



Now, we have to reduce it to half, so scaling factor
$S_x = S_y = \frac{1}{2}$

We have scaling matrix as

$$S = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

So Applying the transformation, we get

$$A' = SA = \begin{bmatrix} \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

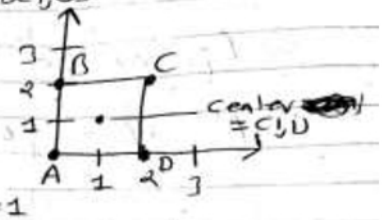
$$B' = SB = \begin{bmatrix} 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 4 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}$$

$$C' = SC = \begin{bmatrix} 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 4 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}$$

$$D' = SD = \begin{bmatrix} 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix}$$

∴ New points after scaling are
A(0,0), B(0,2) C(2,2) D(2,0)

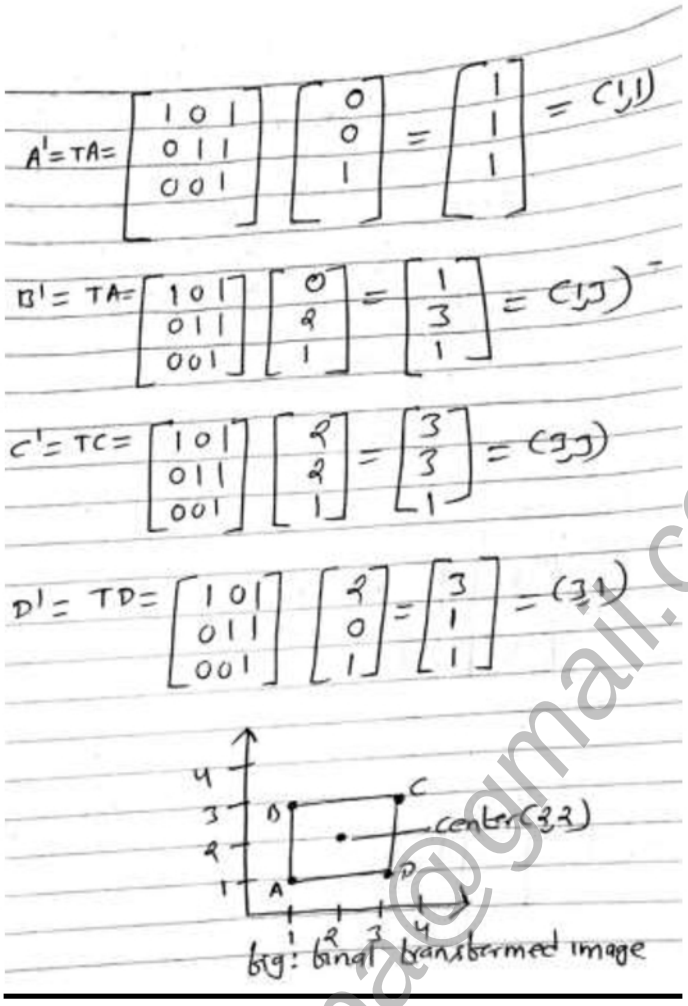
So finally translate so as to shift the center (1,1) to (2,2)
i.e $t_x = 1$ and $t_y = 1$



The translation matrix is given by

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

So applying this transformation we get



9. The reflection along the line y=x is equivalent to reflection along X axis followed by counter clockwise rotation by θ degree. Find the θ.

Solution:

The transformation matrix for y = x is

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

Reflection about X – axis is

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

Counter clockwise rotation about θ

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

Therefore the composite transformation matrix is given by

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

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

it is given that,

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

i.e. sin θ = 1 i.e. θ = 90

Composite Transformation

- In many cases we need to apply several transformations (scaling, rotation, translation) to one object.
- It is possible to set up a matrix for any sequence of transformations as a composite transformation matrix by calculating the matrix product of the individual transformations.
- The basic purpose of composite transformations is gain efficiency by applying a single composed transformation to a point, rather than applying a series of transformations, one after another
- We can set up matrix for any sequence of transformations as a composite transformation matrix by calculating matrix product of individual transformation.
    i. Successive translations are additive
    It two successive translations are applied then they are additive.
    Proof:
    If two successive translation vectors (t_{x1}, t_{y1}) and (t_{x2}, t_{y2}) are applied to a coordinate position P, the final transformed location P' is calculated with the following composite transformation as

$$T' = T_{(x2,y2)} \cdot T_{(x1,y1)}$$

$$\begin{bmatrix} 1 & 0 & Tx2 \\ 0 & 1 & Ty2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & Tx1 \\ 0 & 1 & Ty1 \\ 0 & 0 & 1 \end{bmatrix}$$

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

$$= T(tx_1 + tx_2, ty_1 + ty_2)$$

$T(t_{x2},t_{y2}).T(t_{x1},t_{y1}) = T(t_{x1}+t_{x2}, t_{y1}+t_{y2})$, which demonstrates that two successive translations are additive.

ii. Successive rotations are additive
If two successive rotations are applied, then they are additive.

Proof:
Let P be point anticlockwise Rotated by angle θ_1 to point P' and again let P' be rotated by angle θ_2 to point P'', then the combined transformation can be calculated with the following composite matrix
as

$$T = R(\theta_2)\, R(\theta_1)$$

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

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

$$\begin{bmatrix} \cos(\theta 1+\theta 2) & -\sin(\theta 1+\theta 2) & 0 \\ \sin(\theta 1+\theta 2) & cos\theta 1+\theta 2) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

i.e. R(θ_2) **R** $(\theta_1) = R(\theta_1 + \theta_2)$, which demonstrates that two successive rotations are additive.

iii. Successive Scaling are multiplication

If two successive Scaling are applied, then they are multiplicative/commutative.

Proof:

Let point P is first scaled with scaling factor Sx1, Sy1 to Point P' and again let P' be scaled by scaling factor Sx2,Sy2 to Point P'', then the combined transformation can be calculated with the following composite matrix

$T = S(Sx2, Sy2)\ S(Sx1, Sy1)$

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

i.e. S (sx2,sy2) S(sx2,sy2) = S(sx1.sx2, sy1.sy2), which demonstrates that two successive scaling are multiplicative.

The Viewing Pipeline

- *World coordinate system (WCS)* is the co-ordinate system where we define the object to be displayed. A finite region in the WCS is called the *Window* i.e. A world coordinate area selected for display is called window.
- The corresponding coordinate system on the display device where the image of the picture is displayed is called the *physical/device coordinate system*.
- An area on display device to which window is mapped is called viewport.
- The window defines what is to be viewed whereas the viewport defines where it is to be displayed.
- Window can have any shape (circle, polygon) however some graphics package provides window and viewport operations on standard rectangle only.
- Window deals with object space whereas viewport deals with image space.
- The process of mapping the world co-ordinate scent to device co-ordinate is called viewing transformation or window to viewport transformation or windowing transformation.

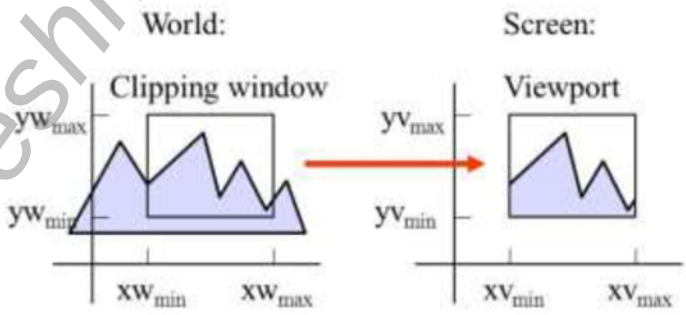


Fig: World Object Space Fig: Viewport Image Space

Fig: Mapping of picture section falling under rectangular Window onto a designated rectangular viewport

- Transformations from world to device coordinates involve translation, rotation and scaling operations, as well as procedures for deleting those parts of the picture that are outside the limits of a selected display area i.e. clipping.
- To make the viewing process independent of the requirements of any output device, graphics systems convert object descriptions to normalized coordinates.

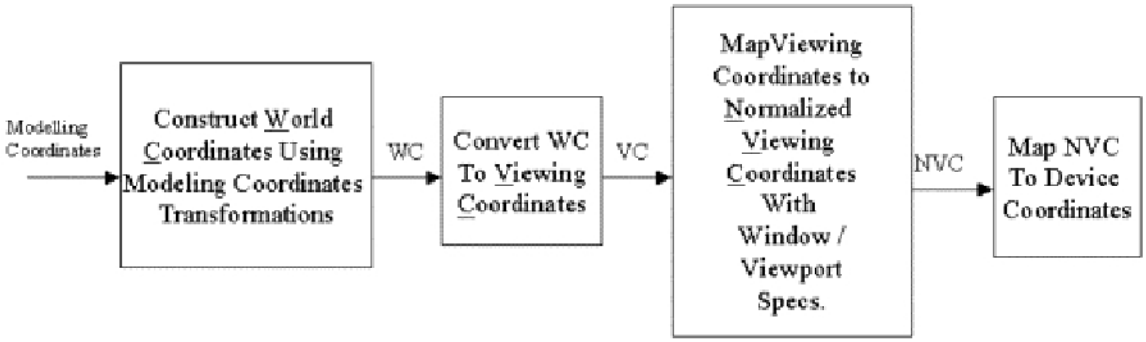


Fig: The 2D Viewing Transformation Pipeline

In the above figure:

1. First we construct the scene in world co-ordinate system by applying modelling transformation.
2. Then we convert the world coordinate system to viewing coordinate system. (clipping window)
3. We then define a viewport in standard normalize coordinates (in the range 0-1). Different display device has different co-ordinate system, so we standardize the world coordinate system by mapping to normalized viewing coordinate system using window-view port specification.
4. Finally, all parts of scene that lie outside the viewport are clipped(viewport), and the contend to viewport are transformed to device coordinate.

Note:

Modelling Coordinate: It is used to define coordinates that are used to construct the shape of individual parts (objects) of a 2D scene

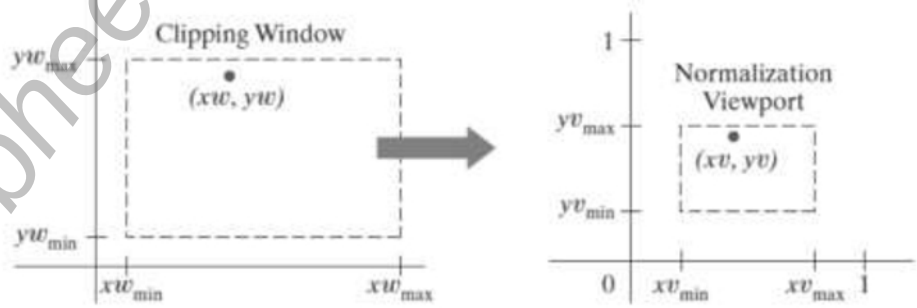
World Coordinate: It is Used to organize the individual objects (points, lines, circles etc) into a 3D scene. A scene is made up of a collection of objects.

Application

1. By changing the position of the viewport, we can view objects at different positions on the display area of an output device.
2. By varying the size of viewports, we can change size of displayed objects.
3. Zooming effects can be obtained by successively mapping different-sized windows on a fixed-sized viewport
4. Panning effects (Horizontal Scrolling) are produced by moving a fixed-size window across the various objects in a scene.

Window to Viewport/Device Coordinate Transformation

- A window can be specified by four world coordinates: x_{wmin}, x_{wmax}, y_{wmin} and y_{wmax}.
- Similarly, a viewport can be described by four device coordinates: x_{vmin}, x_{vmax}, y_{vmin} and y_{vmax}



- Then this window to viewport transformation can be performed by performing following three steps
  a. Translate object along with window such that the lower left corner of the window is at the origin.

     i..e. T ($-x_{wmin}$, $-y_{wmin}$).

  b. Scale the object and the window such that window has the same dimension as that of a viewport.

i.e. S (S_x, S_y).

c. Finally, apply another translation to move the viewport to its original position on the screen.
i.e. T(x_{vmin}, y_{vmin}).

Therefore, the composite transformation matrix for performing the above transformation is given as
T = T(x_{vmin}, y_{vmin}). S (S_x, S_y). T (-x_{wmin}, -y_{wmin}).

Here, s_x and s_y are scaling factors, where;

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

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

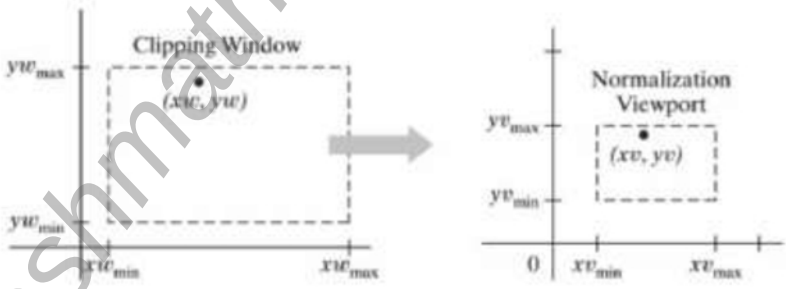
Now, substituting these values;

$$T_{wv} = \begin{bmatrix} 1 & 0 & -x_{wmin} \\ 0 & 1 & -y_{wmin} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{x_{vmax} - x_{vmin}}{x_{wmax} - x_{wmin}} & 0 & 0 \\ 0 & \frac{y_{vmax} - y_{vmin}}{y_{wmax} - y_{wmin}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & x_{vmin} \\ 0 & 1 & y_{vmin} \\ 0 & 0 & 1 \end{bmatrix}$$

Is the required window-to-viewpoint transformation.

Numerical Example

1. Window port is given by (100,100,300,300) and viewport is given by (50,50,150,150). Convert the window port coordinate (200,200) to the viewport coordinate.
   Solution: Using second technique



Here (Xw_{min}, Yw_{min}) = (100, 100) and (Xw_{max}, Yw_{max}) = (300, 300)
(Xv_{min}, Yv_{min}) = (50, 50) and (Xv_{max}, Yv_{max}) = (150, 150)
P(Xw,Yw)=(200,200) P'(Xv,Yv)=?

Then, the composite transformation matrix for transforming the window co-ordinate to viewport coordinate is given as

$$T = T_{(50, 50)}\, S(s_x, s_y) T_{(-100, -100)} \text{---------1}$$

we know

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

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

i.e. S_x = (150- 50)/ (300 - 100) = 0.5
S_y = (150- 50)/ (300 - 100) = 0.5

From above equation 1

$$T=\begin{bmatrix}1 & 0 & 50\\ 0 & 1 & 50\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}0.5 & 0 & 0\\ 0 & 0.5 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & -100\\ 0 & 1 & -100\\ 0 & 0 & 1\end{bmatrix}$$

$$\begin{bmatrix}1 & 0 & 50\\ 0 & 1 & 50\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}0.5 & 0 & -50\\ 0 & 0.5 & -50\\ 0 & 0 & 1\end{bmatrix}$$

$$\begin{bmatrix}0.5 & 0 & 0\\ 0 & 0.5 & 0\\ 0 & 0 & 1\end{bmatrix}$$

Now, we know

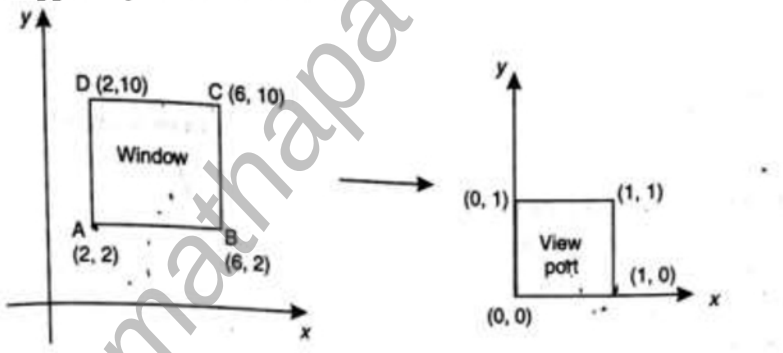
$$P' = T.P$$

$$\begin{bmatrix}0.5 & 0 & 0\\ 0 & 0.5 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}200\\ 200\\ 1\end{bmatrix}$$

$$\begin{bmatrix}100\\ 100\\ 1\end{bmatrix}$$

hence, the transformed viewport coordinate is P'(100,100).

2. Find the normalization transformation for window to viewport which uses the rectangle whose lower left corner is at (2,2) and upper right corner is at (6,10) as a window and the viewport that has lower left corner at (0,0) and upper right corner at (1,1)



The composite transformation matrix for transforming the window co-ordinate to viewport coordinate is given as

$$T = S(s_x, s_y)T(-2, -2)$$

Now we know,

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

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

i.e. $S_x = (1-0)/(6-2) = 0.25$
$S_y = (1-0)/(10-2) = 0.125$

Then from above, we get required transformation matrix as below

$$T=\begin{bmatrix}Sx & 0 & 0\\ 0 & Sy & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & tx\\ 0 & 1 & ty\\ 0 & 0 & 1\end{bmatrix}$$

$$\begin{bmatrix} 0.25 & 0 & 0 \\ 0 & 0.125 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -2 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0.25 & 0 & -0.5 \\ 0 & 0.125 & -0.25 \\ 0 & 0 & 1 \end{bmatrix}$$

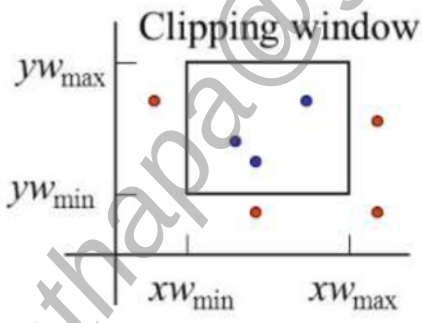
Clipping

- The process of discarding those parts of a picture which are outside of a specified region or window is called clipping.
- The procedure using which we can identify whether the portions of the graphics object is within or outside a specified region or space is called clipping algorithm.
- Clipping is necessary to remove those portions of the objects which are not necessary for further operation's. excludes unwanted graphics from the screen. So there are three cases
    i. The object may be completely outside the viewing area defined by the window port.
    ii. The object may be seen partially in the window port.
    iii. The object may be seen completely in the window port

  For case i and ii, clipping operation is necessary but not for case iii.

Point Clipping

- Point clipping is the process of removing all those points that lies outside a given region or window.
- Let xw_{min} and xw_{max} be the edge of the boundaries of the clipping window parallel to Y axis and yw_{min} and yw_{max} be the edge of the boundaries of the clipping window parallel to X axis as shown in figure below.



- So any point P(x,y) of world coordinate can be saved for display if the following conditions are satisfied

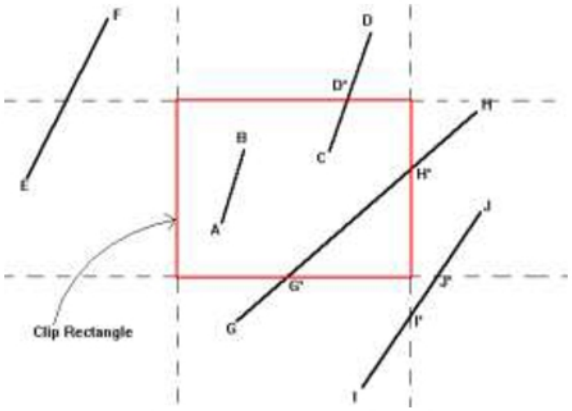
$$xw_{min} \le x \le xw_{max}$$

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

  So in the above figure all the red points don't satisfy the above conditions so are selected for clipping i.e. discarded.

Line Clipping

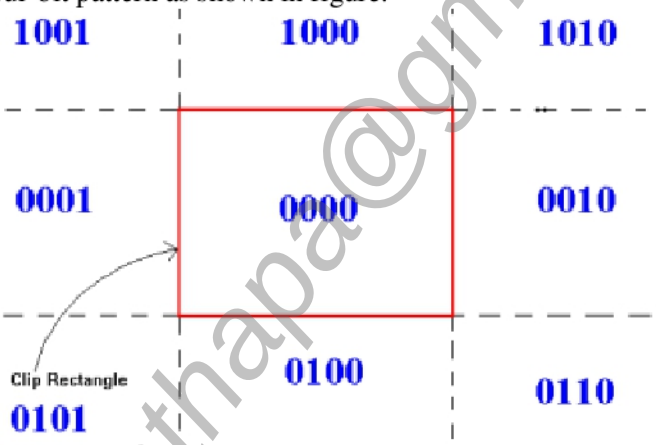
- In line clipping, a line or part of line is clipped if it is outside the window port.
- All the line segment falls into one of the following clipping cases.
    a. The line is **totally outside the window** port so is directly clipped.
    b. The line is partially clipped if a part of the line lies outside the window port i.e. **partial visible**
    c. The line is not clipped if all whole line lies **totally inside the window** port.



So in the above figure, line FE require total clipping, CD, GH, IJ require partial clipping and AB requires no clipping.

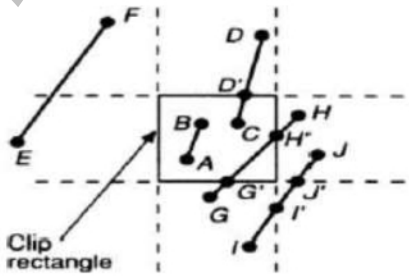
Cohen –Sutherland Line Clipping Algorithm

- It is one of the oldest and most popular line-clipping algorithms based on divide and conquer approach.
- In this method, coordinate system is divided into nine regions.
- 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).
- Each region is assigned a four-bit pattern as shown in figure.



- Any point inside the clipping window has a region code 0000.
- Any endpoint (x, y) of a line segment, the code can be determined as follows:
  - If $x < xw_{min}$, first bit is 1, (Point lies to left of window(**Left**)) (0th bit) Otherwise 0.
  - If $x > xw_{max}$, second bit is 1, (Point lies to right of window(**Right**)) (1st bit), otherwise 0.
  - If $y < yw_{min}$, third bit is 1, (Point lies to below window(**Bottom**)) (2nd bit), otherwise 0.
  - If $y > yw_{max}$, fourth bit is 1, (Point lies to above window(**Top**)) (3rd bit), otherwise 0.

Example:



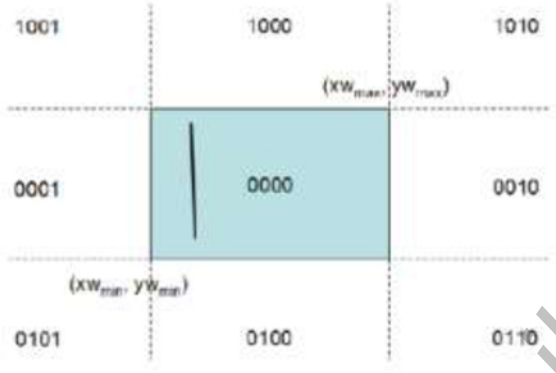
Line	Code 1	Code 2
A⇒B	0000	0000
C⇒D	0000	1000
E⇒F	0001	1001
G⇒H	0100	0010
I⇒J	0100	0010

Algorithm

1. Given a line segment with end points P1=(x1,y1) and P2(x2,y2), compute 4 bit region code for each end point.
2. To clip a line, first we have to find out which regions its two endpoints lie in.

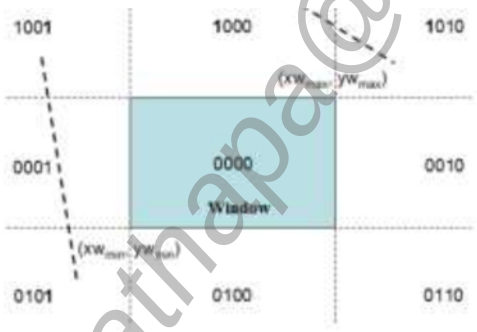
Case 1:

- If both end point code is 0000, then the line segment is completely inside the window. i.e. if bitwise OR of the codes yields 0000, then the line segment is accepted for display.



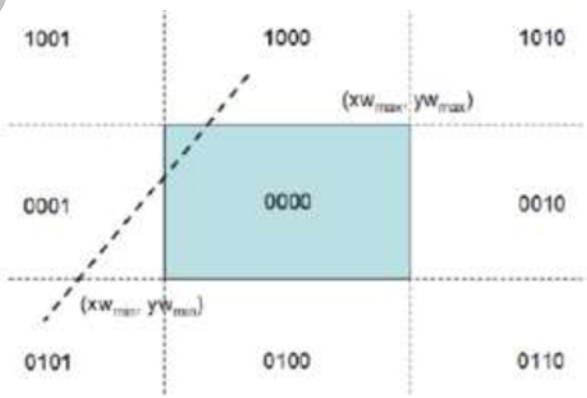
Case 2:

- If the two end point region code both have a 1 in the same bit position, then the line segment lies completely outside the window, so discarded. i.e. i.e. if bitwise anding of the codes not equal to 0000, then the line segment is rejected.



Case 3:

- If the line can't be identified as completely inside or outside (i.e. partially visible) then, we have to compute the intersection of the line segment with window boundary, and discard the portion of the segment that falls completely outside the window. For this, take the end point (say P2) which is outside boundary (i.e. end point having region code! =0000) and read its region code in order Top-bottom-right-left. When a set bit is found, compute the intersection point 'I' of the corresponding window edge with the line P1 and P2. Replace P2 with I and assign a new four-bit code to the intersection point and repeat until either case1 or case2 are satisfied



For finding the intersection point we can use the slop intercept formula as

$$y-y_1 = m\,(x-x_1)$$

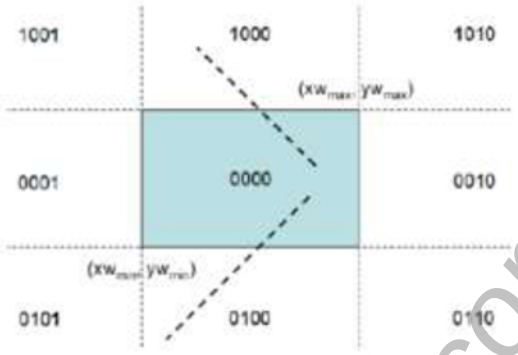
where $m = (y_2-y_1)/(x_2-x_1)$

Condition1:

i. If the intersection is with horizontal boundary

$y = y_{min}$ or y_{max}.

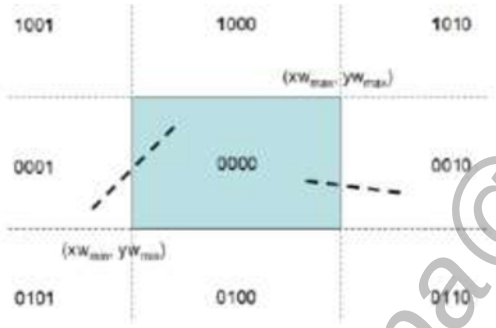
$x = x_1 + 1/m * (y - y_1)$,



ii. If the intersection is with vertical boundary

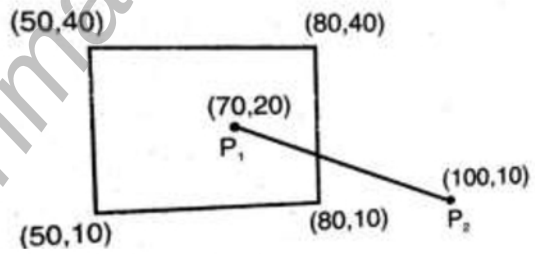
$x = x_{min}$ or x_{max}

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



Numerical Example

1. Use the Cohen –Sutherland algorithm to clip the line P1(70,20) and P2(100,10) against a window lower left hand corner (50,10) and upper right hand corner (80,40)
   Solution:



Assigning 4 bit binary outcode code to the two end point

P1 = 0000 and P2 = 0010

P1∨ P2 = 0000 ∨ 0010 = 0010, since P1∨ P2!=0000,hence the two point doesn't lie completely inside the window.

P1∧ P2 = 0000 ∧ 0010 = 0000, since P1 ∧ P2=0000, hence line is partially visible.

Now, For, finding the intersection of P1 and P2 with the boundary of Window,

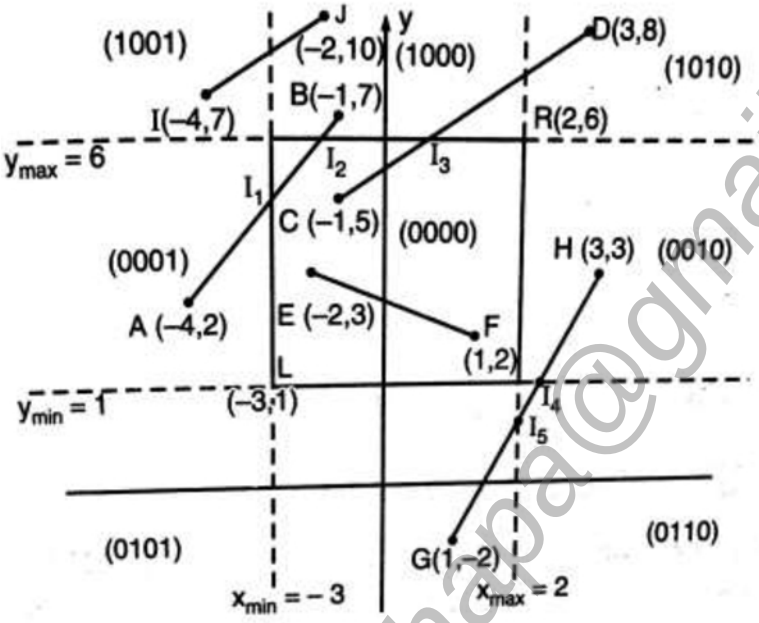
Starting from point p2(because. p2 lie outside the window), we get

P1 = (x1,y1) = (70,20) and p2 = (x2,y2) = (100,10)
Therefore, Slope M = (10-20)/(100-70) = -1/3
We have to find the intersection with right edge of window, here x=80, y=?
We have (p2 selected since outside boundary)

$$y-y2= m(x-x2)$$
$$y = y2 + m(x-x2)$$
$$= 10 +(-1/3)(80-100)$$
$$= 10 + 6.67$$
$$= 16.67$$

Thus the intersection point p3 = (80,16.66)
So discarding the line segment that lie outside the boundary i.e. p_3p_2 we get new line P_1P_3 with coordinate P1(70, 20) and P3(80,16.67) both of which have region code 0000. Hence line segment (p1, p3) is selected for display.

2. Let R be the rectangle window whose lower left hand corner is (-3,1) and upper right hand corner is (2,6).
   a. Find out the region code for end points in the following figure
   b. Find the clipping categories for the line segments
   c. Use cohen Sutherland algorithm to clip the line segment.



Solution:

i.

The region code is set according to following rule

- Any endpoint (x, y) of a line segment, the code can be determined as follows:
  - If $x < x_{min}$, first bit is 1, (Point lies to left of window(**Left**)) (0th bit) Otherwise 0.
  - If $x > x_{max}$, second bit is 1, (Point lies to right of window(**Right**)) (1st bit), otherwise 0.
  - If $y < y_{min}$, third bit is 1, (Point lies to below window(**Bottom**)) (2nd bit), otherwise 0.
  - If $y > y_{max}$, fourth bit is 1, (Point lies to above window(**Top**)) (3rd bit), otherwise 0.

Region Code:
A = (0001)
B=(1000)
C=(0000)
D=(1010)
E=(0000)
F=(0000)
G=(0100)
H=(0010)
I=(1001)
J=(1000)

ii.

To get appropriate categories, we test region code
Category 1: Inside Window i.e. visible
Line segment EF because the region code for both end point is 0000 i..e. E ∨ F = 0000 holds so is selected for display

Category 2: Outside Window i.e. Not visible
Line segment IJ because the region code of I and J have 1 at same bit position i.e. I ∧ J != 0000, so is selected for clipping i.e. discarded

Category 3: Partial visible
Line segment AB, CD and GH since A ∧ B=0000, similarly C ∧ D = 0000 and similarly G ∧ H=0000. So become candidate for clipping.

iii.

So line AB, CD and GH need to be clipped

For line AB
We can start from A Or B, let start from A.
We have to find the intersection with left edge of window, here x=-3, y=?
We have (A selected)

Slope m= (7 – 2) /(-1 + 4) = (5/3)

y = y1 + m(x-x1)
= 2 + (5/3) (-3 +4)
= 3.667

Thus the intersection point I_1 = (-3, 3.66) and region code is 0000
Hence we clip AI_1
Now we work on I_1B
Since $I_1 \wedge B$ = 0000 * 1000 = 0000 hence is again partially visible.
Now we have to find the intersection with top edge of window, here y= 6, X=? (B selected)
Slope M = 5/3
And
X=x1 + 1/m (y- y1)
= -1 + 3/5 (6-7)
=1.6

Thus the intersection point I_2 = (-1.6,6) with region code 0000
Hence we clip I_2B
Now we work on I_1I_2
Since the region code of I_1 and I_2 both is 0000 i.e. $I_1 \vee I_2$ = 0000, hence both point lie inside the window. Hence selected for display.

For Line GH
We can start with G or H, for now, let start from G. (see case 3: opcode of G = 0100 i.e. bottom edge)
We have to find the intersection with the bottom edge of window, here y=1 , x=?
Slope M = (3+2) / (3-1)= 5/2
And
X=x1 + 1/m (y- y1)
= 3 + 2/5 (1-3)
=2.2
Thus the intersection point I_4 = (2.2, 1) and region code is 0010
Hence GI_4 is clipped
Now we work on I_4H
since $I_4 \wedge H$ = 0010 ∧ 0010 = 0010 i.e. !=0000 hence both point lie outside the window hence clipped i.e. not displayed.

For Line CD
Starting from point D (because D lie outside the window)
We have to find the intersection with the top edge of window, here y = 6, x=?

Slope M = (8-5) / (3+1)= 3/4
And
X=x1 + 1/m (y- y1)
= 3+ 4/3 (6-8)
=033
Thus the intersection point I_3 = (0.33, 6) and region code is 0000
Hence I_{3D} is clipped.
Now we work on CI_3
since both the region code of C and I_3 is 0000 i.e C ∨ I_3 = 0000, hence the segment CI_3 is displayed.

Polygon Clipping: Sutherland – Hodgeman Algorithm

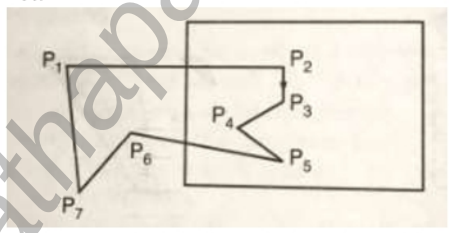
- The Sutherland–Hodgeman algorithm is used for clipping polygons. Similar to Cohen Sutherland algorithm, this algorithm is also based on divide and conquer approach.
- This algorithm clips the polygon against four boundary edge of clipping window in succession.
- Let a polygon to be clipped consists of ordered sequence of vertices A, B, C, D…… then edges will be AB, BC, CD……, then in this algorithm, edges will be processed in the order of the vertex pair.

Algorithm

- The clipping algorithm will be called once for each ordered sequence (clockwise or anticlockwise direction) of vertex of the polygon and each call will return either no vertex, or original vertex or one or more vertex.
- We have to consider the following four cases when we are clipping a polygon with respect to any particular (**left**, right, top and bottom) window boundary. *(left boundary considered below)*

Case-I

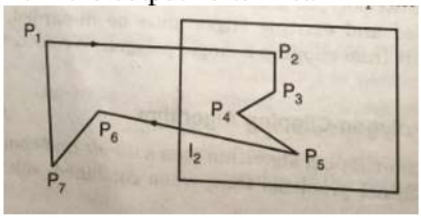
- If both first and second vertex are inside the window boundary, then we have to store the second vertex only in the output vertex list.



Here, for vertex P2 and P3, since both are inside the window boundary, we store P3 in the output vertex list.

Case-II

- If the first vertex is inside the window and second vertex is outside the window boundary, then we have to store only intersection point in the output vertex list.



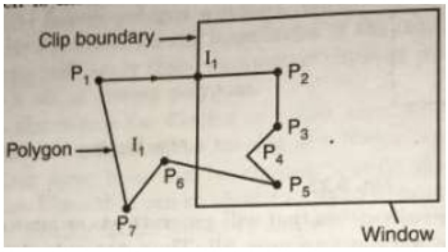
Here, for vertex P5 and P6, since P5 is inside and P6 is outside the window boundary, we store I_2 in the output vertex list.

Case-III

- If both first and second vertex lie outside the window boundary, then no vertex is stored in the output list.
  For example, in above figure for vertex P_6 and P_7, nothing is stored in the output vertex list.

Case-IV

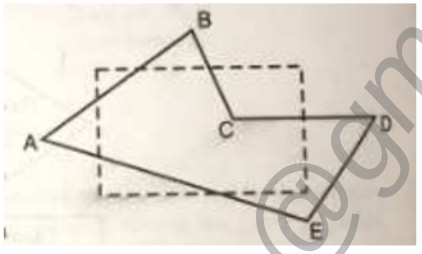
- If the first vertex is outside the window boundary and the second vertex is inside the window, then the intersection point of the polygon with the boundary edge of window and the vertex which is inside the window is stored in the output vertex list.



Here, fore vertex P1 and P2, since p1 is outside and p2 is inside the window boundary, we store I_1P_2 in the output vertex list.

- Once all vertices have been considered for one clip window boundary, the output list of vertices is clipped against the next window boundary in succession for each of the four window boundary.

Explain the steps for clipping the polygon given in the below figure against the clipping window using Sutherland Hodgeman algorithm.



Solution:

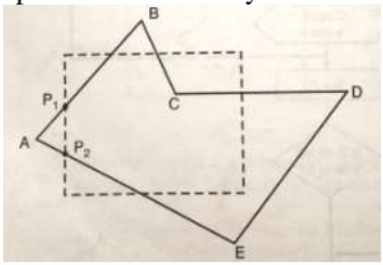
Here order vertex list will be A, B, C, D, E and edge will be AB, BC, CD, DE and EA.

From Sutherland Hodgeman algorithm, assuming starting vertex as A, we have the following case

Case	Vertex1	Vertex2	Output Vertex
1	Inside	Inside	Vertex2
2	Inside	Outside	Intersection
3	Outside	Outside	None
4	Outside	Inside	Intersection and Vertex2

Step-1: Clipping against Left window edge/boundary

We apply each edge of polygon with respect to left boundary of window.



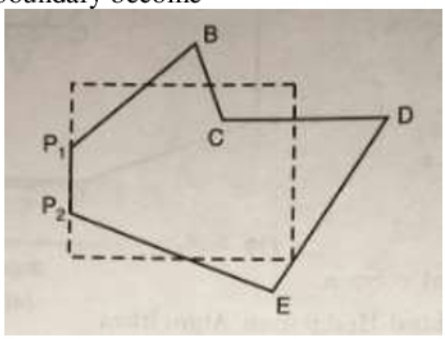
Processed Edge(vertex1, Vertex2)	Output Vertex	Case
A B	P1 B	4
B C	C	1
C D	D	1
D E	E	1

E A	P2	2

Where, P1 is intersection point between A and B, P2 is intersection point between A and E.

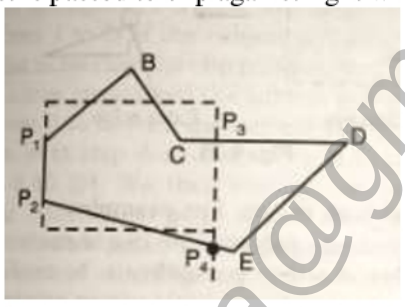
Then output vertex list = (P_1, B, C, D, E, P2)

So our clipped polygon against left boundary become



Step-2: Clipping against Right window edge/boundary

Now, the updated output vertex list is passed to clip against right window boundary.

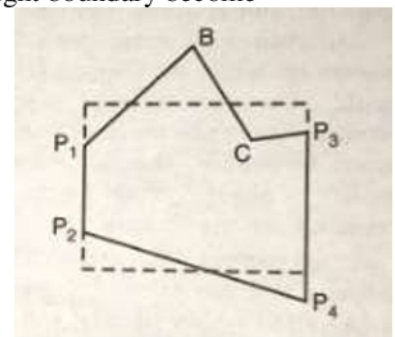


Processed Edge(vertex1, Vertex2)	Output Vertex	Case
P1 B	B	1
B C	C	1
C D	P_3	2
D E	.....	3
E P_2	P_4 P_2	4
P_2 P_1	P_1	1

Where, P_3 is intersection point between C and D, P_4 is intersection point between E and P_2.

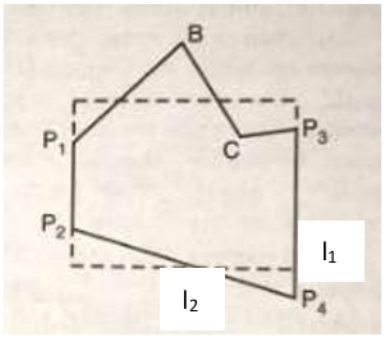
Then output vertex list = (B, C, P_3, P_4, P_2, P_1)

So our clipped polygon against Right boundary become



Step-3: Clipping against Bottom window edge/boundary

Now, the updated output vertex list is passed to clip against bottom window boundary.

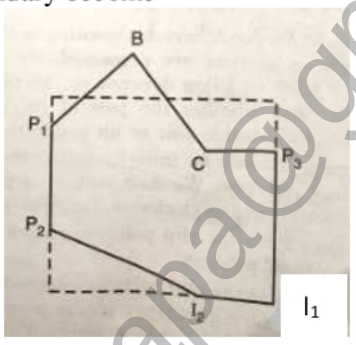


Similarly,

Processed Edge(Vertex1, Vertex2)	Output Vertex	Case
$P_2 P_1$	P1	1
P_1 B	B	1
B C	C	1
C P_3	P_3	1
$P_3 P_4$	I_1	3
$P_4 P_2$	I_2 P_2	4

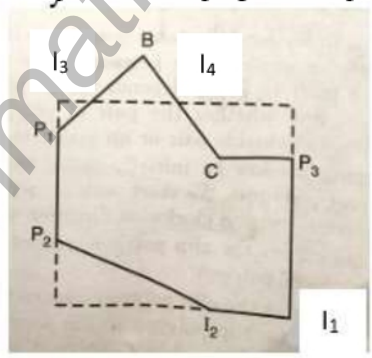
Where, I_1 is intersection point between P_3 and P_4, I_2 is intersection point between P_4 and P_2.

Then output vertex list = (P_1, B, C, P_3, I_1, I_2, P_2)
So our clipped polygon against Bottom boundary become



Step-4: Finally clipping against Top window edge/boundary

Now, the updated output vertex list is passed to clip against Top window boundary.

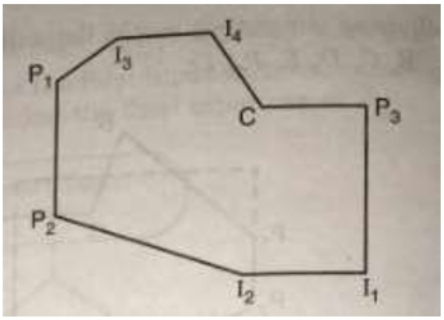


Similarly,

Processed Edge(Vertex1, Vertex2)	Output Vertex	Case
$P_2 P_1$	P1	1
P_1 B	I_3	2
B C	I_4 C	4
C P_3	P_3	1
$P_3 I_1$	I_1	1
I_1 I_2	I_2	1
$I_2 P_2$	P_2	1

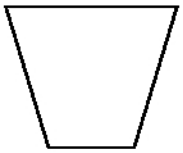
Where, I_3 is intersection point between P_1 and B, I_4 is intersection point between B and C.

Then output vertex list = (P_1, I_3, I_4, C, P_3, I_1, I_2, P_2)
So our clipped polygon against Bottom boundary become



Limitations with Sutherland Hodgeman Algorithm

- A polygon is **convex** if the line joining any two interior points of the polygon lies completely inside the polygon otherwise it is called **concave** polygon.



Polygon-1

Polygon-2

- In the above figures, polygon-1 is convex because any line drawn from any two vertex of polygon-1 lies inside the polygon but polygon 2 is concave because some line drawn from two vertex of polygon lies outside the polygon, for example, line from vertex E1 and E2 lie outside the polygon.
- **All convex polygons are correctly clipped by this algorithm but some concave polygon may be displayed with extraneous lines**. For example,

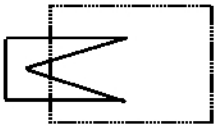


Fig: Before Clipping

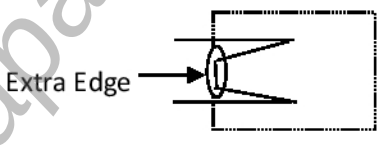


Fig: After Clipping

Assignment-3

1. Derive Scaling Matrix about any arbitrary point.
2. Derive Rotation matrix (clockwise direction) about any arbitrary point.
3. Suppose there is a rectangle ABCD whose coordinate A(1,1), B(4,1), C(4,4), D(1,4) and the window coordinate are (2,2), (5,2), (5,5), (2,5) and the given view port location is (0.5, 0), (1,0), (1,0.5), (0.5,0.5). calculate the viewing transformation matrix.