UNIT-2

2-D Transformation -

① Translation -

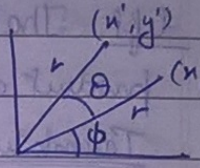
(t_x, t_y) = translation vector or shift vector

$\boxed{x' = x + t_x}$, $\boxed{y' = y + t_y}$

$P_1' = \begin{bmatrix} x_1' \\ y_1' \end{bmatrix}$, $P_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$, $T = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$ $\Rightarrow \boxed{P_1' = P_1 + T}$

$P_2' = \begin{bmatrix} x_2' \\ y_2' \end{bmatrix}$, $P_2 = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$, $T = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$ $\Rightarrow \boxed{P_2' = P_2 + T}$

② Rotation -



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

$\Rightarrow x' = r\cos\phi\cos\theta - r\sin\phi\sin\theta$, $\Rightarrow y' = r\sin\phi\cos\theta + r\cos\phi\sin\theta$

Also, $x = r\cos\phi$ and $y = r\sin\phi$

$\Rightarrow \boxed{x' = x\cos\theta - y\sin\theta}$, $\Rightarrow \boxed{y' = y\cos\theta + x\sin\theta}$

Also, $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$ $\Rightarrow \boxed{P' = R \cdot P}$

Counter-clockwise rotation

③ Scaling -

$\boxed{x' = x \cdot S_x}$, $\boxed{y' = y \cdot S_y}$

(S_x, S_y : scaling factor)

If $S_x = S_y$ $\Rightarrow$ uniform scaling

If $S_x \neq S_y$ $\Rightarrow$ non-uniform scaling

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

④ Homogenous coordinate system -

We cannot do combine transformations using 2×2 matrix with initial coordinates. A more efficient approach would be to combine the transformations so that the final coordinate positions are obtained directly from the initial coordinates by using a 3×3 matrix, thereby eliminating the calculation of intermediate coordinate values. Therefore,

$P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$, $P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$, $T = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$, $R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $S = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$

⑤ Composite Transformation -

We can set up a matrix for any sequence of transformations as a composite matrix transformation matrix by calculating the matrix product of the individual transformations. 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 transformations by multiplying matrices in order from right to left. That is, each successive transformation matrix premultiplies the product of the preceding transformation matrices.

Translation -

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

Scaling -

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

Rotation -

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

⑥ Reflection -

(1) About x-axis,

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

(2) About y-axis,

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

(3) About axis perpendicular to XY plane,

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

(4) About $x=y$.

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

(5) About $x=-y$.

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

Pure rotation, $\theta = -90°$

⑦ Shearing - distortion of shape of the object

(1) About x-axis, (2) About y-axis (3) About x-direction with reference to y-axis (4) About y-direction with reference to x-axis

$$\begin{bmatrix} 1 & Sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 \\ Sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & Sh_x & -Sh_x\, y_{ref} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 \\ Sh_y & 1 & -Sh_y\, x_{ref} \\ 0 & 0 & 1 \end{bmatrix}$$

⑧ Inverse Transformation -

Translation - (-T) Rotation (-θ) - Scaling ($\frac{1}{s}$) -

$$\begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1/s_x & 0 & 0 \\ 0 & 1/s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Windowing & Clipping -

① World coordinate system - Base reference system for the overall model, to which all other model coordinates relate.

Screen coordinate system - This 2D coordinate system refers to the physical coordinates of the pixels on the computer screen, based on current screen resolution.

A world-coordinate area selected for display is called a window.

An area on a display device to which a window is mapped is called a viewport.

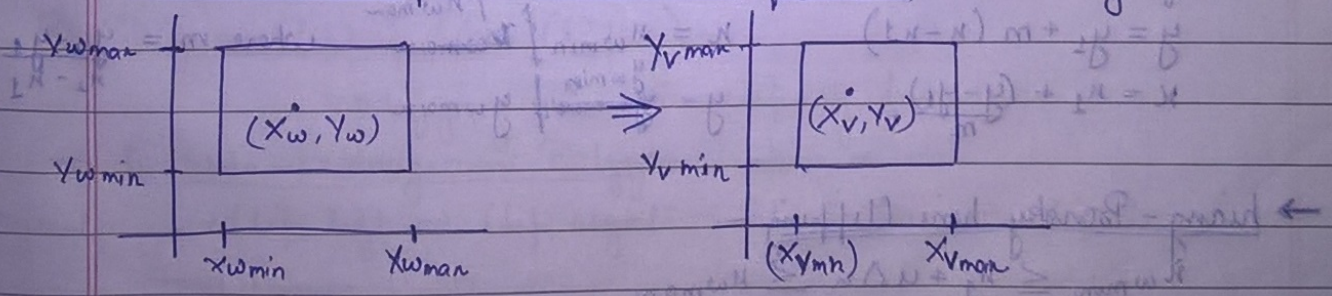
The window defines what is to be viewed whereas the viewport defines where it is to be displayed.

② Viewing Transformation -

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

2D viewing transformation referred as window-to-viewport transformation or the windowing transformation

Window-to-viewport coordinate transformation - (Scaling then translation)



$$\frac{X_w - X_{wmin}}{X_{wmax} - X_{wmin}} = \frac{X_v - X_{vmin}}{X_{vmax} - X_{vmin}} \qquad \frac{Y_w - Y_{wmin}}{Y_{wmax} - Y_{wmin}} = \frac{Y_v - Y_{vmin}}{Y_{vmax} - Y_{vmin}}$$

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

where $S_x = \frac{X_{vmax} - X_{vmin}}{X_{wmax} - X_{wmin}}$ where $S_y = \frac{Y_{vmax} - Y_{vmin}}{Y_{wmax} - Y_{wmin}}$

③ Clipping –

Any procedure that identifies those portions of a picture that are either inside or output of a specified region of space is refered to as a clipping algorithm, or simply clipping. The region against which an object is to be clipped is called a clip window.

There are five types of clipping:- Point Clipping, line Clipping (Straight line segments), Area Clipping (polygons), Curve Clipping and Text Clipping.

④ line Clipping –

$x = x_1 + u(x_2 - x_1)$ & $y = y_1 + u(y_2 - y_1)$ (line → (x_1, y_1) to (x_2, y_2))

If $0 \le u \le 1$ then line segment does indeed cross the clipping area

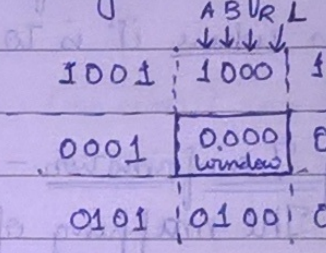
→ Cohen-Sutherland line clipping –

Every line end point in a picture is assigned a four-digit binary code, called a region code.

bit 1 : left bit 3 : below

bit 2 : Right bit 4 = above

(ABRL)



Calculation –

bit 1: $x - x_{wmin}$ bit 3 = $y - y_{wmin}$ If calculation is +ve then

bit 2: $x_{wmax} - x$ bit 4 : $y_{wmax} - y$ bit is 0 Otherwise bit is 1

To find the intersection,

$y = y_1 + m(x - x_1)$ $x = x_{wmin}$ / x_{wmax} where $m = \frac{y_2 - y_1}{x_2 - x_1}$

$x = x_1 + \frac{(y - y_1)}{m}$ $y = y_{wmin}$ / y_{wmax}

→ Liang-Barsky line Clipping –

$x_{wmin} \le x_1 + u\Delta x \le x_{wmax}$

$y_{wmin} \le y_1 + u\Delta y \le y_{wmax}$

Each of these four eq inequalities can be expressed as

$up_k \le q_k$, $k = 1, 2, 3, 4$.

(left) $p_1 = -\Delta x$, $q_1 = x_1 - x_{wmin}$ (bottom) $p_3 = -\Delta y$, $q_3 = y_1 - y_{wmin}$

(right) $p_2 = \Delta x$, $q_2 = x_{wmax} - x_1$ (top) $p_4 = \Delta y$, $q_4 = y_{wmax} - y_1$

Initially, $u_1 = 0$ and $u_2 = 1$.

(1) If $p_k = 0$ then line is parallel to one of the clipping boundaries and if $q_k < 0$ then line is completely outside the boundary.

(2) If $p_k < 0$ then u_1 is updated by $r_k = q_k / p_k$.

(3) If $p_k > 0$ then u_2 is updated by $r_k = q_k / p_k$

(4) If $u_1 > u_2$, we reject the line.

The endpoints of the clipped line are determined from values of u_1 and u_2.

→ Nicholl-Lee-Nicholl line clipping - (NLN)

Determine P_1 position by using three regions:- P_1 in window, P_1 in edge region and P_1 in corner region. Then find P_2 w.r.t P_1. by dividing into four regions using corners of the window.

If $x = x_L$ (x intersection position on the left window boundary) with $u = \frac{x_L - x_1}{x_2 - x_1}$, so that the y intersection position is,

$$y = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x_L - x_1)$$

(T→top) Similarly, $y = y_T$ with $u = \frac{y_T - y_1}{y_2 - y_1} \Rightarrow x = x_1 + \frac{x_2 - x_1}{y_2 - y_1}(y_T - y_1)$

⑤ Polygon Clipping (Area Clipping) -

modifying the line-clipping procedures.

→ Sutherland-Hodgeman Polygon Clipping -

Original polygon → Clip left → Clip right → Clip bottom → Clip top.

Four cases -

(1) If 1st vertex outside & 2nd vertex inside ⇒ intersection & 2nd vertex in output

(2) If both inside ⇒ only 2nd vertex is added to the output vertex list

(3) If 1st vertex inside & 2nd vertex outside ⇒ intersection in output

(3) If both outside ⇒ nothing in output.

→ Weiler - Atherton Polygon Clipping -

It follows the window boundaries. (clockwise or counterclockwise)

For clockwise processing of polygon vertices, we use the following rules -

(1) For an outside-to-inside pair of vertices, follows the polygon boundary.

(2) For an inside-to-outside pair of vertices, follow the window boundary in a clockwise direction.