Matrix representation of transformation

Many graphic applications involves sequence of geometric transformations. For example , animation transformation which is require an object to be translated and rotated at each increment of the motion.

- Transformation can be represented as a product of the row vector [x,y] and a 2x2 matrix accept for the translation.
- Transformations can be combined using matrix multiplication

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

- Matrices are convenient to represent a sequence of transformations

1- Translation matrix T(tx , ty)

We can represent the translation transformation as follows:

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

2- Scaling matrix

- If a point P $\begin{bmatrix} x \\ y \end{bmatrix}$ is being a 2x1 vector. If we multiply it by 2x2 matrix S=$\begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$. We will obtain another 2x1 vector which we can interpret as another point:
  P'= S . P
  $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} . \begin{bmatrix} x \\ y \end{bmatrix}$

- What will happen if we transfer every point by means of multiplication by S and display the result:
  1- If S is the Identity matrix: S=$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$→ No change
  2- If S=$\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$ then $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2x \\ y \end{bmatrix}$
  That mean:
  - every new x coordinate would be twice as large as the old value of vertical lines.
  - x coordinate would be twice as width and the same tall.

  3- If S= $\begin{bmatrix} 0.5 & 0 \\ 0 & 1 \end{bmatrix}$→ shrink all x coordinate (shrink the width with the same tall)

  Examples:

  1- Stretch the image/object to twice and then compress it to one-half of the new width?

  P'= (S_1S_2). P

  $S_1S_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \end{bmatrix}$ → identity matrix then no change

2- Make an image twice as width and twice as tall?

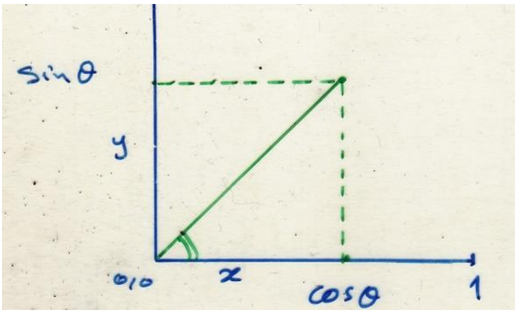
P'= (S_1S_2). P

$$(S_1S_2)\,.\,P=\begin{bmatrix}2 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & 2\end{bmatrix}.\,P \quad --\rightarrow \begin{bmatrix}2 & 0\\ 0 & 2\end{bmatrix}.\begin{bmatrix}x\\ y\end{bmatrix}$$

2- Rotation matrix

If a line segment have two endpoints (0,0) and (x,y), and length $L = \sqrt{x^2 + y^2}$:

- The ratio of the height of the (x,y) endpoint with x-axis have the y coordinates value and the length of the segment will be the *Sin* of the angle: $\mathrm{Sin}(\theta) = \frac{y}{\sqrt{x^2+y^2}}$

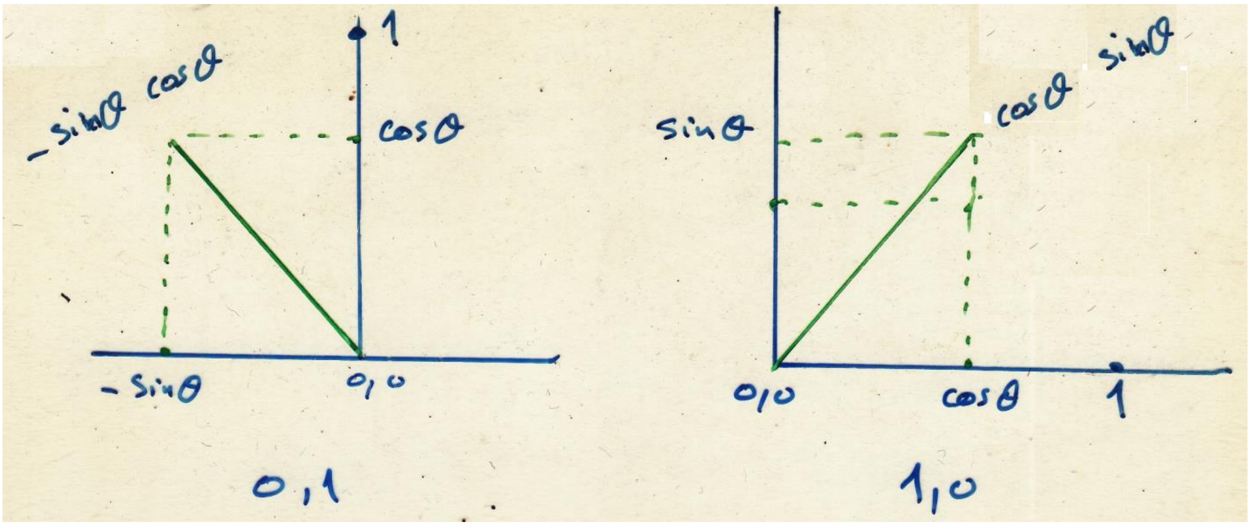


- The ratio of the distance of the (x,y) endpoint with y-axis have the x coordinates value and the length of the segment will be the *Cosine* of the angle: $\mathrm{Cos}(\theta) = \frac{x}{\sqrt{x^2+y^2}}$
- If L=1 then Sin(θ)=y and Cos(θ)=x
- If we rotate the point (1,0) in counterclockwise by an angle θ, it becomes (Cos(θ), Sin(θ)) so:

$$\begin{bmatrix}\cos(\theta)\\ \sin(\theta)\end{bmatrix}=\begin{bmatrix}a & b\\ c & d\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix}=\begin{bmatrix}a\\ c\end{bmatrix}$$

- If we rotate the point (0,1) in counterclockwise by an angle θ, it becomes (-Sin(θ), Cos(θ)) so:

$$\begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} b \\ d \end{bmatrix}$$



From these equations we can see the values of a,b,c,d needed to form the rotation matrix:

$$R = \begin{bmatrix} Cos(\theta) & -Sin(\theta) \\ Sin(\theta) & Cos(\theta) \end{bmatrix}$$

<u>Example:</u>

Rotate the point (2,3) in counterclockwise by an angle $\pi/6$?

The rotation matrix is:

$$R = \begin{bmatrix} Cos\left(\frac{\pi}{6}\right) & -Sin(\pi/6) \\ Sin(\pi/6) & Cos(\pi/6) \end{bmatrix} = \begin{bmatrix} 0.866 & -0.5 \\ 0.5 & 0.866 \end{bmatrix}$$

Then the rotated point would be :

P' = R . P

$$P' = \begin{bmatrix} 0.866 & -0.5 \\ 0.5 & 0.866 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 3.232 \\ 3.598 \end{bmatrix}$$

<u>Notes:</u>

- We can rotate an entire line segment by rotating both endpoints which specify it.

- The rotation matrix for an angle θ in clockwise would be:

$$\begin{bmatrix} Cos(-\theta) & -Sin(-\theta) \\ Sin(-\theta) & Cos(-\theta) \end{bmatrix} = \begin{bmatrix} Cos(\theta) & Sin(\theta) \\ -Sin(\theta) & Cos(\theta) \end{bmatrix}$$

H.W-1\: Rotate the triangle (5,2),(1,1),(0,0) by rotation in counterclockwise with angle 45° about origin?

H.W-2

a- Find the matrix that represent rotation of an object by 30° about the origin?

b- what are the new coordinates of the point p(2,-4) after the rotation?

Homogenous coordinate in transformation matrix

- Why Homogeneous Coordinates?
    1. Mathematicians commonly use homogeneous coordinates as they allow scaling factors to be removed from equations.
    2. Using homogeneous coordinates allows us use matrix multiplication to calculate transformations – extremely efficient!
    3. Homogeneous coordinates seem unintuitive, but they make graphics operations much easier.
    4. Since a 2x2 matrix representation of translation does not exist!!. So by using a homogenous coordinate system then we can represent *2x2 translation transformation* as a matrix multiplication.
    5. It provides a consistent, uniform way of handling *affine transformations*. 2D affine transformations always have a bottom row of **[0 0 1].**

       An “affine point” is a “linear point” with an added w-coordinate which is always 1:

$$\mathrm{p_{aff}} = \begin{bmatrix} \mathrm{p_{lin}} \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Applying an affine transformation gives another affine point.

- A point (x, y) can be re-written in homogeneous coordinates as (xw, yw,w)
  - The homogeneous parameter w is a non-zero value such that x and y coordinates can easily be recovered by dividing the first and second numbers by the third.

$$x = \frac{xw}{w} \qquad\qquad y = \frac{yw}{w}$$

  - We can then write any point (x, y) as :

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} w.x \\ w.y \\ w \end{bmatrix}, w \neq 0$$

  - We can conveniently choose w = 1 so that (x, y) becomes:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- In homogeneous coordinates the **scaling matrix** as follows:

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

$\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} xw \\ yw \\ w \end{bmatrix} = \begin{bmatrix} S_x xw \\ S_y yw \\ w \end{bmatrix}$ → divide by w then $\begin{bmatrix} S_x x \\ S_y y \end{bmatrix}$ is the correctly scaled point.

- The counterclockwise **rotation matrix** is

$$\begin{bmatrix} Cos(\theta) & -Sin(\theta) \\ Sin(\theta) & Cos(\theta) \end{bmatrix}$$

Using homogeneous coordinates we get: $\begin{bmatrix} Cos(\theta) & -Sin(\theta) & 0 \\ Sin(\theta) & Cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Applying it to the point (x,y) with homogeneous (xw,yw,w) gives:

$$R.P = \begin{bmatrix} Cos(\theta) & -Sin(\theta) & 0 \\ Sin(\theta) & Cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} xw \\ yw \\ w \end{bmatrix}$$

- The homogeneous coordinate **translation matrix** of t_x , t_y is

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

$$T.P = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} xw \\ yw \\ w \end{bmatrix} = \begin{bmatrix} (xw + t_x w) \\ (yw + t_y w) \\ w \end{bmatrix} \rightarrow$$

divide by w then we get the point P= $\begin{bmatrix} x + t_x \\ y + t_y \end{bmatrix}$ is the correctly translated point.

In general:

1- Translation

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

2- Rotation

$$\begin{bmatrix} \bar{x} \\ \bar{y} \\ 1 \end{bmatrix} = \begin{bmatrix} Cos(\theta) & -Sin(\theta) & 0 \\ Sin(\theta) & Cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

3- Scaling

$$\begin{bmatrix}\bar{x}\\ \bar{y}\\ 1\end{bmatrix} = \begin{bmatrix}S_x & 0 & 0\\ 0 & S_y & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ 1\end{bmatrix}$$

- **Rotation about an arbitrary point**

- The homogeneous coordinate transformation matrix for counterclockwise rotation about point $(x_c\ ,\ y_c)$ is done by three steps as follows:

1- Translation to the origin

$$T_1=\begin{bmatrix}1 & 0 & -x_c\\ 0 & 1 & -y_c\\ 0 & 0 & 1\end{bmatrix},$$

$$\begin{bmatrix}\bar{x}\\ \bar{y}\\ 1\end{bmatrix} = \begin{bmatrix}1 & 0 & -x_c\\ 0 & 1 & -y_c\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ 1\end{bmatrix}$$

2- Rotation about the origin

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

$$\begin{bmatrix}\bar{\bar{x}}\\ \bar{\bar{y}}\\ 1\end{bmatrix} = \begin{bmatrix}Cos(\theta) & -Sin(\theta) & 0\\ Sin(\theta) & Cos(\theta) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\bar{x}\\ \bar{y}\\ 1\end{bmatrix}$$

3- Translation back to its correct position

$$T_2=\begin{bmatrix}1 & 0 & x_c\\ 0 & 1 & y_c\\ 0 & 0 & 1\end{bmatrix}$$

$$\begin{bmatrix}\bar{\bar{\bar{x}}}\\ \bar{\bar{\bar{y}}}\\ 1\end{bmatrix} = \begin{bmatrix}1 & 0 & x_c\\ 0 & 1 & y_c\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\bar{\bar{x}}\\ \bar{\bar{y}}\\ 1\end{bmatrix}$$

Composite Transformations

- Matrices are a convenient and efficient way to represent a sequence of transformations.
- Matrix multiplication is not commutative so that the order of transformations is important.
- What if we want to rotate and translate?

To rotate a point $(T_2\ (\mathrm{R}\ (T_1 \begin{bmatrix} xw \\ yw \\ w \end{bmatrix}))\ )$

- Now we must form an overall transformation matrix as follows:-

$$(T_2(\mathrm{R}\ T_1)) \begin{bmatrix} xw \\ yw \\ w \end{bmatrix}$$

$$T_2\mathrm{R}\ T_1 = \begin{bmatrix} 1 & 0 & x_c \\ 0 & 1 & y_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Cos(\theta) & -Sin(\theta) & 0 \\ Sin(\theta) & Cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & x_c \\ 0 & 1 & y_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Cos(\theta) & -Sin(\theta) & -x_c\,\mathrm{Cos}(\theta) + y_c Sin(\theta) \\ Sin(\theta) & Cos(\theta) & -x_c\,\mathrm{sin}(\theta) - y_c Cos(\theta) \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} Cos(\theta) & -Sin(\theta) & -x_c Cos(\theta) + y_c Sin(\theta) + x_c \\ Sin(\theta) & Cos(\theta) & -x_c Sin(\theta) - y_c Cos(\theta) + y_c \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} Cos(\theta) & -Sin(\theta) & x_c(1 - Cos(\theta)) + y_c Sin(\theta) \\ Sin(\theta) & Cos(\theta) & y_c(1 - Cos(\theta)) - x_c Sin(\theta) \\ 0 & 0 & 1 \end{bmatrix}$$

Example: Rotate line segment by 45 degrees about endpoint **a** and note that tx=3?

$$\begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(45) & -\sin(45) & 0 \\ \sin(45) & \cos(45) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ 1 \end{bmatrix} = \begin{bmatrix} a'_x \\ a'_y \\ 1 \end{bmatrix}$$

H.W Rotate the triangle (5,2),(1,1),(0,0) by rotation in counterclockwise with angle 45° about fixed point (-1,-1)?

Window to Viewport Mapping

Window:

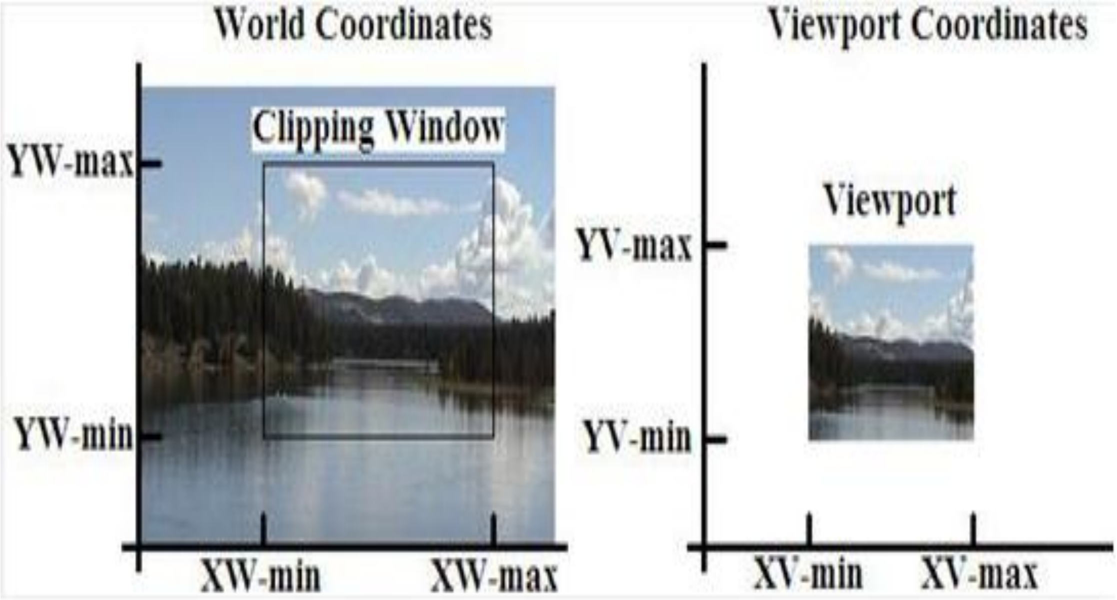
1. A world-coordinate area selected for display is called a window.
2. In computer graphics, a window is a graphical control element.
3. It consists of a visual area containing some of the graphical user interface of the program it belongs to and is framed by a window decoration.
4. A window defines a rectangular area in world coordinates. You can define the window to be larger than, the same size as, or smaller than the actual range of data values, depending on whether you want to show all of the data or only part of the data.

Viewport:

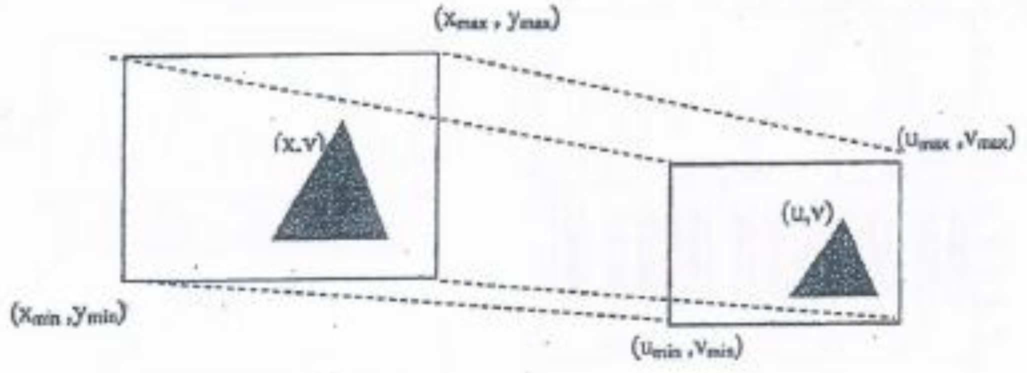
1. An area on a display device to which a window is mapped is called a viewport.
2. A viewport is a polygon viewing region in computer graphics. The viewport is an area expressed in rendering-device-specific coordinates, e.g. pixels for screen coordinates, in which the objects of interest are going to be rendered.
3. A viewport defines in **normalized coordinates** a rectangular area on the display device where the image of the data appears. You can have your graph take up the entire display device or show it in only a portion, say the upper-right part.

Window to viewport transformation

1. Window-to-Viewport transformation is the process of transforming a two-dimensional, world-coordinate scene to device coordinates.
2. In particular, objects inside the world or clipping window are mapped to the viewport. The viewport is displayed in the interface window on the screen.
3. In other words, the clipping window is used to select the part of the scene that is to be displayed. The viewport then positions the scene on the output device.
4. **Example:**



This transformation involves developing formulas that start with a point in the world window, say (x, y).



$$(x,y) \longrightarrow (u,v)$$

Figure (4.2)

$$\frac{x-x_{\min}}{x_{\max}-x_{\min}}=\frac{u-u_{\min}}{u_{\max}-u_{\min}}$$

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

By rewriting this relationship we get the following formula:

$$u = c1\,x + c2 \qquad c_1=\frac{u_{\max}-u_{\min}}{x_{\max}-x_{\min}}$$

$$c_2=u_{\min}-c_1x_{\min}$$

$$v = d1\,y + d2 \qquad d_1=\frac{v_{\max}-v_{\min}}{y_{\max}-y_{\min}}$$

$$d_2=v_{\min}-d_1y_{\min}$$

Example1:

A normalized window has left and right boundaries of (-0.05 to +0.05) and lower and upper boundaries of (0.1 to 0.2). the viewport window left and right is (250,550) and lower to upper is (100,400),find the coordinate of any point (u,v) in the viewport window.

Solution

Window(xmin=-0.05 , xmax=+0.05 , ymin=0.1, ymax=0.2)

Viewport (umin=250, umax=550, vmin=100, vmax=400)

u=$c1x + c2$

$$c1 = \frac{umax - umin}{xmax - xmin}$$

c1=$\frac{(550-250)}{0.05-(-0.05)}$=300/0.1=3000

c2=umin-c1xmin

=250-3000(-0.05) =250+150 =400

u=3000x+400

v=d1y+d2

$$d1 = \frac{vmax - vmin}{ymax - ymin}$$

$d1 = \frac{(400-100)}{(0.2-0.1}$ =300/0.1=3000

d2=vmin-d1ymin

=100-3000(0.1)

=-200

v=3000y-200

Example2:

A normalized window has left(xmin=10) and right(xmax=50) boundaries and lower(ymin=5) and upper(ymax=30) boundaries .the viewport window left(umin=25) and right(umax=75) and lower(vmin=25) to upper (vmax=75) find the coordinate of any point (u,v) in the viewport window.

u=$c1x + c2$

$$c1 = \frac{umax - umin}{xmax - xmin}$$

c1 =75-25/50-10

c1=50/40=1.25

c2=umin-c1xmin

=25-1.25*10

= 25-12.5=12.5

u=1.25x+12.5

v=d1y+d2

$$d1 = \frac{vmax - vmin}{ymax - ymin}$$

=75-25/30-5

d1=50/25=2

d2= vmin-d1ymin

=25-2*5

=25-10=15

v=2y+15

Window to Viewport Transformation N

We can express these two formula for computing (u,v) from (x,y) by term:

(translate-scale-translate)

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} . N$$

$N=T_2ST_1$

1. T_1 is the translation matrix about window origin :

$$T1 = \begin{bmatrix} 1 & 0 & -xmin \\ 0 & 1 & -ymin \\ 0 & 0 & 1 \end{bmatrix}$$

2. S is the scaling transformation matrix:

$$S = \begin{bmatrix} \frac{umax - umin}{xmax - xmin} & 0 & 0 \\ 0 & \frac{vmax - vmin}{ymax - ymin} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3. T2 is the translation matrix position of the viewport :

$$T2 = \begin{bmatrix} 1 & 0 & umin \\ 0 & 1 & vmin \\ 0 & 0 & 1 \end{bmatrix}$$

Example1:

A normalized window has left and right boundaries of (-0.05 to +0.05) and lower and upper boundaries of (0.1 to 0.2). the viewport window left and right is (250,550) and lower to upper is (100,400),find the transformation N.

Solution **N=T$_2$ST$_1$**

$$\mathrm{T1} = \begin{bmatrix} 1 & 0 & -xmin \\ 0 & 1 & -ymin \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{T1} = \begin{bmatrix} 1 & 0 & -(-0.05) \\ 0 & 1 & -0.1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{S} = \begin{bmatrix} \frac{umax-umin}{xmax-xmin} & 0 & 0 \\ 0 & \frac{vmax-vmin}{ymax-ymin} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{S} = \begin{bmatrix} 3000 & 0 & 0 \\ 0 & 3000 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{T2} = \begin{bmatrix} 1 & 0 & umin \\ 0 & 1 & vmin \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{T2} = \begin{bmatrix} 1 & 0 & 250 \\ 0 & 1 & 100 \\ 0 & 0 & 1 \end{bmatrix}$$

$$N = \begin{bmatrix} 1 & 0 & 250 \\ 0 & 1 & 100 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3000 & 0 & 0 \\ 0 & 3000 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -(-0.05) \\ 0 & 1 & -0.1 \\ 0 & 0 & 1 \end{bmatrix}$$

Example 2

Specify individually the translation and scaling matrices required to transform a 2D window of [Xmin=-234, Ymin=156] and [Xmax=66, Ymax=456] to a display viewport of [Umin=45, Vmin=35] and [Umax=245, Vmax=185].

Solution $N=T_2ST_1$

$$T1 = \begin{bmatrix} 1 & 0 & -xmin \\ 0 & 1 & -ymin \\ 0 & 0 & 1 \end{bmatrix}$$

$$T1 = \begin{bmatrix} 1 & 0 & -(-234) \\ 0 & 1 & -156 \\ 0 & 0 & 1 \end{bmatrix}$$

$$S = \begin{bmatrix} \frac{umax-umin}{xmax-xmin} & 0 & 0 \\ 0 & \frac{vmax-vmin}{ymax-ymin} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$S = \begin{bmatrix} \frac{245-45}{66+234} & 0 & 0 \\ 0 & \frac{185-35}{456-156} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$S = \begin{bmatrix} 0.6 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{T2} = \begin{bmatrix} 1 & 0 & umin \\ 0 & 1 & vmin \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{T2} = \begin{bmatrix} 1 & 0 & 45 \\ 0 & 1 & 35 \\ 0 & 0 & 1 \end{bmatrix}$$

$$N = \begin{bmatrix} 1 & 0 & 250 \\ 0 & 1 & 100 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3000 & 0 & 0 \\ 0 & 3000 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -(-0.05) \\ 0 & 1 & -0.1 \\ 0 & 0 & 1 \end{bmatrix}$$