Window /View port

Windows are areas on screen where graphical information can be displayed

View ports refer to rectangular areas inside window that display graphical data.

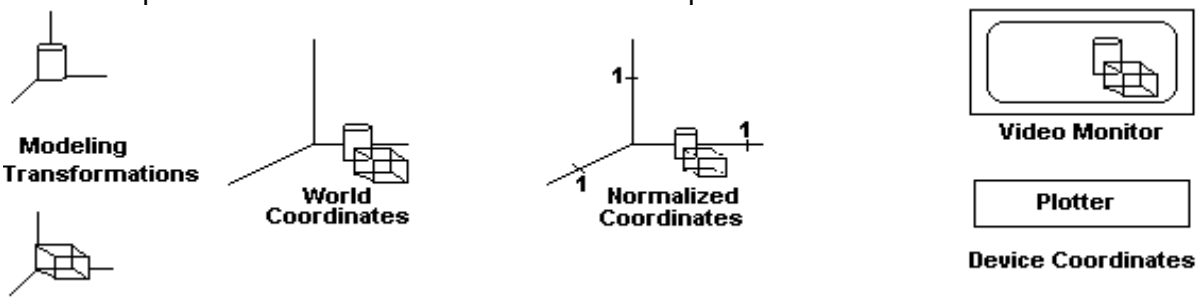
View ports are or can be of different sizes but are always smaller than size if window.

For practical applications we need a transformation to translate and scale window to any size by moving it to specified rectangular area on screen

This area is commonly called the **view port**

The choice of window decides what we want to see on display and choice of view port decides where we want to see on display screen

This concept allows user to divide screen into virtual partitions.



The display hardware divides screen into a number of pixels arranged in a grid with each pixel are associated its x , y coordinates.

Top left corner of screen is called origin of coordinate system

Value of ‘x’ coordinate increases from left to right

Value of ‘y’ coordinate increases from top towards bottom.

These coordinates are reference d to as device for screen coordinates.

For the VGA graphics card the size of display grid is 640 x 480.

Similarly suitable coordinate system can be selected for objective in space

The coordinates so defined are world coordinates

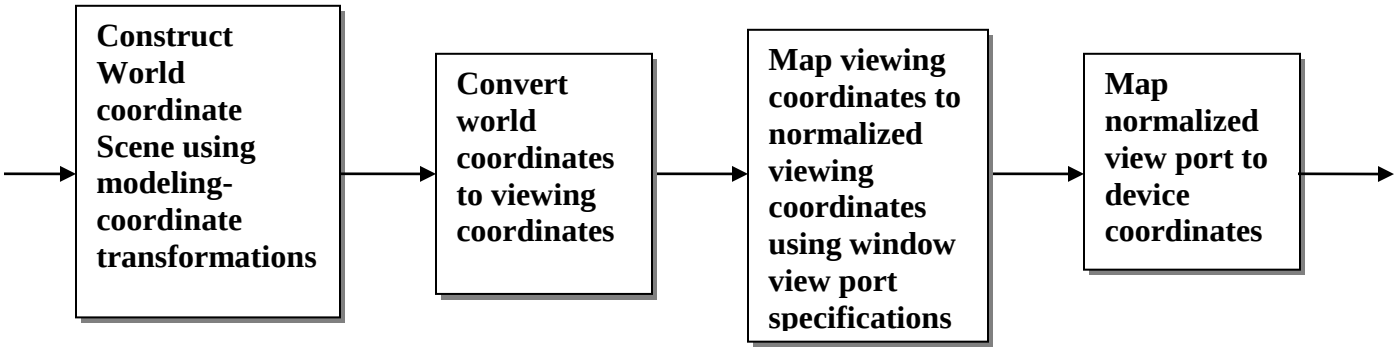
It may sometimes be desirable to select a part of object or drawing for display in order to obtain sufficient detail of the object on display

The part of interest is enclosed n a rectangular boundary called a window

For displaying one has to convert world coordinates into screen coordinates. This transformation is called **viewing transformation**

In general view transformation consists of operations such as scaling, translation , rotation etc.

Window to View Port Transformation (2-D Viewing Transformation Pipeline)



A point in position(x_w,y_w) in window is mapped into position (x_v, y_v) in the view port

To maintain same relative placement in view port as in window we require,

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

and,

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

solving equations for view port position (x_v , y_v)

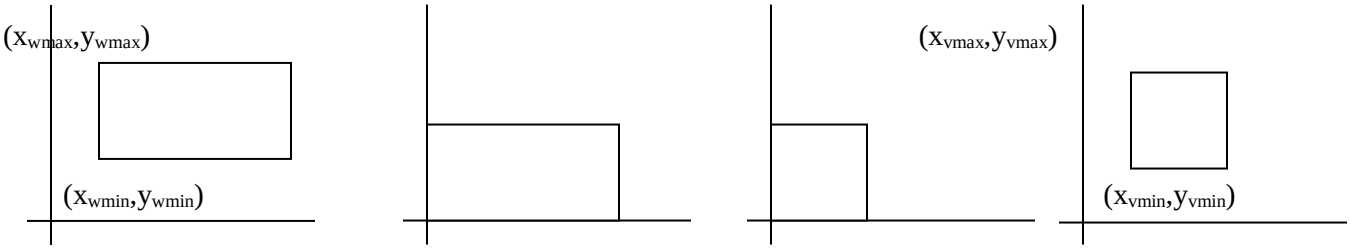
$$x_v = x_{vmin} + (x_w - x_{wmin}) \frac{(x_{vmax} - x_{vmin})}{(x_{wmax} - x_{wmin})} = x_{vmin} + (x_w - x_{wmin}) . s_x$$

$$y_v = y_{vmin} + (y_w - y_{wmin}) \frac{(y_{vmax} - y_{vmin})}{(y_{wmax} - y_{wmin})} = y_{vmin} + (y_w - y_{wmin}) . s_y$$

where, s_x and s_y are the scaling factors.

Sequence of Transformations

i. Perform scaling transformation using fixed point position(x_{wmin} , y_{wmin}) that scales the window area to the size of the view port
ii. Translate the scaled window area to the position of the view port



Window
in World Coordinate System

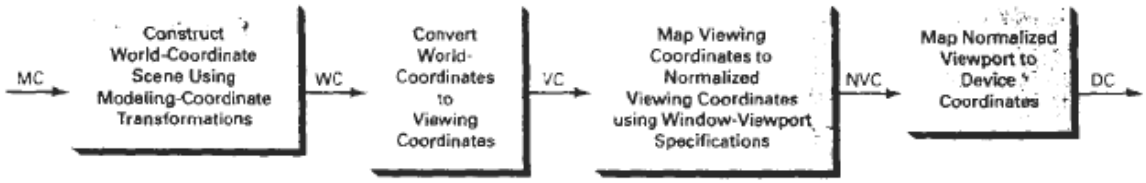
Viewport
in Device Coordinate System

THE VIEWING PIPELINE: 2D

A world-coordinate area selected for display is called a window, defines ***what*** is to be viewed. An area on a display device to which a window is mapped is called a viewport defines ***where*** it is to be displayed.

Viewing Transformation: The mapping of a part of a world-coordinate scene to device coordinates

Sometimes the two-dimensional viewing transformation is simply referred to as the ***window-to-viewport transformation*** or the ***windowing transformation.*** BUT, in general, viewing involves more than just the transformation from the window to the viewport.



First, we construct the scene in World Coordinates from objects modeled in their individual coordinate systems called **Modeling Coordinate System**

Next. to obtain a particular orientation for the window, we can set up a two-dimensional **Viewing-Coordinate System** in the world-coordinate plane, and define a window

In the viewing-coordinate system, the viewing coordinate reference frame is used to provide a method for setting up arbitrary orientations for rectangular windows. Once the viewing reference frame is established, descriptions from world coordinates are transferred to viewing coordinates. So **clipping** and **2D transformations** take place in bringing objects from World Coordinate System to Viewing Coordinate System.

Then a viewport in normalized coordinates (in the range from **0** to 1) is defined and the viewing-coordinate description of the scene are mapped to **normalized coordinates**.

At the final step, the parts of the picture that are outside the viewport are clipped, and the contents of the viewport are transferred to **device coordinates**.

By changing the position of the viewport, we can view objects at different positions on the display area of an output device. Also, by varying the size of viewports, we can change the size and proportions of displayed objects. We achieve zooming effects by successively mapping different-sized windows on a fixed-size viewport.

3D Viewing Transformation

At the first step, a scene is constructed by transforming object descriptions from **modeling coordinates** to **world coordinates**.

Next, a view mapping convert: the world descriptions to **viewing coordinates**.

At the projection stage, the viewing coordinates are transformed to **projection coordinates**, which effectively converts the view volume into a rectangular parallelepiped.

Then, the parallelepiped is mapped into the unit cube, a normalized view volume called the **normalized projection coordinate system**.

The mapping to normalized projection coordinates is accomplished by transforming points within the rectangular parallelepiped into a position within a specified three-dimensional viewport, which occupies part or all of the unit **cube.**

Finally, at the workstation stage, normalized projection coordinates are converted to **device coordinates** for display. The normalized view volume is a region defined by the planes

$$x = 0, \quad x = 1, \quad y = 0, \quad y = 1, \quad z = 0, \quad z = 1$$

Mapping positions within a rectangular view volume to a three-dimensional rectangular viewport is accomplished with a combination of scaling and translation, similar to the operations needed for a two-dimensional window-to viewport mapping. We can express the three-dimensional transformation matrix for these operations in the form

$$\begin{bmatrix} D_x & 0 & 0 & K_x \\ 0 & D_y & 0 & K_y \\ 0 & 0 & D_z & K_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We can express the three-dimensional transformation matrix for these operations in the form

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

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

$$D_z = \frac{zv_{max} - zv_{min}}{z_{back} - z_{front}}$$

Factors D_x D_y , and D_z are the ratios of the dimensions of the viewport and regular parallelepiped view volume in the *x,* y, and z directions where the view-volume boundaries are established by the window limits ***(xw$_{min}$, xw$_{max}$, , yw$_{min}$, yw$_{max}$)*** and the positions z_{front} and z_{back} of the front and back planes.

Viewport boundaries are set with the coordinate values ***xv$_{min}$, xv$_{max}$, , yv$_{min}$, yv$_{max}$,*** zv_{min} and ***zv$_{max}$***. The additive translation factors $\mathbf{K_x}$, $\mathbf{K_y}$, and $\mathbf{K_z}$ in the transformation are

$$K_x = xv_{min} - xw_{min} D_x$$

$$K_y = yv_{min} - yw_{min} D_y$$

$$K_z = zv_{min} - z_{front} D_z$$