

Viewing Transformations and the 2-D Pipeline

- 2-D Transformations, brief review
- Canonical 2-D Coordinate Systems
 - Local Coordinates
 - World Coordinates
 - Viewing Coordinates
 - Normalized Device Coordinates
 - Device Coordinates
- Clipping Algorithms
- Interactive Input Devices
- New Homework Assignment

Two-dimensional transformations

- Two ways to apply these operators:
 - Object transformations
 - Coordinate transformations
- Basic Geometric operations:
 - Translation
 - Rotation
 - Scaling
- Other operations:
 - Reflection
 - Shear

Use of Homogeneous Coordinates

Allows our transformations to be represented as matrix multiplications.

- Given an $h \in \mathbb{R}$, the homogeneous point (x_h, y_h, h) of the Cartesian point (x, y) is obtained from the map

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x_h \\ y_h \\ h \end{pmatrix} = \begin{pmatrix} h \cdot x \\ h \cdot y \\ h \end{pmatrix}$$

- Converting back (assuming $h \neq 0$):

$$\begin{pmatrix} x_h \\ y_h \\ h \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_h/h \\ y_h/h \end{pmatrix}$$

- Two homogeneous points are *equivalent* if they are proportional to each other (by a nonzero factor), e.g.

$$\begin{pmatrix} 3 \\ -5 \\ 1 \end{pmatrix} \equiv \begin{pmatrix} -6 \\ 10 \\ -2 \end{pmatrix} \equiv \begin{pmatrix} 3/5 \\ -1 \\ 1/5 \end{pmatrix}$$

all represent the Cartesian point $(3, -5)$.

Examples of Translation

- Translate (by t_x to the right and t_y vertically) an *object* with vertices (x_i, y_i) , $i = 1, \dots, n$.

$$\begin{pmatrix} x'_{hi} \\ y'_{hi} \\ h' \end{pmatrix} = T(t_x, t_y) \begin{pmatrix} x_{hi} \\ y_{hi} \\ h \end{pmatrix} \quad \text{where, } T(t_x, t_y) = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

(The primed coordinates denote the vertices of the translated object in the original coordinate system.)

- Translate (by t_x to the right and t_y vertically) a *coordinate system* that contains an object with vertices (x_i, y_i) , $i = 1, \dots, n$.

$$\begin{pmatrix} x'_{hi} \\ y'_{hi} \\ h' \end{pmatrix} = T(-t_x, -t_y) \begin{pmatrix} x_{hi} \\ y_{hi} \\ h \end{pmatrix}$$

(The primed coordinates describe the object's vertices in the new coordinate system.)

Examples of Rotation

- Rotate (by angle θ , counter-clockwise) an *object* with vertices (x_i, y_i) , $i = 1, \dots, n$.

$$\begin{pmatrix} x'_{hi} \\ y'_{hi} \\ h' \end{pmatrix} = R(\theta) \begin{pmatrix} x_{hi} \\ y_{hi} \\ h \end{pmatrix} \quad \text{where, } R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(The primed coordinates denote the vertices of the rotated object in the original coordinate system.)

- Rotate (by angle θ , counter-clockwise) a *coordinate system* that contains an object with vertices (x_i, y_i) , $i = 1, \dots, n$.

$$\begin{pmatrix} x'_{hi} \\ y'_{hi} \\ h' \end{pmatrix} = R(-\theta) \begin{pmatrix} x_{hi} \\ y_{hi} \\ h \end{pmatrix}$$

(The primed coordinates describe the object's vertices in the new coordinate system.)

Examples of Scaling

- Magnify (by factors s_x horizontally, and s_y vertically) an *object* with vertices (x_i, y_i) , $i = 1, \dots, n$.

$$\begin{pmatrix} x'_{hi} \\ y'_{hi} \\ h' \end{pmatrix} = S(s_x, s_y) \begin{pmatrix} x_{hi} \\ y_{hi} \\ h \end{pmatrix} \quad \text{where, } S(s_x, s_y) = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

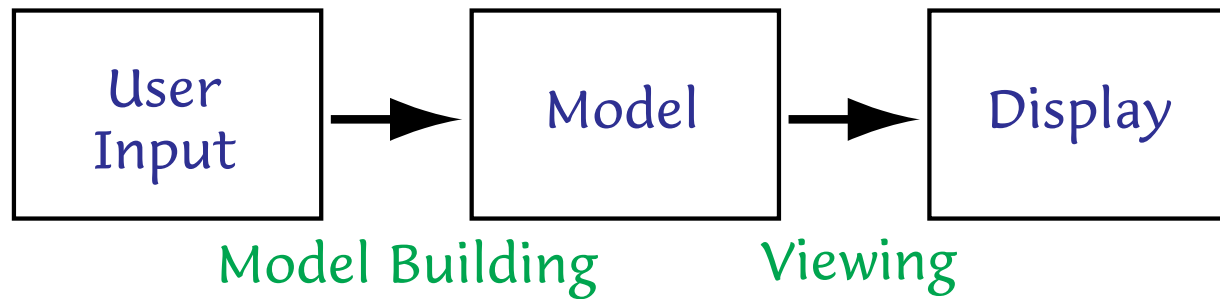
(The primed coordinates denote the vertices of the scaled object in the original coordinate system.)

- Magnify (by factors s_x horizontally, and s_y vertically) a *coordinate system* that contains an object with vertices (x_i, y_i) , $i = 1, \dots, n$.

$$\begin{pmatrix} x'_{hi} \\ y'_{hi} \\ h' \end{pmatrix} = S\left(\frac{1}{s_x}, \frac{1}{s_y}\right) \begin{pmatrix} x_{hi} \\ y_{hi} \\ h \end{pmatrix}$$

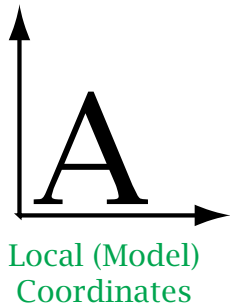
(The primed coordinates describe the object's vertices in the new coordinate system.)

Structure of Graphics Applications



The Viewing Pipeline (2-D)

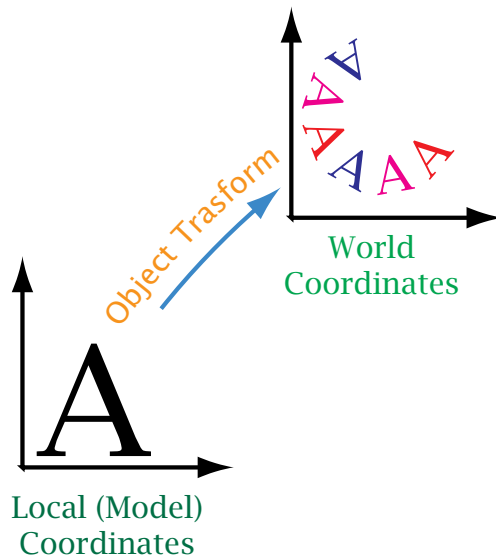
Local C.



Each model component can use its own **local** coordinate system.

The Viewing Pipeline (2-D) cont.

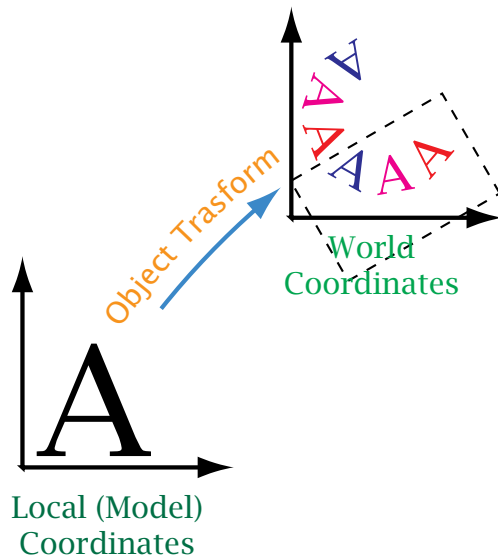
Local C. \rightarrow World C.



The **world** coordinate system describes the relative positions and orientations of every generated object.

The Viewing Pipeline (2-D) cont.

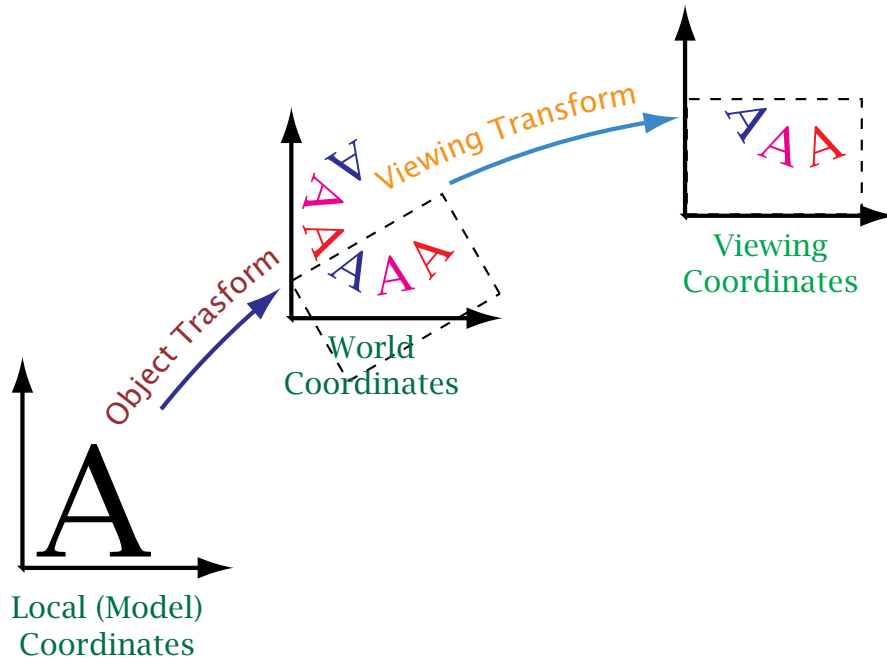
Local C. \rightarrow World C.



A **window** in the world coordinates describes a rectangular region that will be mapped to a device.

The Viewing Pipeline (2-D) cont.

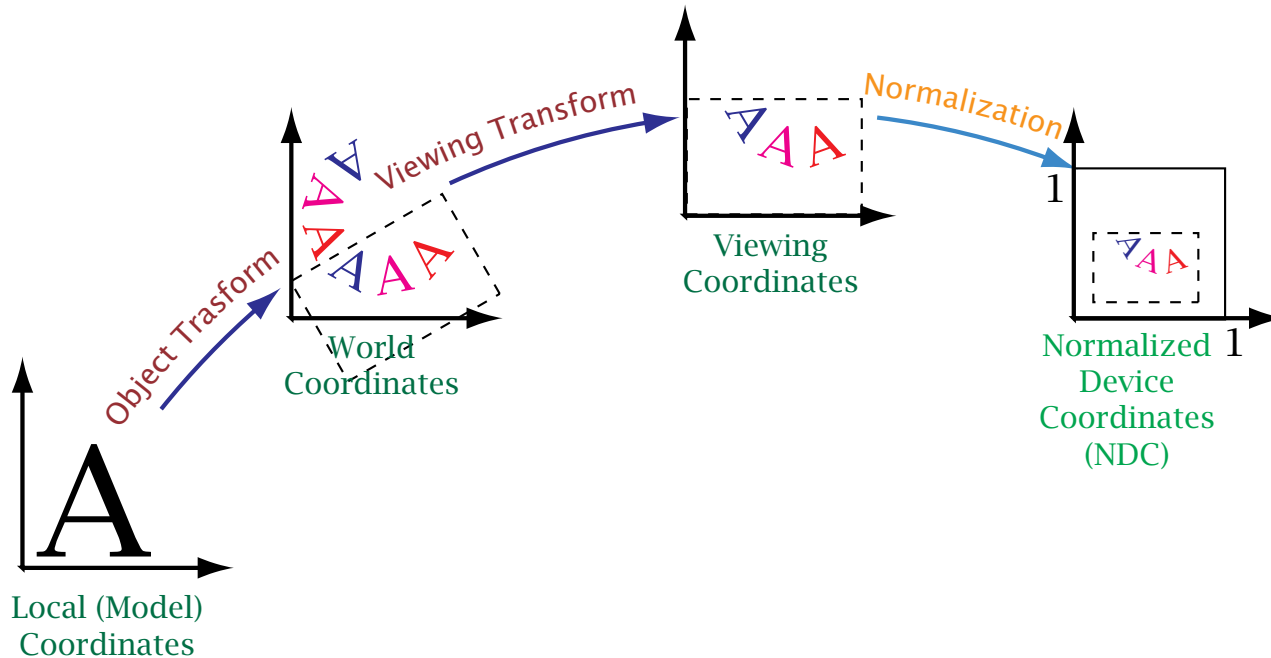
Local C. \rightarrow World C. \rightarrow Viewing C.



Viewing Coordinates describe the position and orientation of each object with respect to a given window. Objects outside of the window are **clipped**.

The Viewing Pipeline (2-D) cont.

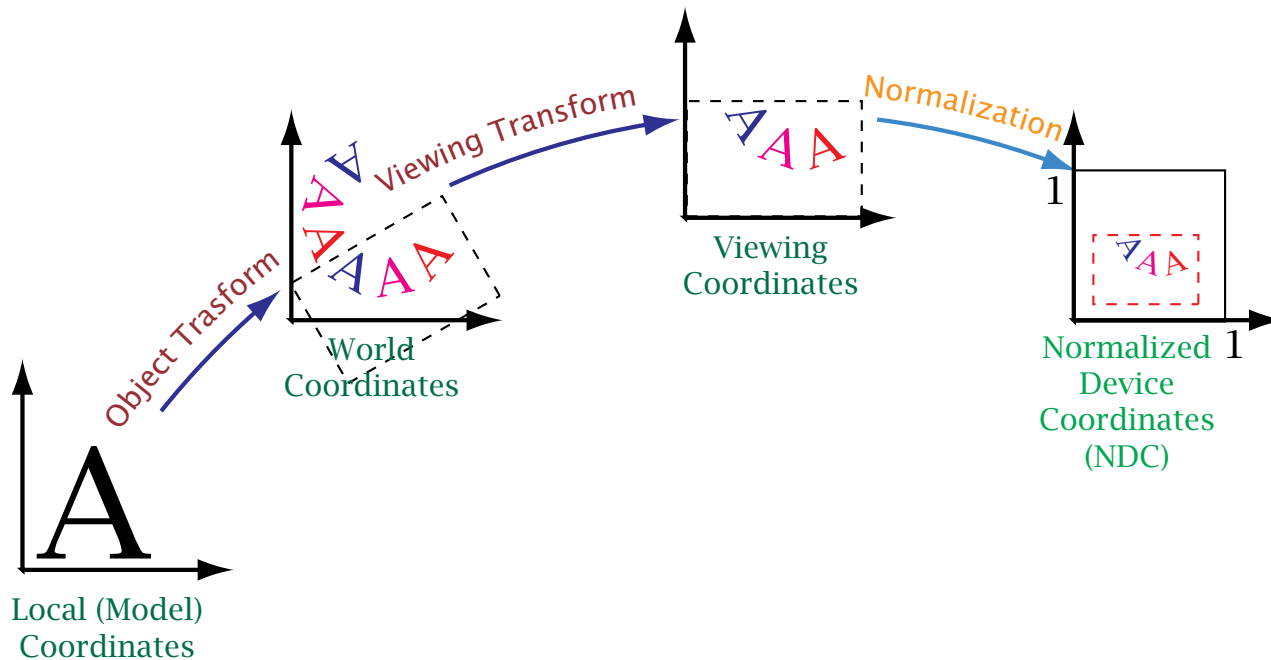
Local C. \rightarrow World C. \rightarrow Viewing C. \rightarrow **N.D.C.**



The **NDC** describe a virtual display to facilitate device compatibility.

The Viewing Pipeline (2-D) cont.

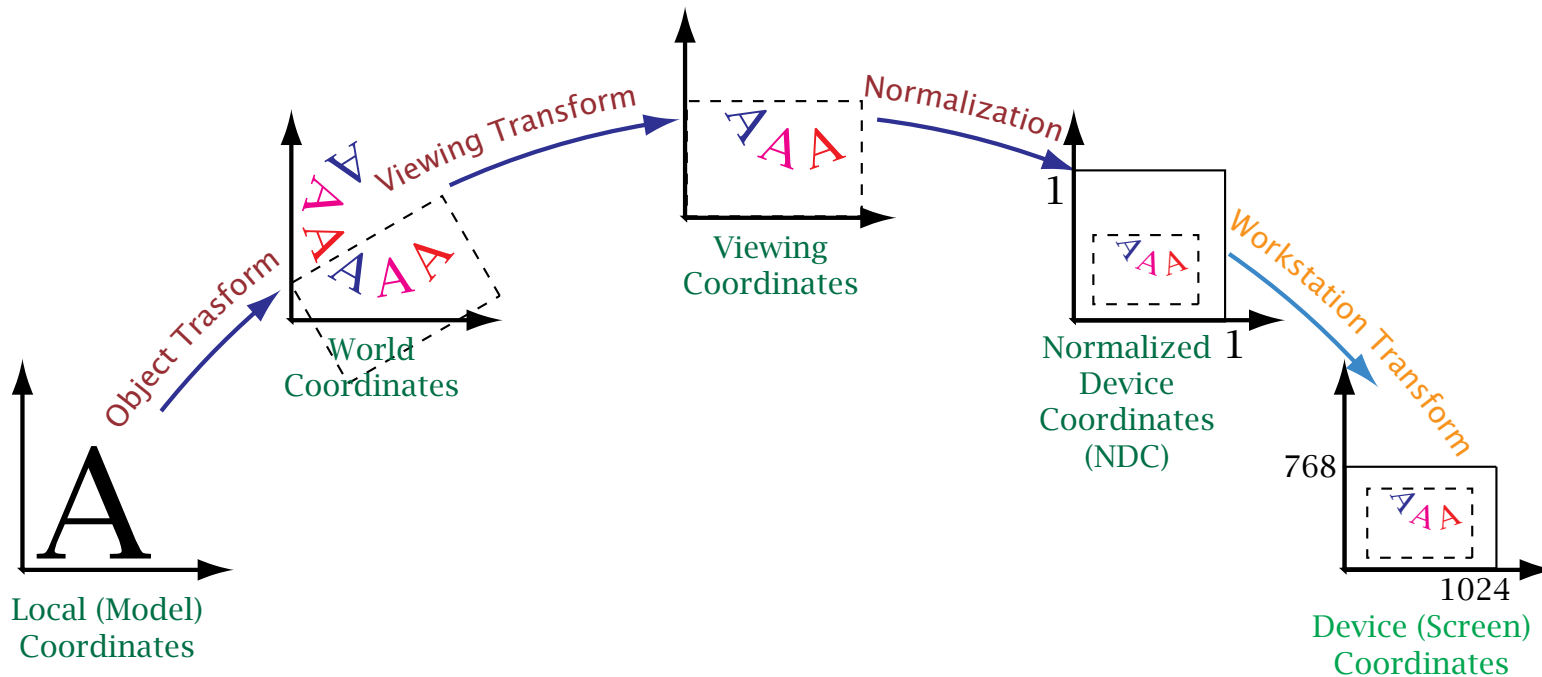
Local C. \rightarrow World C. \rightarrow Viewing C. \rightarrow N.D.C.



The rectangular region in the NDC onto which the window is mapped is called the **viewport**.

The Viewing Pipeline (2-D) cont.

Local C. \rightarrow World C. \rightarrow Viewing C. \rightarrow N.D.C. \rightarrow Device C.



The **device coordinates** reflect the aspect and resolution of a graphics display, printer, or plotter. Scanline conversion is performed if the device is discrete.

Local Coordinates \longrightarrow World Coordinates

Ingredients:

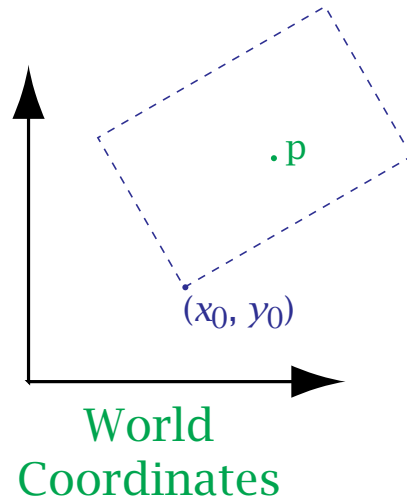
- An assortment of graphical primitives and model components.
- *Object* transformations:

$$T(t_x, t_y), R(\theta), S(s_x, s_y), \text{ etc.}$$

Use the object transformations to map the primitives and model components from local coordinates to world coordinates.

World Coordinates \rightarrow Viewing Coordinates

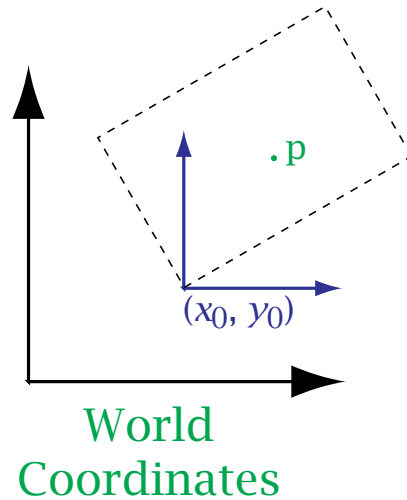
Let \mathbf{p} denote a vertex in the world coordinate system.



\mathbf{p}

World Coordinates \rightarrow Viewing Coordinates

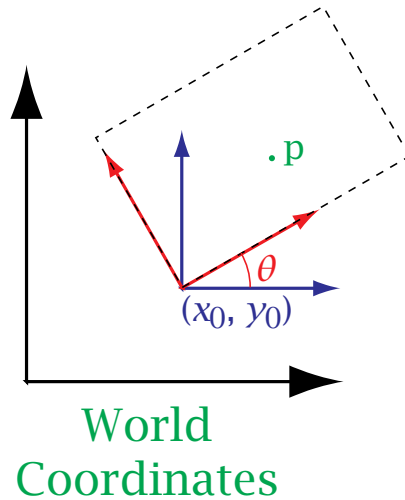
Let \mathbf{p} denote a vertex in the world coordinate system.



$$T(-x_0, -y_0)\mathbf{p}$$

World Coordinates \rightarrow Viewing Coordinates

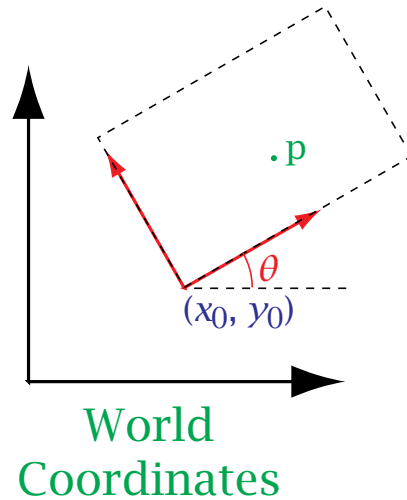
Let \mathbf{p} denote a vertex in the world coordinate system.



$$R(-\theta)T(-x_0, -y_0)\mathbf{p}$$

World Coordinates \rightarrow Viewing Coordinates

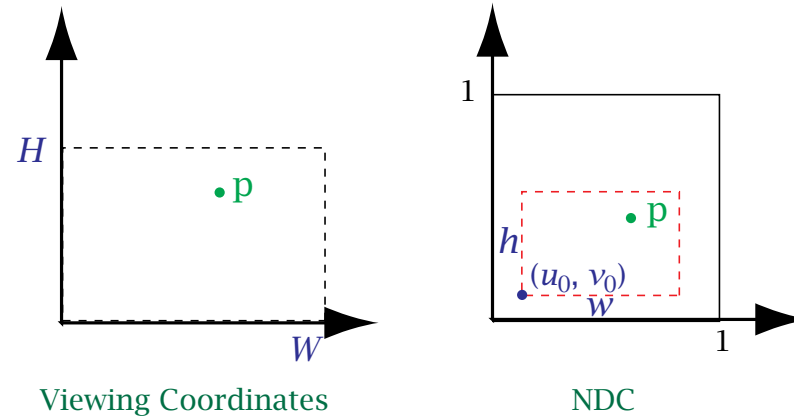
Let \mathbf{p} denote a vertex in the world coordinate system.



$$M_{WC,VC} = R(-\theta)T(-x_0, -y_0)$$

Viewing Coordinates \rightarrow NDC Coordinates

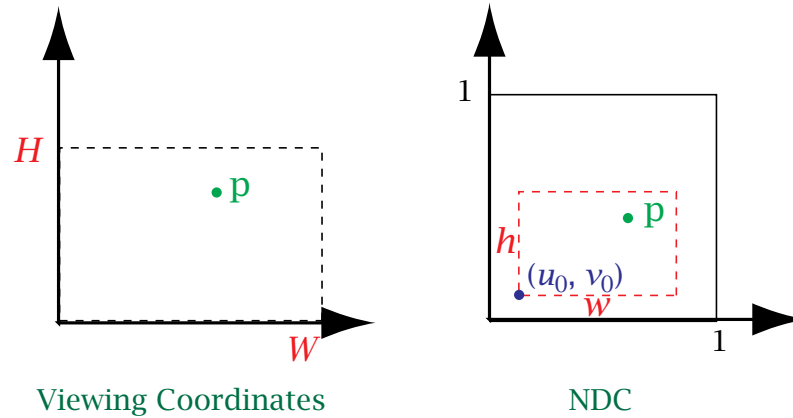
Let \mathbf{p} denote a vertex in the viewing coordinate system. How does one map \mathbf{p} to the **viewport** in the normalized device coordinate system?



\mathbf{p}

Viewing Coordinates \rightarrow NDC Coordinates

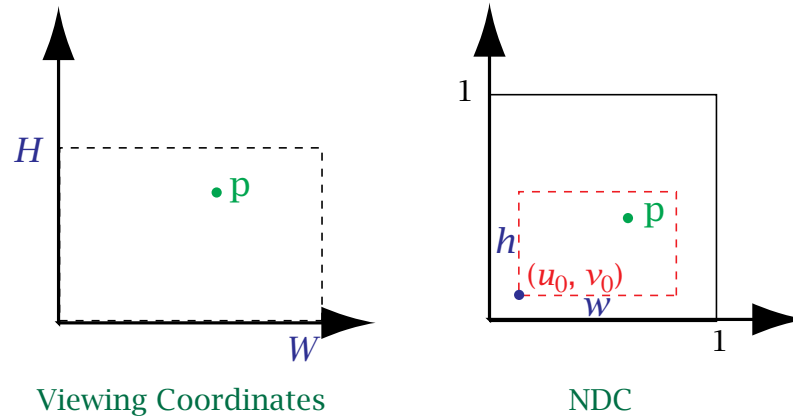
Let \mathbf{p} denote a vertex in the viewing coordinate system. How does one map \mathbf{p} to the **viewport** in the normalized device coordinate system?



$$S \left(\frac{w}{W}, \frac{h}{H} \right) \mathbf{p}$$

Viewing Coordinates \rightarrow NDC Coordinates

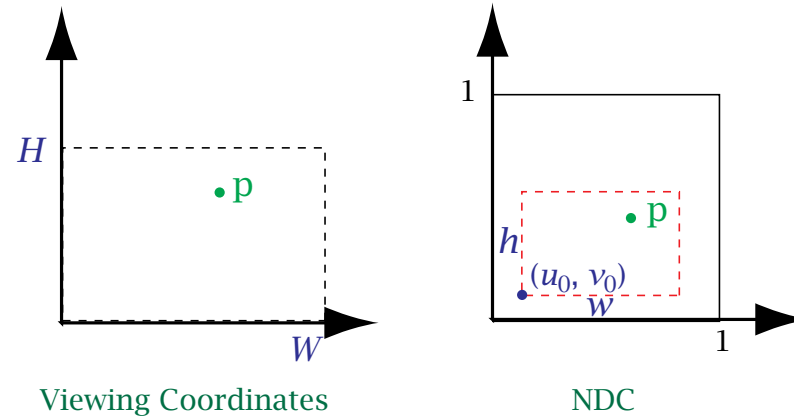
Let \mathbf{p} denote a vertex in the viewing coordinate system. How does one map \mathbf{p} to the **viewport** in the normalized device coordinate system?



$$T(u_0, v_0)S\left(\frac{w}{W}, \frac{h}{H}\right)\mathbf{p}$$

Viewing Coordinates \rightarrow NDC Coordinates

Let \mathbf{p} denote a vertex in the viewing coordinate system. How does one map \mathbf{p} to the **viewport** in the normalized device coordinate system?



$$M_{VC,NDC} = T(u_0, v_0)S\left(\frac{w}{W}, \frac{h}{H}\right)$$

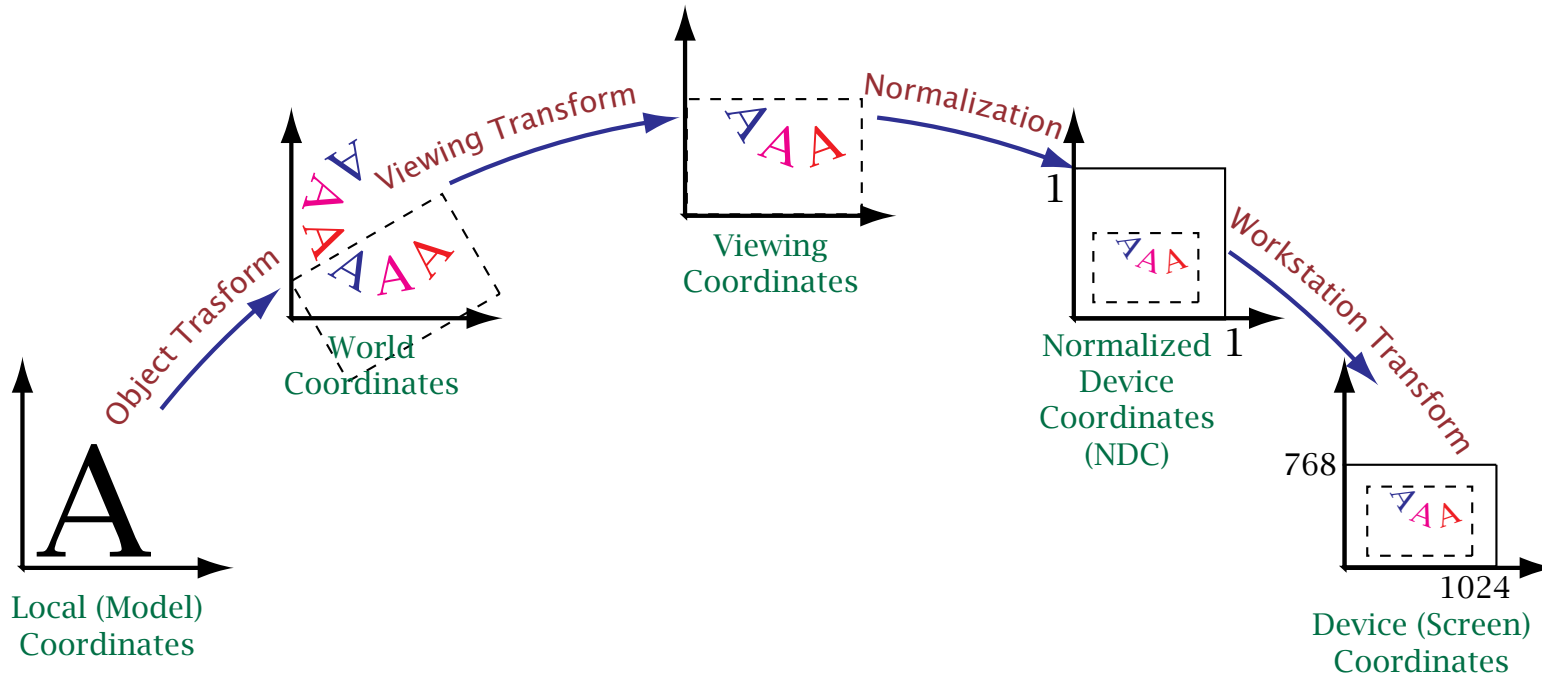
NDC Coordinates \rightarrow Device Coordinates

How this last step is performed depends upon the features of the graphics device. As the latter can be any type of display, printer, or plotter, many possible transformations exist. If we assume the device is a raster display, with r rows and c columns, with an origin at the lower left corner, then the **workstation transformation** might be

$$S(c, r).$$

The Viewing Pipeline (2-D) Summary

Local C. → World C. → Viewing C. → N.D.C. → Device C.



If \mathbf{p} is a point in the world coordinate system, then the displayed point is

$$S(c, r)T(u_0, v_0)S\left(\frac{w}{W}, \frac{h}{H}\right)R(-\theta)T(-x_0, -y_0)\mathbf{p}.$$