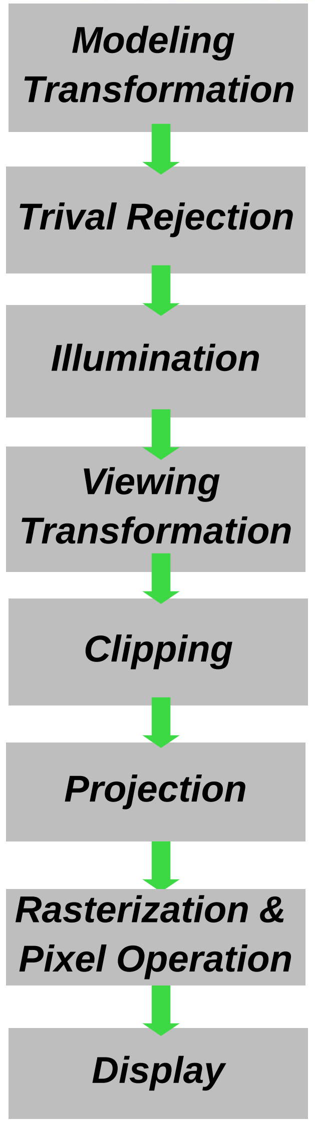
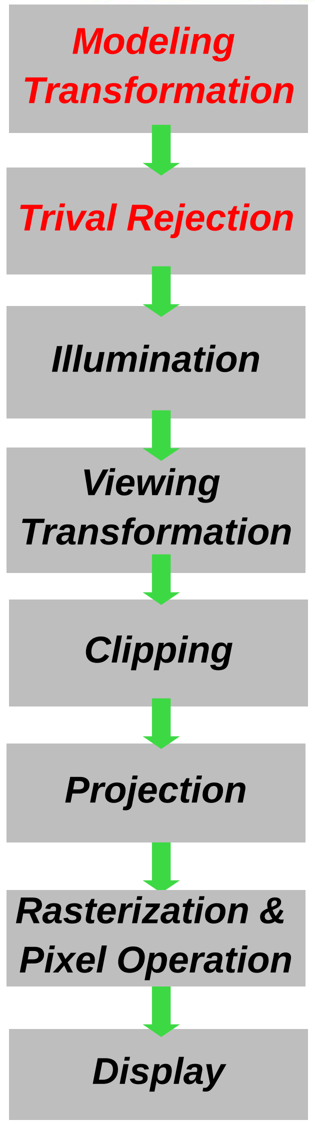
The 3-D Graphics Rendering Pipeline



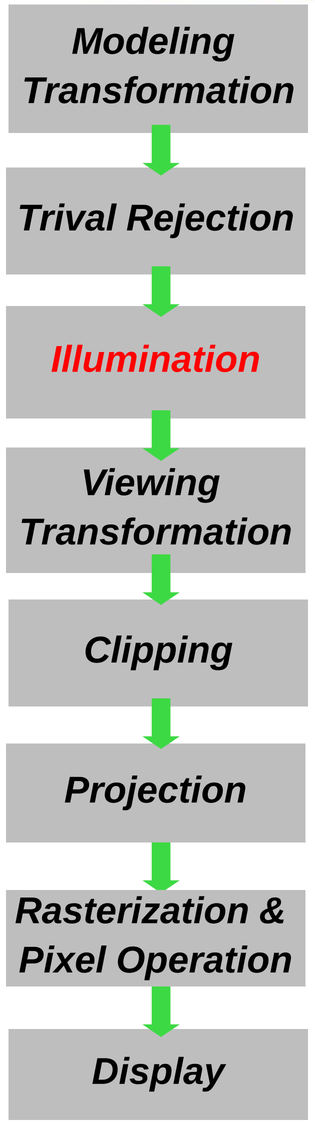
- *Almost every discussion of 3-D graphics begins here*
- *Seldom are any two versions drawn the same way*
- *Seldom are any two versions implemented the same way*
- *Primitives are processed in a series of steps*
- *Each step forwards its result on to the next step*

Modeling transformations



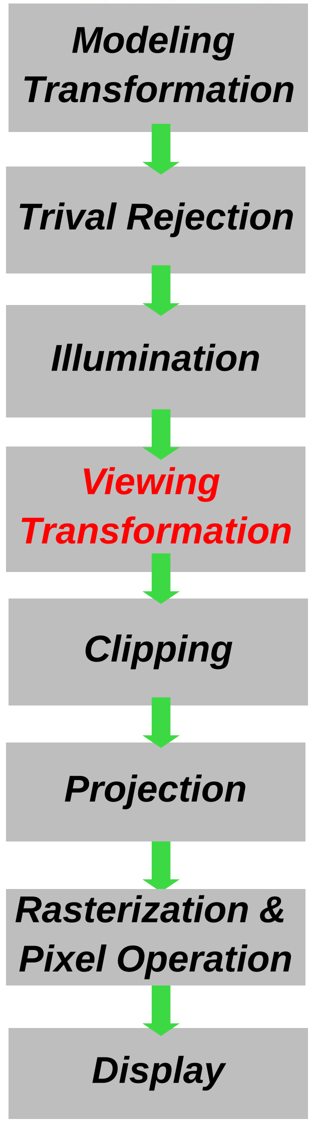
- *We start with 3-D models defined in their own model space*
- *Modeling transformations orient models within a common coordinate system called world space*
- *All objects, light sources, and the viewer live in world space*
- *Trivial rejection attempts to eliminate objects that cannot possibly be seen (an optimization)*

Illumination



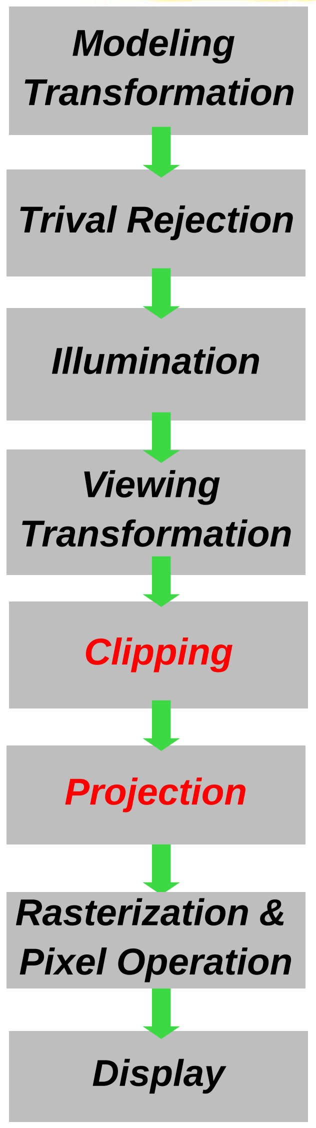
- *Next we illuminate potentially visible objects*
- *Object colors are determined by their material properties, and the light sources in the scene*
- *Illumination algorithm depends on the shading model and the surface model*

Viewing Transformation



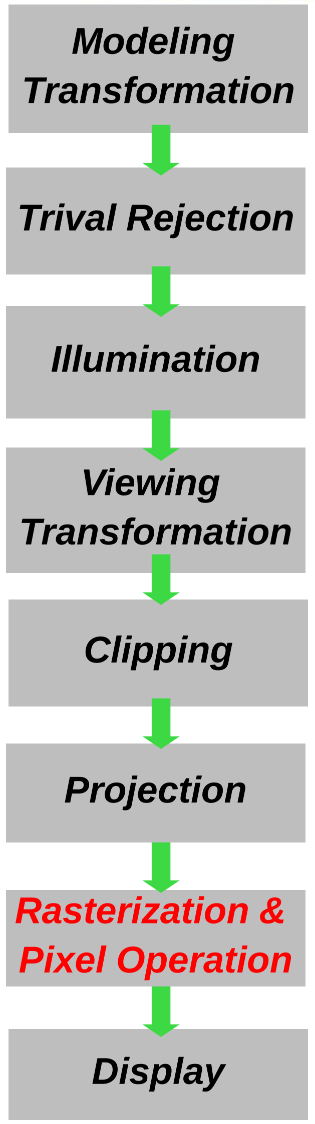
- *Another change of coordinate systems*
- *Maps points from world space into eye space*
- *Viewing position is transformed to the origin*
- *Viewing direction is oriented along some axis*
- *A viewing volume is defined*

Clipping and Projection



- *Next we perform clipping of the scene's objects against a three dimensional viewing volume called a viewing frustum*
- *This step totally eliminates any objects (and pieces of objects) that are not visible in the image*
- *A clever trick is used to straighten out the viewing frustum in to a cube*
- *Next the objects are projected into two-dimensions*
- *Transformation from eye space to screen space*

Rasterization & Pixel Operation

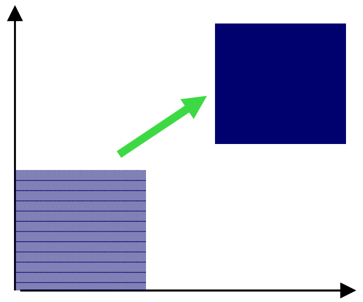


- *One last transformation from our screen-space coordinates into a viewport coordinates*
- *The rasterization step scan converts the object into pixels*
- *Involve interpolating parameters as we go*
- *Purely 2D operation*
- *A lot going on here*

Transformation

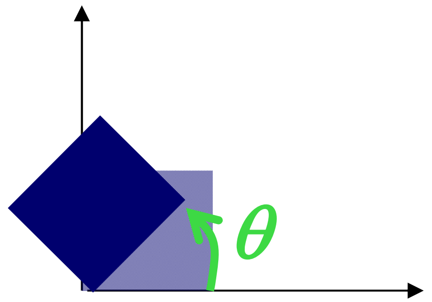
1. 2D Transformation
2. 3D Transformation
3. Viewing Projection

2D Translation



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

2D Rotation

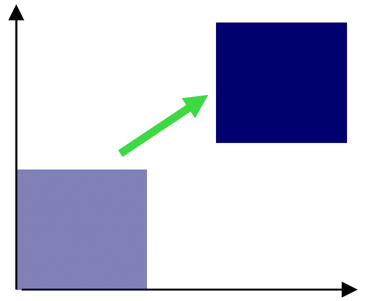


$$\begin{cases} x' = x \cdot \cos\theta \text{ - } y \cdot \sin\theta \\ y' = x \cdot \cos\theta + y \cdot \sin\theta \end{cases}$$

Matrix and Vector format:

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

Back to Translation



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

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

Matrix format?

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = M \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ?? & ?? \\ ?? & ?? \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Homogenous coordinates!

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = M \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2D Translation Properties

1. There exists an inverse mapping for each function
2. There exists an identity mapping

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

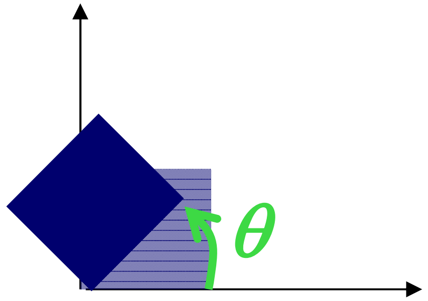
$$M\Big|_{\substack{t_x=0 \\ t_x=0}} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = Identity(I)$$

2D Translation Properties

These properties might seem trivial at first glance, but they are actually very important, because when these conditions are shown for any class of functions it can be proven that such a class is closed under composition (i.e. any series of translations can be composed to a single translation). In mathematical parlance this is the same as saying that translations form an algebraic group.

$$x' = \underbrace{T_1 T_2 \bullet \cdots T_n}_{T'} x$$

Back to Rotation



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

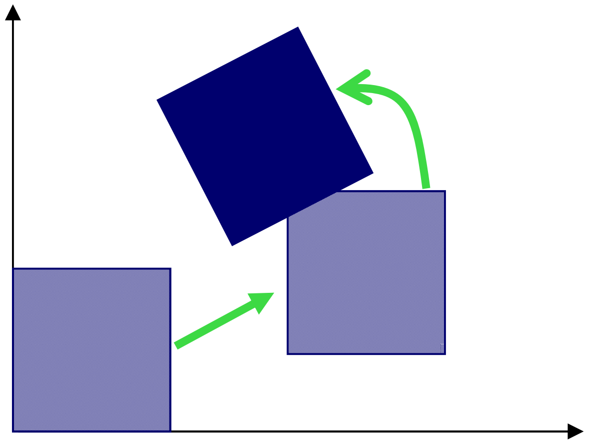
$$M_R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

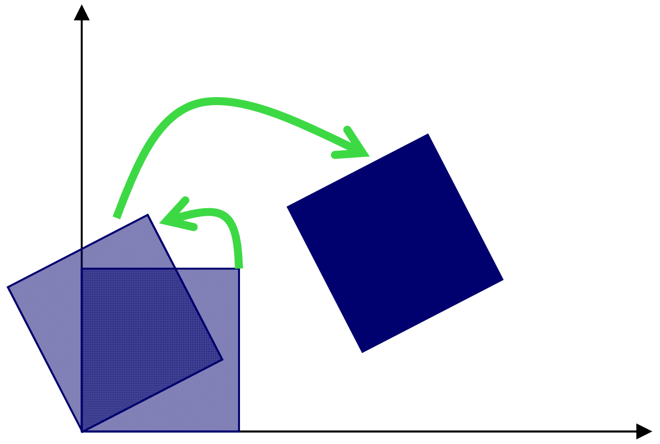
$$M_R\big|_{\theta=0} = Identity$$

Transformation Order

Order matters!

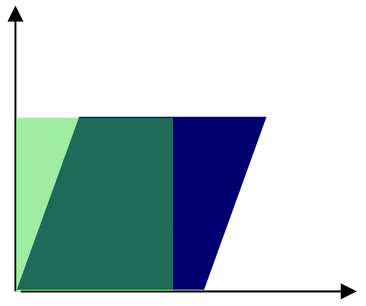


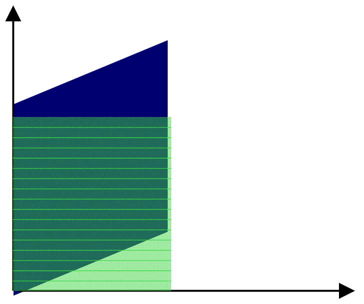
translation ---> rotation

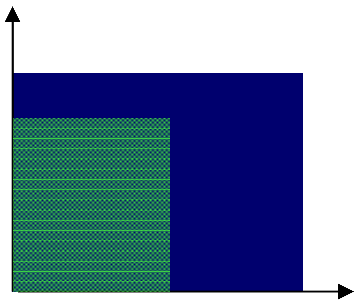


rotation ---> translation

Other 2D Transformations

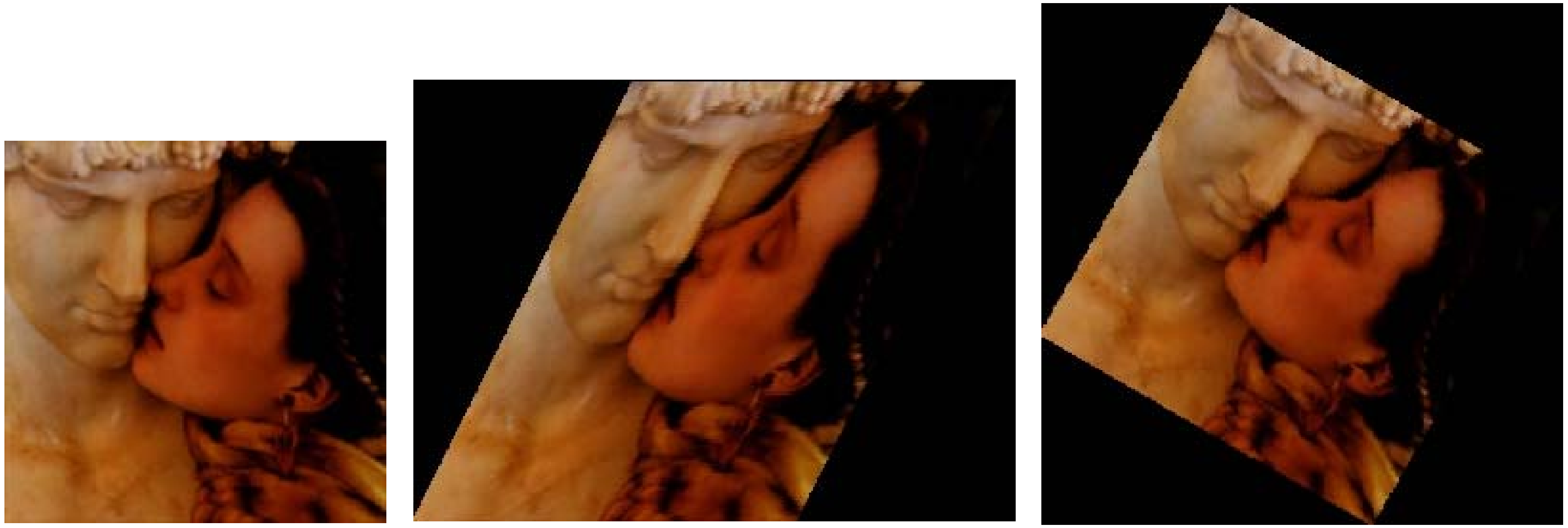

X-shear


Y-shear


scaling

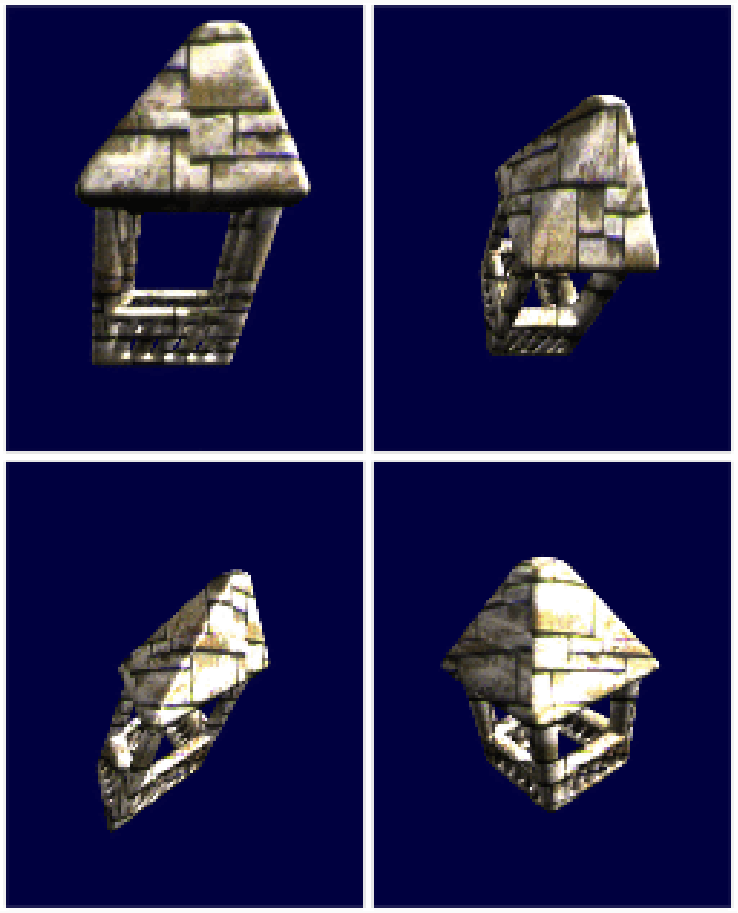
And more …….

2D Rotation by Shears



http://www.cs.sdu.edu.cn/~baoquan/papers/rot2p.pdf

3D Rotation by Shears



http://www.cs.sdu.edu.cn/~baoquan/papers/rot.pdf

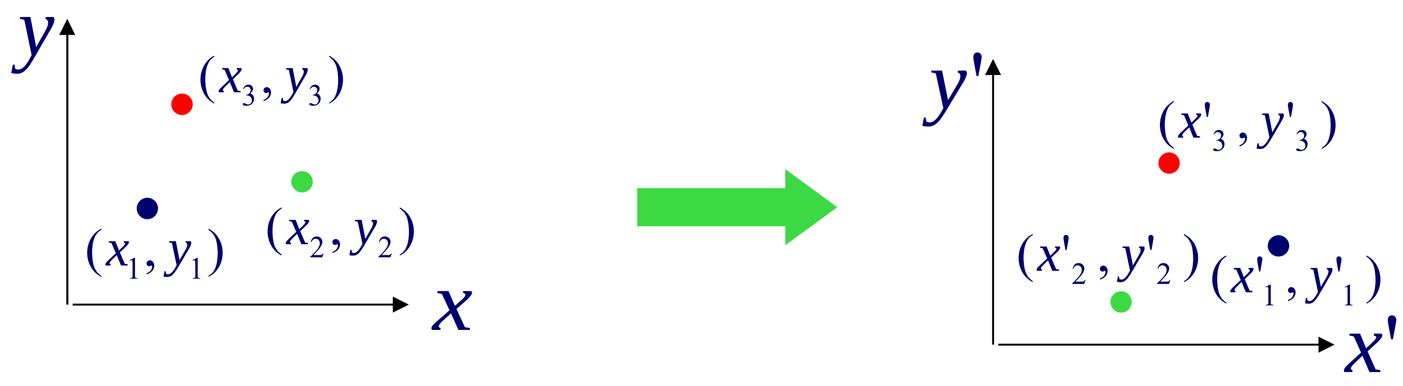
Affine transformation

Property: preserve parallel lines

Remember affine function on vector is equal to linear plus translation

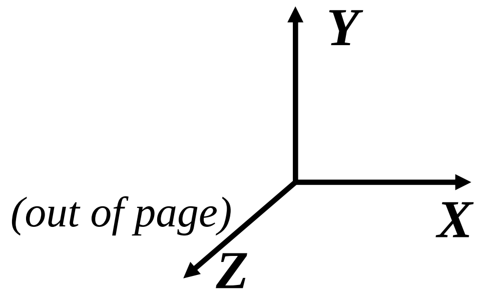
$$M = \begin{bmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ 0 & 0 & 1 \end{bmatrix}$$

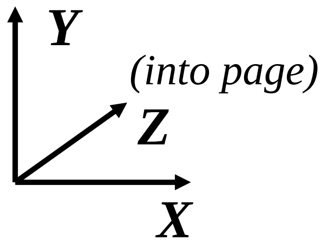
The coordinates of three corresponding points uniquely determine *any* Affine Transform!!



From 2D to 3D: Preliminary

- Right-handed *vs.* left-handed





- Z-axis determined from X and Y by cross product: Z=X×Y

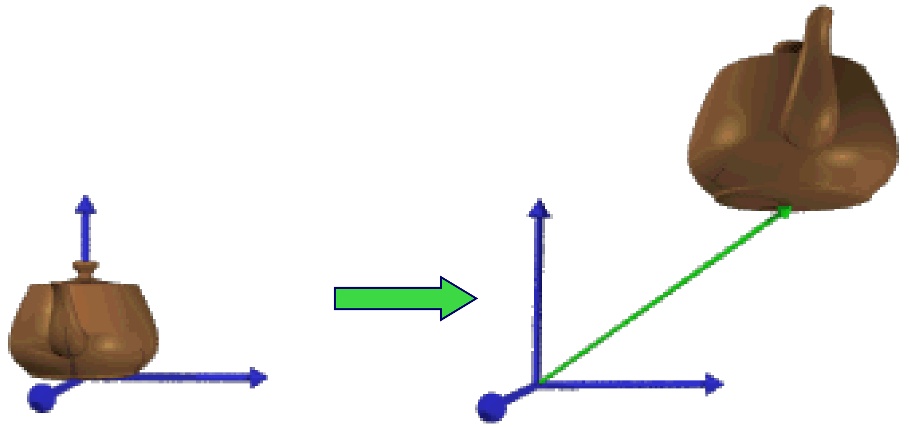
$$\mathbf{Z} = \mathbf{X} \times \mathbf{Y} = \begin{bmatrix} X_2 Y_3 - X_3 Y_2 \\ X_3 Y_1 - X_1 Y_3 \\ X_1 Y_2 - X_2 Y_1 \end{bmatrix}$$

- Cross product follows right-hand rule in a right-handed coordinate system, and left-hand rule in left-handed system.

Transformation

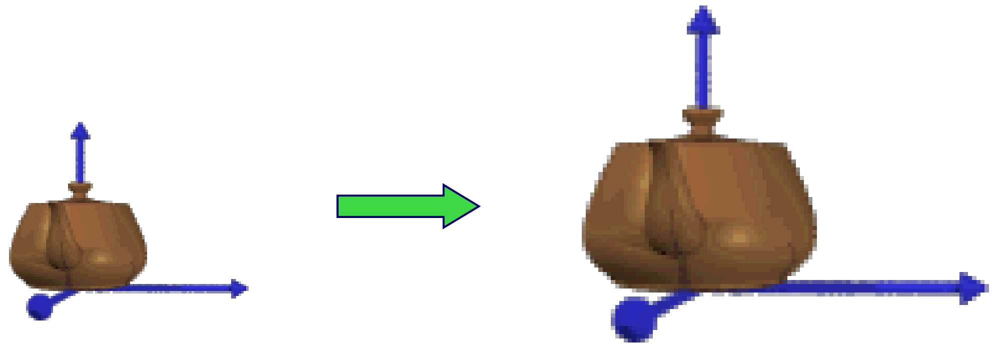
1. 2D Transformation
2. 3D Transformation
3. Viewing Projection

3D Translation



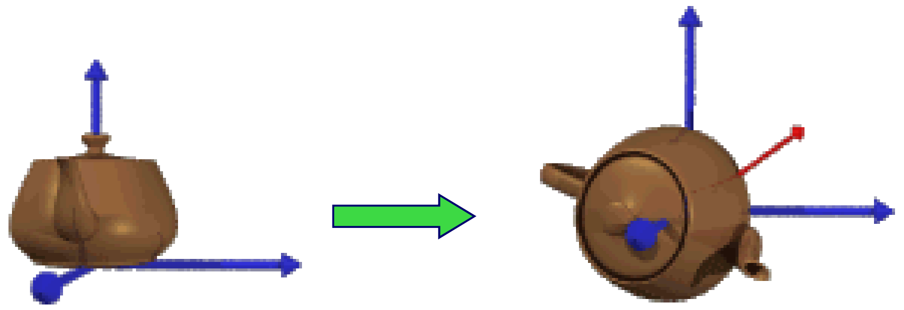
$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & \mathbf{t}_0 \\ 0 & 1 & 0 & \mathbf{t}_1 \\ 0 & 0 & 1 & \mathbf{t}_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Scaling



$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_0 & 0 & 0 & 0 \\ 0 & \mathbf{s}_1 & 0 & 0 \\ 0 & 0 & \mathbf{s}_2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Rotation



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

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