

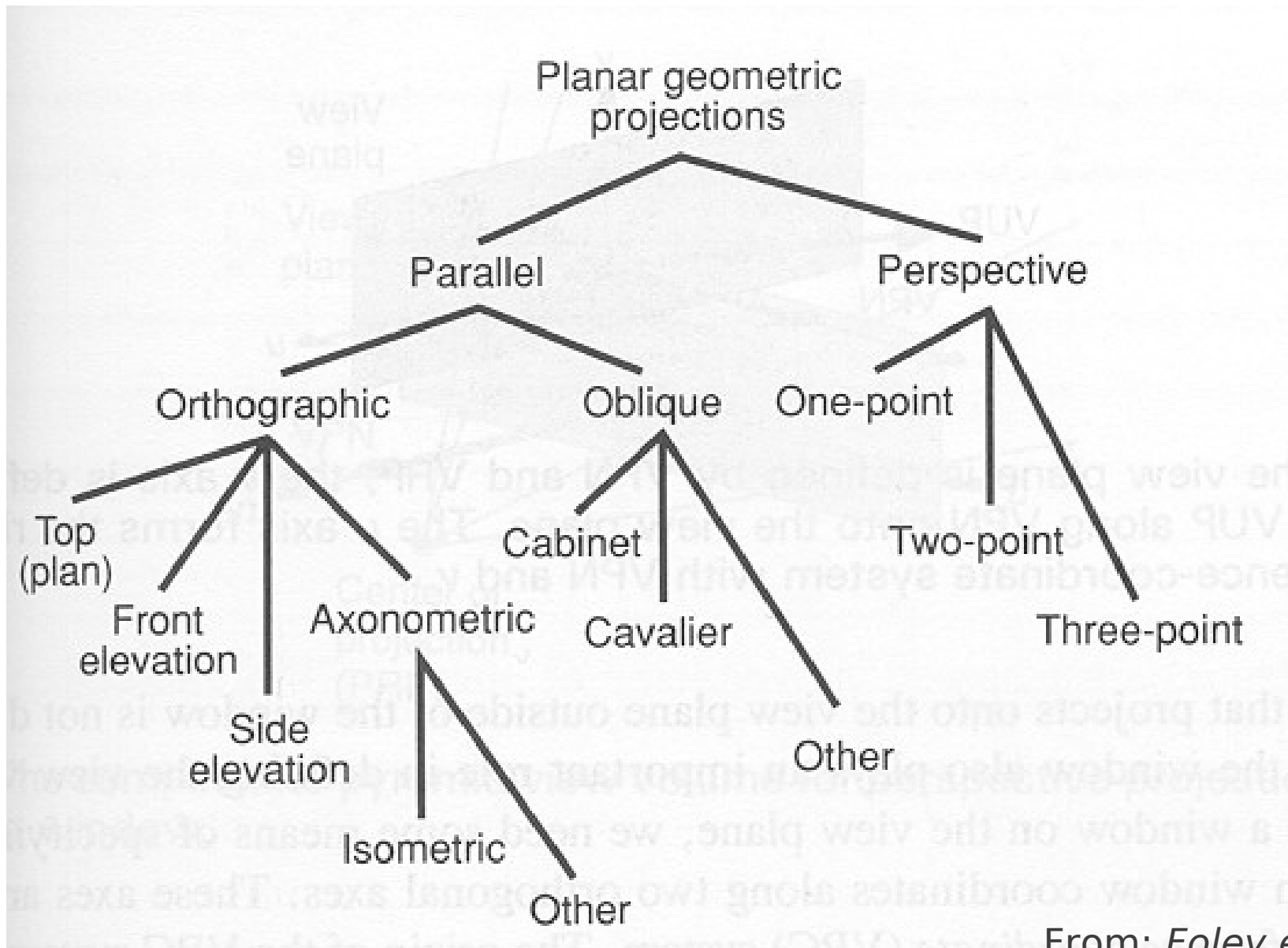
Drawing in 3D Projections

**CSCI 4229/5229
Computer Graphics
Fall 2013**

Types of Projections

- Parallel Projections
 - Orthogonal, isometric, ...
 - Size does not diminish with distance
- Perspective
 - Realistic based on an observer's point of view
 - Nearer bigger, farther smaller
 - One or more vanishing points

Taxonomy of Projections

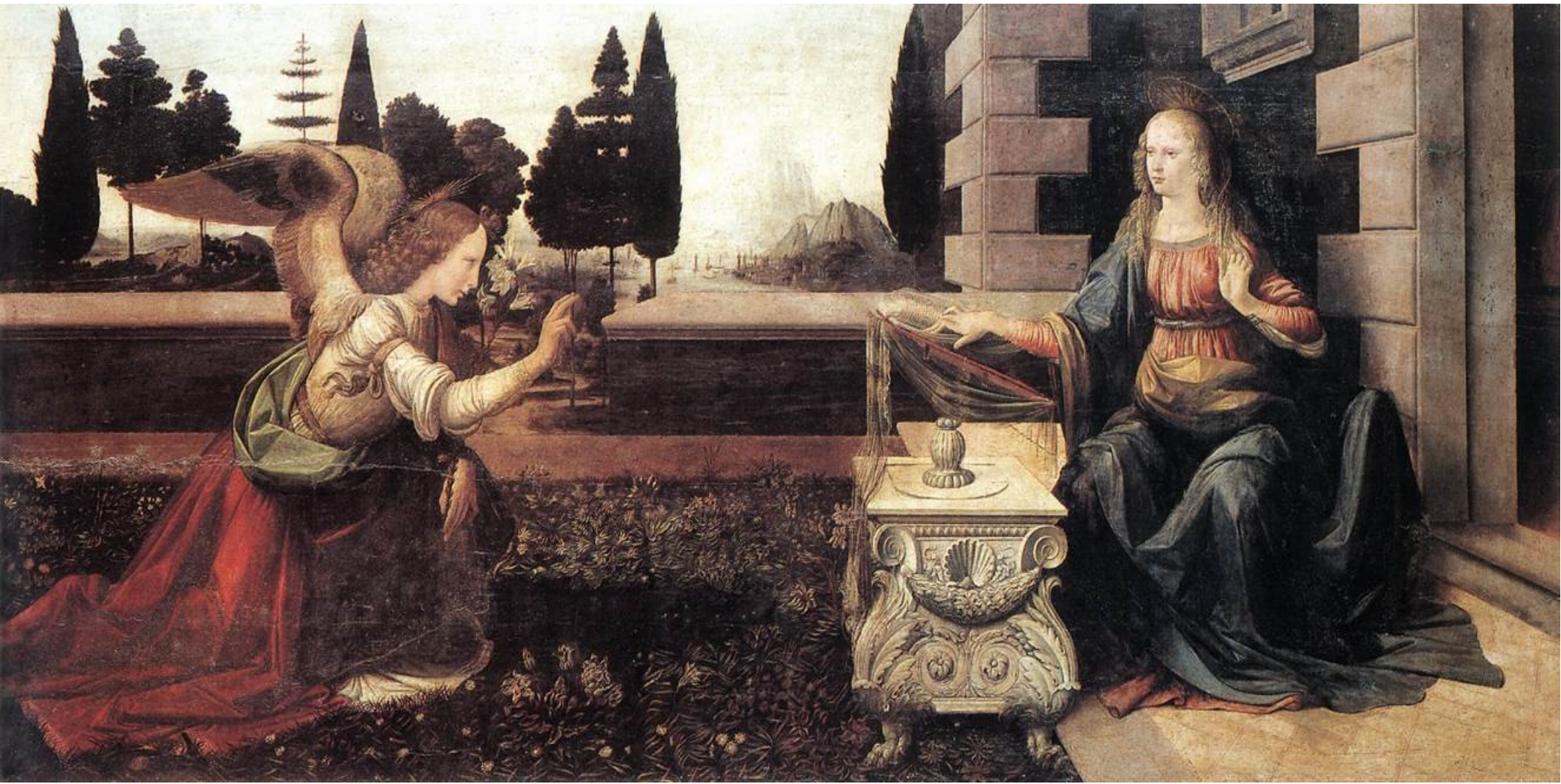


From: *Foley et al*

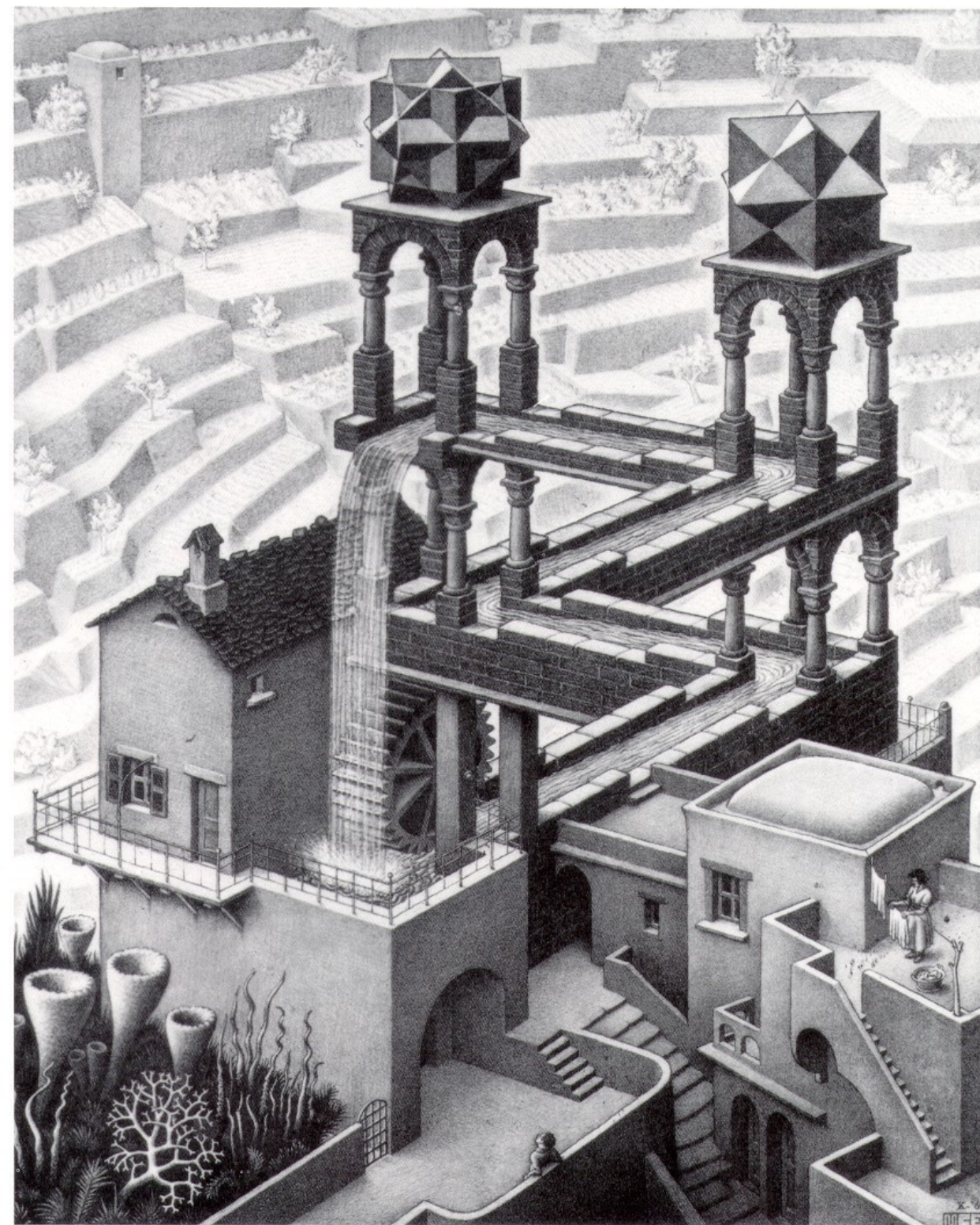
Egyptian Tomb Painting



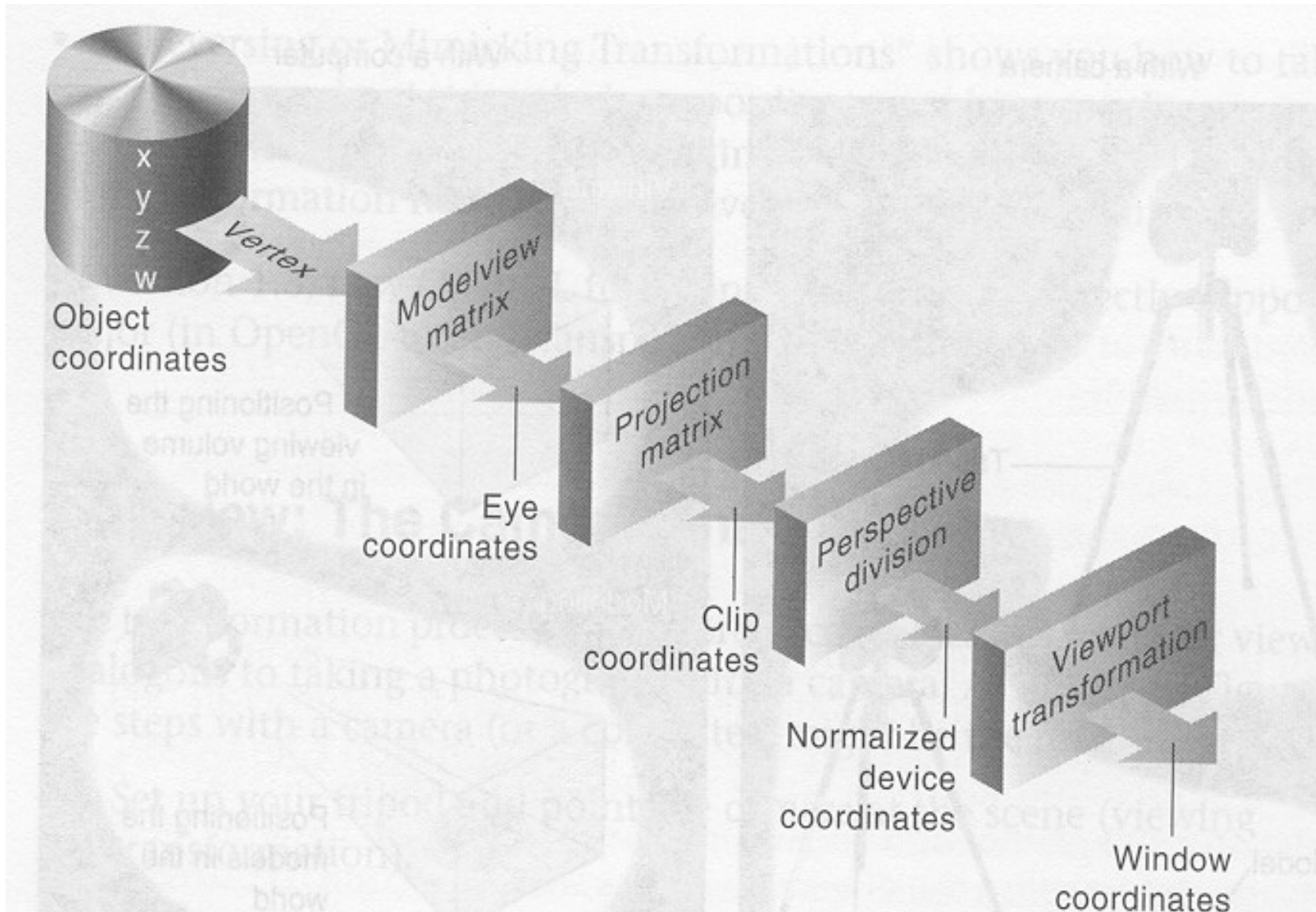
Annunciation Leonardo da Vinci (1472)



Waterfall
M.C.
Escher (1961)

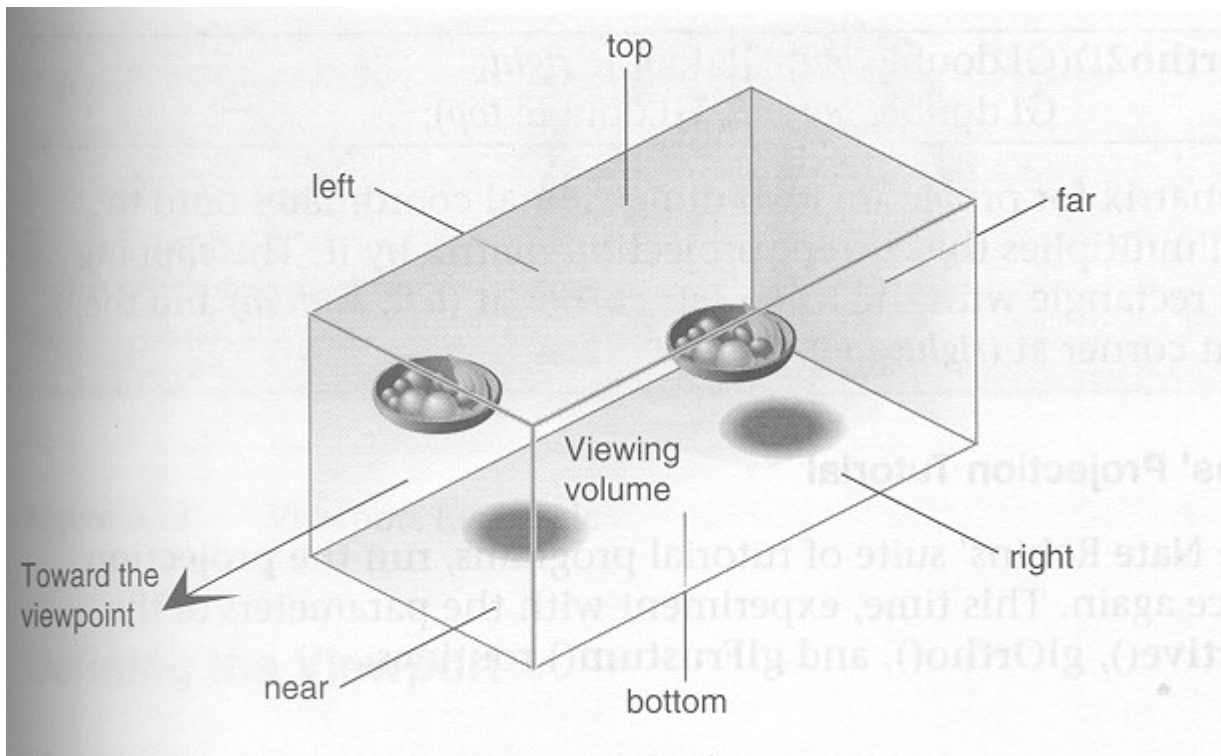


OpenGL Transformation Pipeline



Parallel Projection

- Apply rotation matrix to map direction of projection to Z axis and up to Y axis
- Scale to canonical volume



From: *OpenGL Red Book*

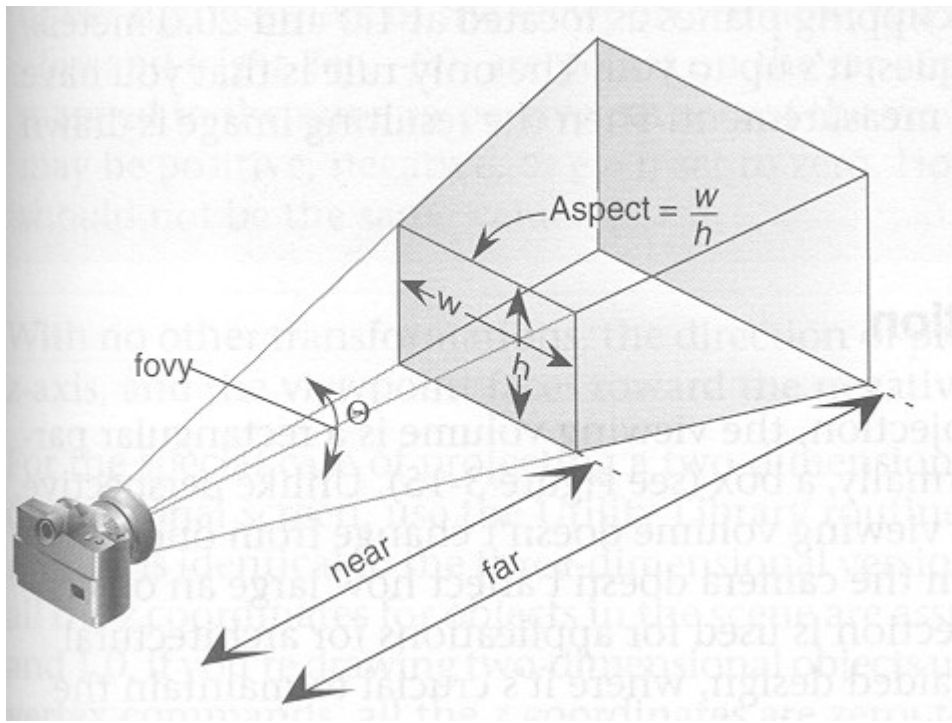
glOrtho(x_{min} , x_{max} , y_{min} , y_{max} , z_{min} , z_{max})

glOrtho Projection Matrix

$$\begin{pmatrix} \frac{2}{x_{max}-x_{min}} & 0 & 0 & -\frac{x_{max}+x_{min}}{x_{max}-x_{min}} \\ 0 & \frac{2}{y_{max}-y_{min}} & 0 & -\frac{y_{max}+y_{min}}{y_{max}-y_{min}} \\ 0 & 0 & \frac{-2}{z_{max}-z_{min}} & \frac{z_{max}+z_{min}}{z_{max}-z_{min}} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

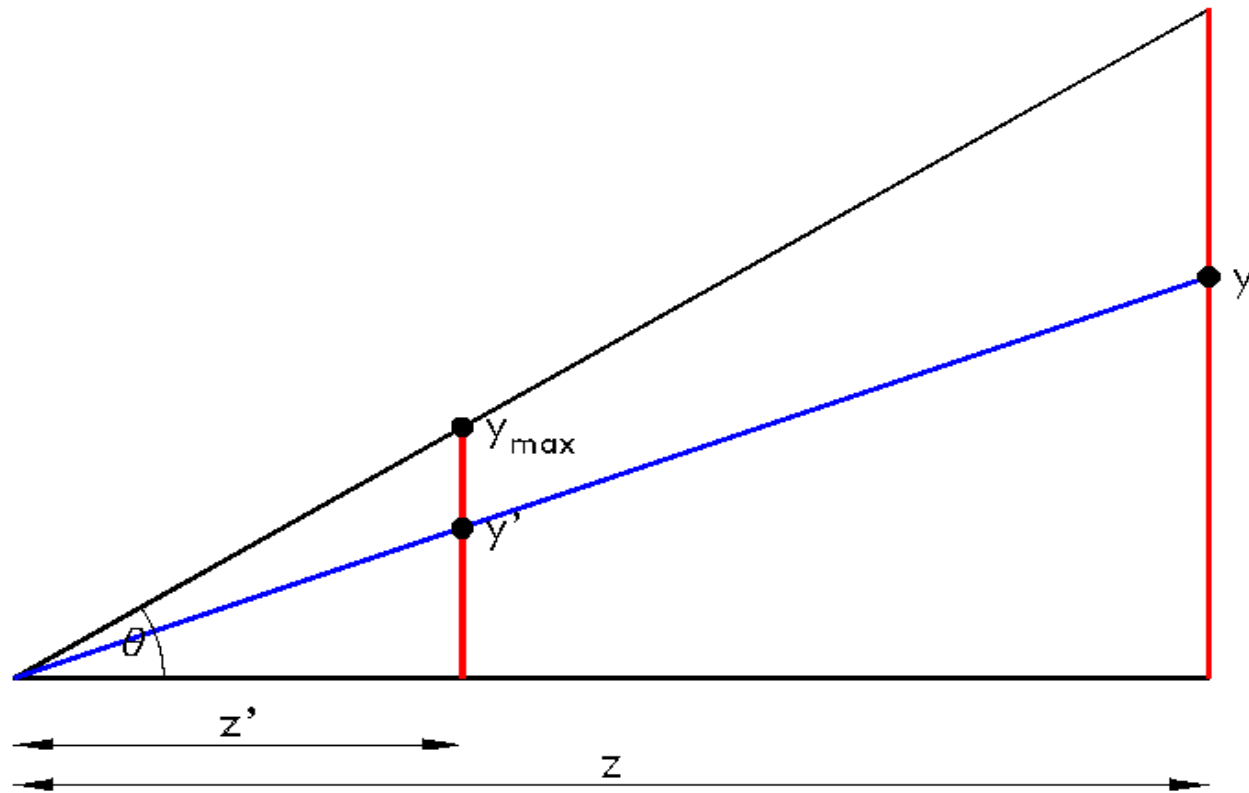
Perspective Transformation

- Apply rotation matrix to map eye position to center of scene to negative Z and up to Y axes
- Scale (x,y) inversely proportional to distance
- Scale to canonical volume



From: *OpenGL
Red Book*

Perspective Transformation



Similar triangles: $y'/z' = y/z$, so $y' = y/z z'$

Let $y_{max} = 1$ (NDC), $\tan \theta = y_{max} / z'$, $z' = \cot \theta$

$$y' = \cot \theta y/z$$

Homogeneous Perspective Multiply

$$\begin{pmatrix} \frac{\cot \theta}{\text{aspect}} & 0 & 0 & 0 \\ 0 & \cot \theta & 0 & 0 \\ 0 & 0 & \frac{z_{\text{far}} + z_{\text{near}}}{z_{\text{far}} - z_{\text{near}}} & \frac{2z_{\text{far}}z_{\text{near}}}{z_{\text{far}} - z_{\text{near}}} \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} \frac{\cot \theta}{\text{aspect}} x \\ \cot \theta y \\ \frac{z_{\text{far}} + z_{\text{near}}}{z_{\text{far}} - z_{\text{near}}} z + \frac{2z_{\text{far}}z_{\text{near}}}{z_{\text{far}} - z_{\text{near}}} \\ -z \end{pmatrix}$$

$$\equiv \begin{pmatrix} \frac{\cot \theta}{\text{aspect}} \frac{x}{z} \\ \cot \theta \frac{y}{z} \\ \frac{-2z_{\text{far}}z_{\text{near}}}{z(z_{\text{far}} - z_{\text{near}})} - \frac{z_{\text{far}} + z_{\text{near}}}{z_{\text{far}} - z_{\text{near}}} \end{pmatrix}$$

gluPerspective(*fovy*, *aspect*, *Znear*, *Zfar*)

- *fovy* is the angle in the up/down direction
- *aspect* is the horizontal to vertical ratio
- *Znear* is the distance to the near clipping plane
 - Killer fact $Znear > 0$
- *Zfar* is the distance to the far clipping plane
 - $Zfar > Znear$
- $Zfar - Znear$ determines *Z* resolution since the *Z* buffer has finite precision
 - $\log_2(Zfar/Znear)$ bits lost

gluPerspective(fovy, aspect, Znear, Zfar)

Let $\theta = \text{fovy}/2$

gluPerspective Projection Matrix

$$\begin{pmatrix} \frac{\cot \theta}{\text{aspect}} & 0 & 0 & 0 \\ 0 & \cot \theta & 0 & 0 \\ 0 & 0 & -\frac{z_{\text{far}} + z_{\text{near}}}{z_{\text{far}} - z_{\text{near}}} & \frac{2z_{\text{far}}z_{\text{near}}}{z_{\text{far}} - z_{\text{near}}} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

$\text{gluLookAt}(E_x, E_y, E_z, C_x, C_y, C_z, U_x, U_y, U_z)$

- (E_x, E_y, E_z) is the eye position
- (C_x, C_y, C_z) is the position you look at
- (U_x, U_y, U_z) is the up direction
- $C-E$ determines the distance in the Z direction
- The Z distance to each object (from E) determines the reduction in the (x, y) direction

`gluLookAt($E_x, E_y, E_z, C_x, C_y, C_z, U_x, U_y, U_z$)`

Forward $\mathbf{F} = \mathbf{C} - \mathbf{E}$

Sideways $\mathbf{S} = \mathbf{F} \times \mathbf{U}$

Up $\mathbf{U} = \mathbf{S} \times \mathbf{F}$

$$\begin{pmatrix} x' \\ y' \\ z' \\ w' \end{pmatrix} = \begin{pmatrix} S_x & U_x & -F_x & 0 \\ S_y & U_y & -F_y & 0 \\ S_z & U_z & -F_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -E_x \\ 0 & 1 & 0 & -E_y \\ 0 & 0 & 1 & -E_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix}$$
$$= \begin{pmatrix} S_x & U_x & -F_x & -E_x S_x - E_y U_x + E_z F_x \\ S_y & U_y & -F_y & -E_x S_y - E_y U_y + E_z F_y \\ S_z & U_z & -F_z & -E_x S_z - E_y U_z + E_z F_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix}$$

(\mathbf{F} and \mathbf{U} must be normalized)