

Viewing Transformations

CS425: Computer Graphics I

Fabio Miranda

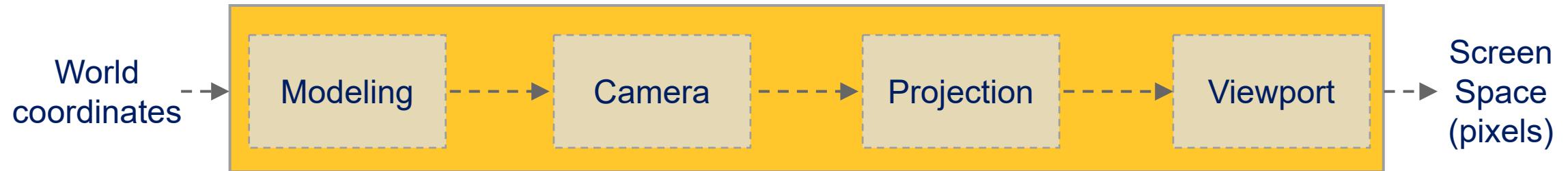
<https://fmiranda.me>

Overview

- Coordinate systems
- Camera
- Viewing transformations
- Orthographic and perspective projections
- Hidden surface removal

Viewing transformations

- Viewing transformation is the mapping of coordinates of points and lines from world coordinates into screen space pixels.



Viewing transformations

- Previously, we saw how transformations can manipulate primitives (points, vectors) in space.
- Today, we will see how transformations can manipulate primitives to 2D screen coordinates (that will be later rasterized).

Perspective projection

Distant objects appear smaller

Parallel lines converge at the horizon



Early paintings



Lorsch Gospels (8th century)

Perspective in art

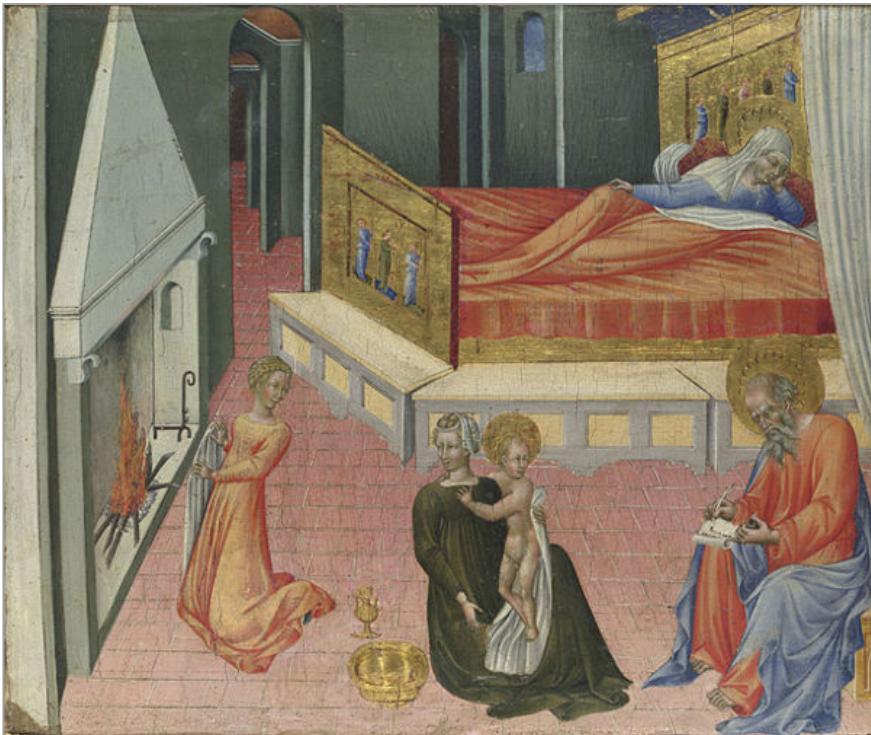


Giotto di Bondone (14th century)



Giotto di Bondone (14th century)

Birth of perspective in art: One-point perspective



Giovanni di Paolo (15th century)



Piero della Francesca (15th century)

Birth of perspective in art: One-point perspective



Perugino (15th century)

Birth of perspective in art: One-point perspective



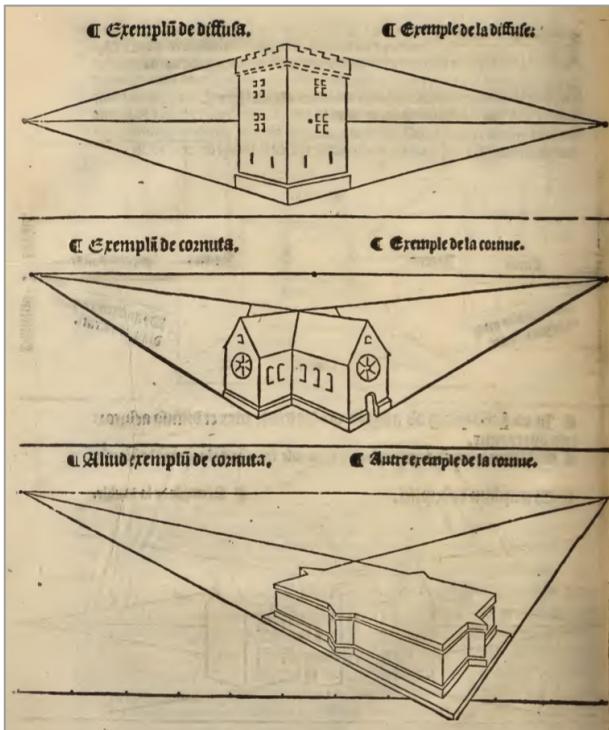
The Ideal City (15th century)

Birth of perspective in art: One-point perspective



Rafael (16th century)

Birth of perspective in art: Two-point perspective

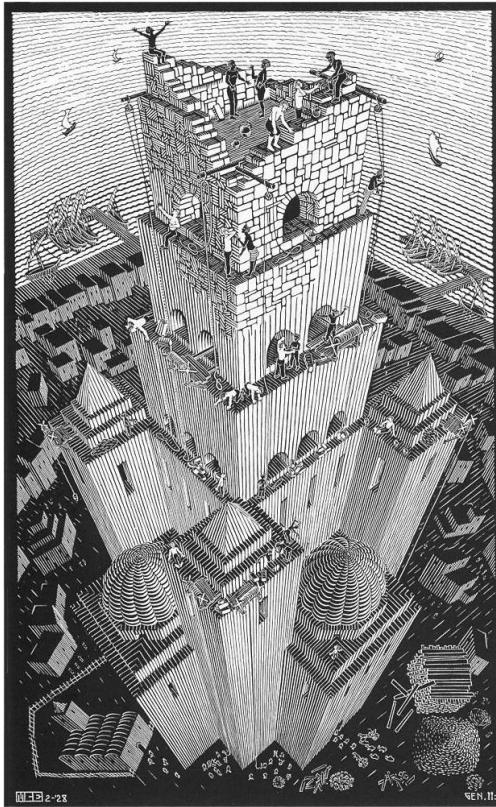


Jean Pélérin (16th century)

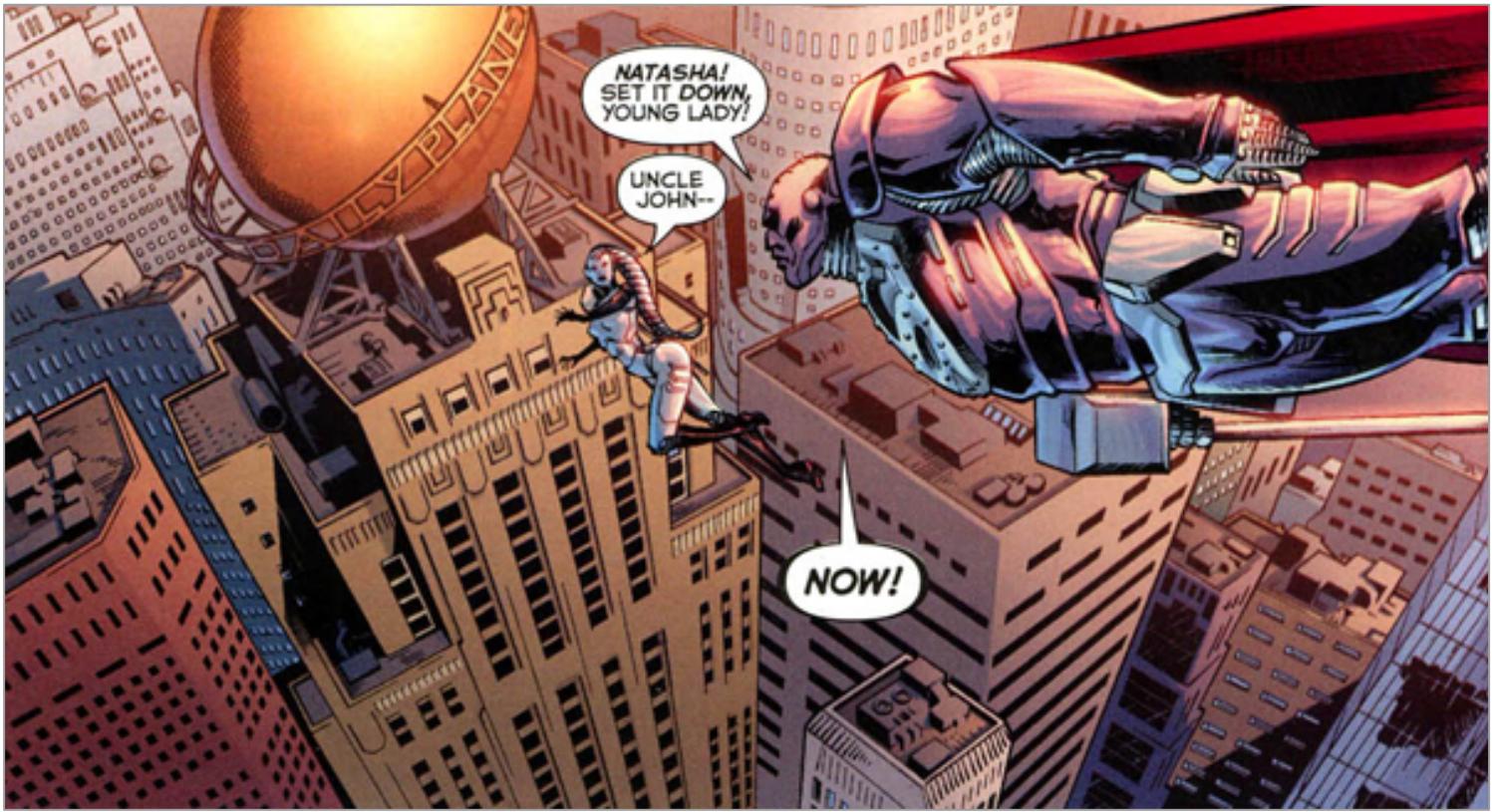


Gustave Caillebotte (19th century)

Birth of perspective in art: Three-point perspective



M.C. Escher (1928)

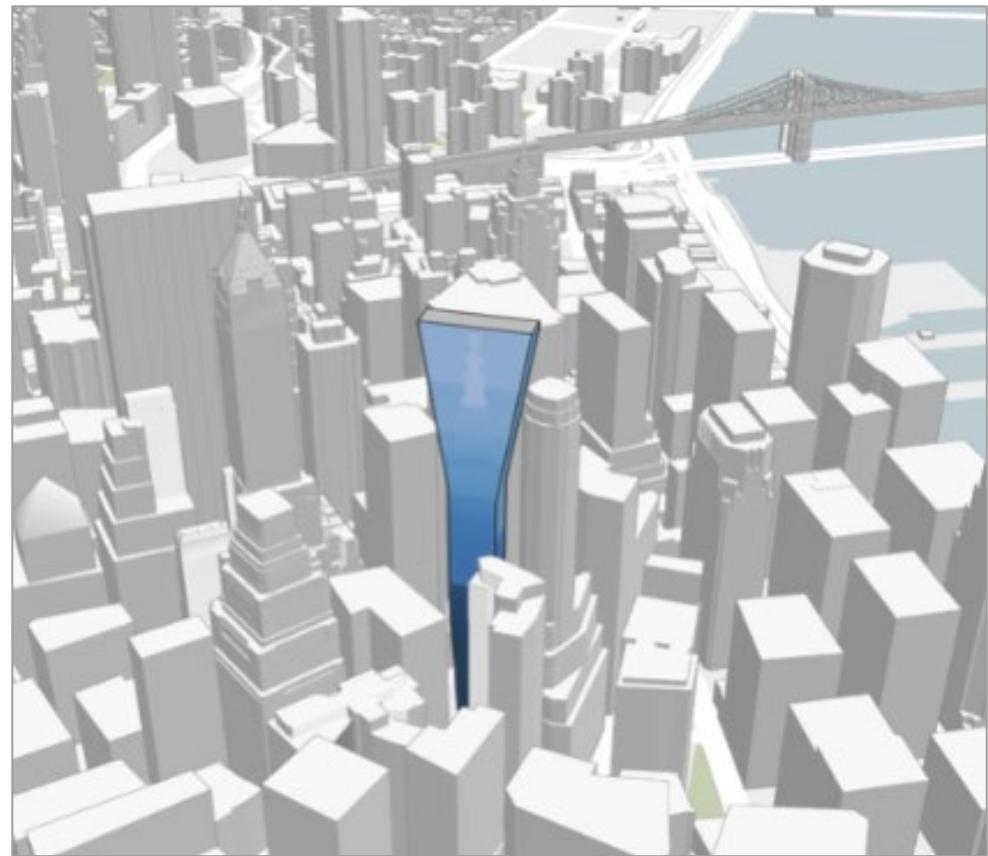
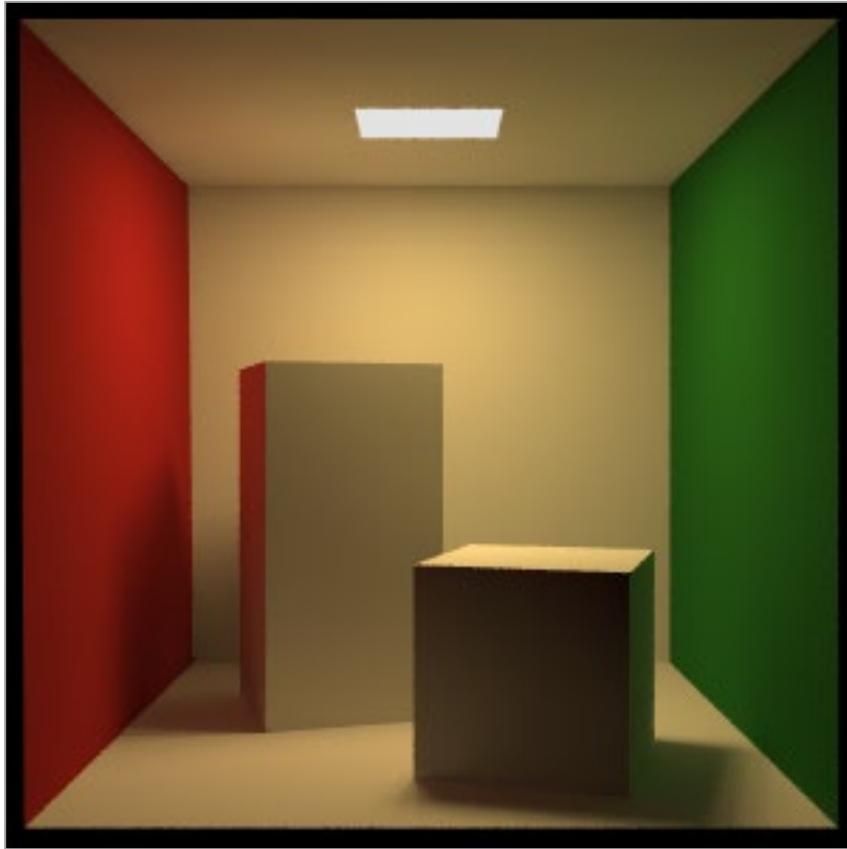


Multi perspective



The Frozen City by
Matthias A. K.
Zimmermann (2006)

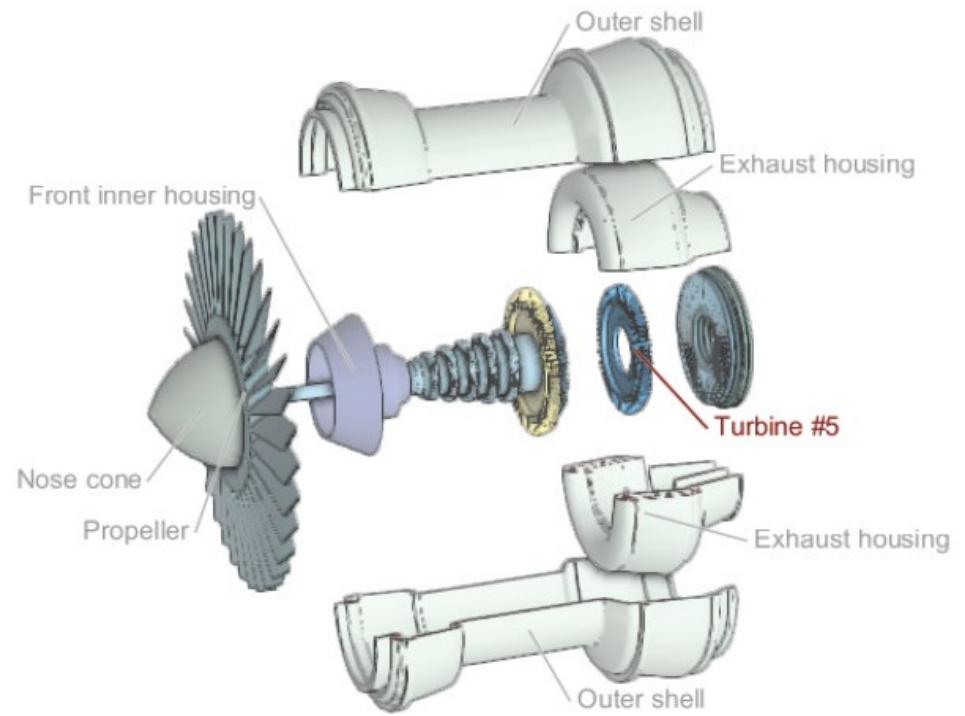
Perspective in computer graphics



Rejection of perspective in CG

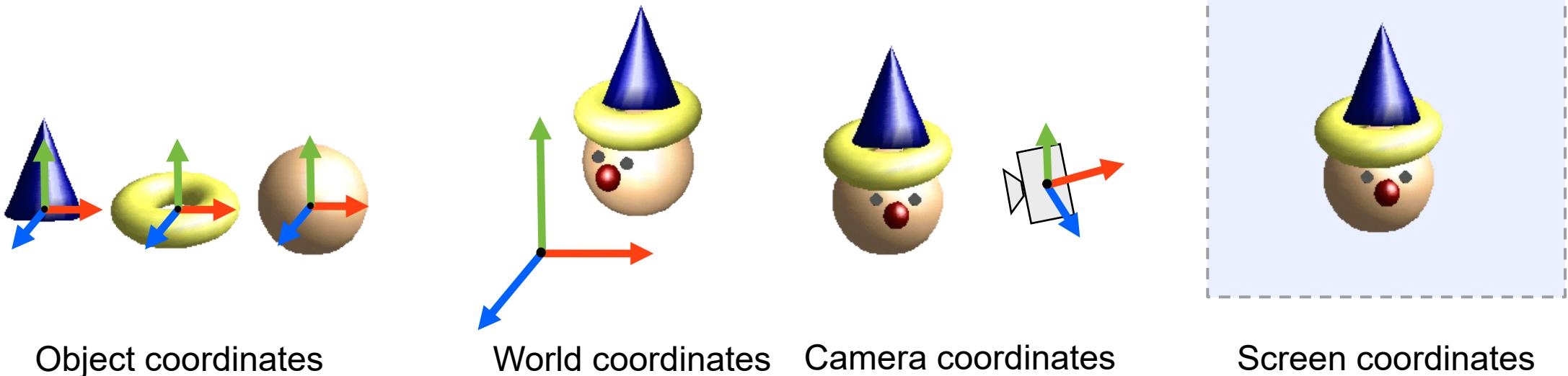


SimCity 2000



“Automated Generation of Interactive 3D
Exploded Views”

Coordinate spaces



Object coordinates

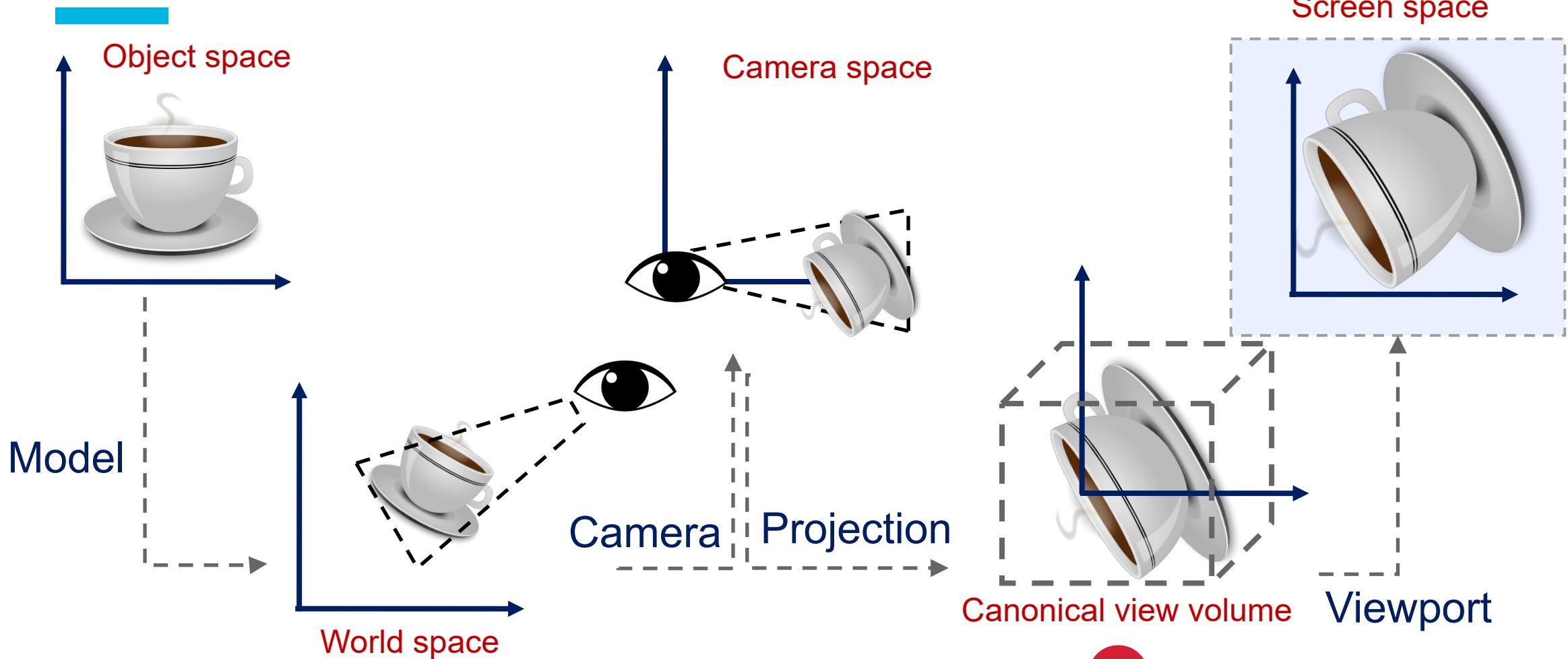
World coordinates

Camera coordinates

Screen coordinates

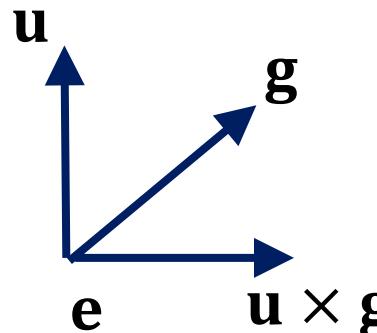
From: Mark Pauly

Viewing transformation



Camera transformation

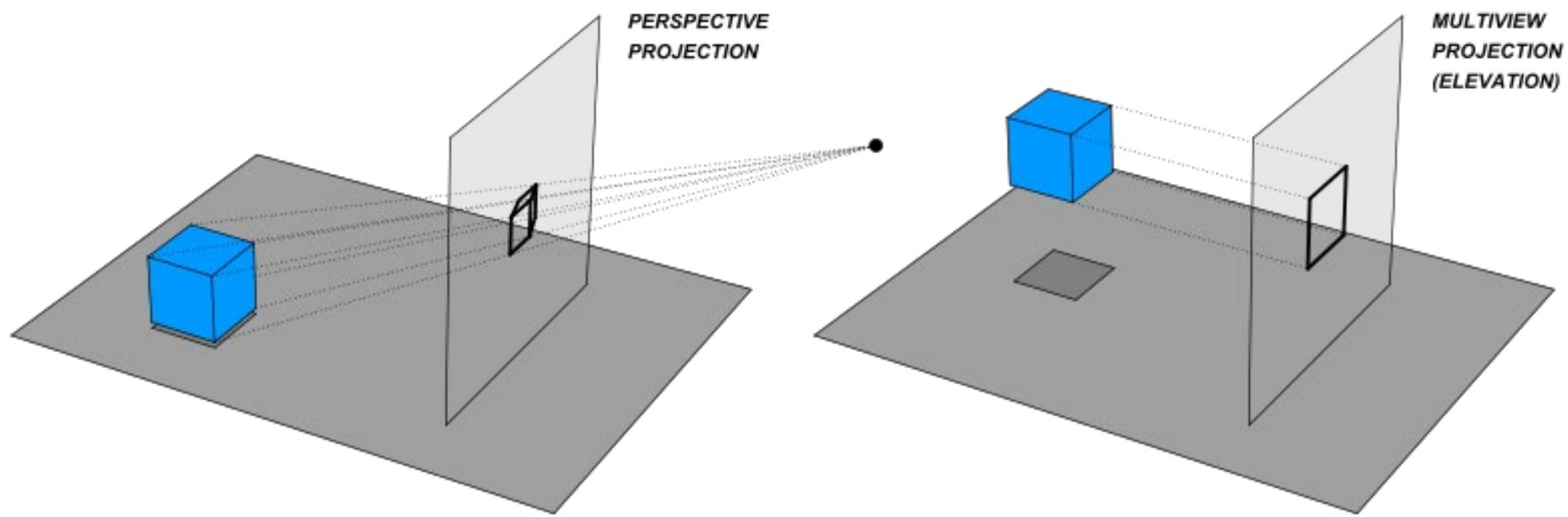
- Construct the camera reference system:
 - The eye position e .
 - The forward direction d .
 - The view-up vector u .
- A view matrix transform all coordinates into view coordinates.



$$\mathbf{M}_{camera2world} = \begin{pmatrix} \mathbf{u} \times \mathbf{g} & \mathbf{u} & \mathbf{d} & \mathbf{e} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

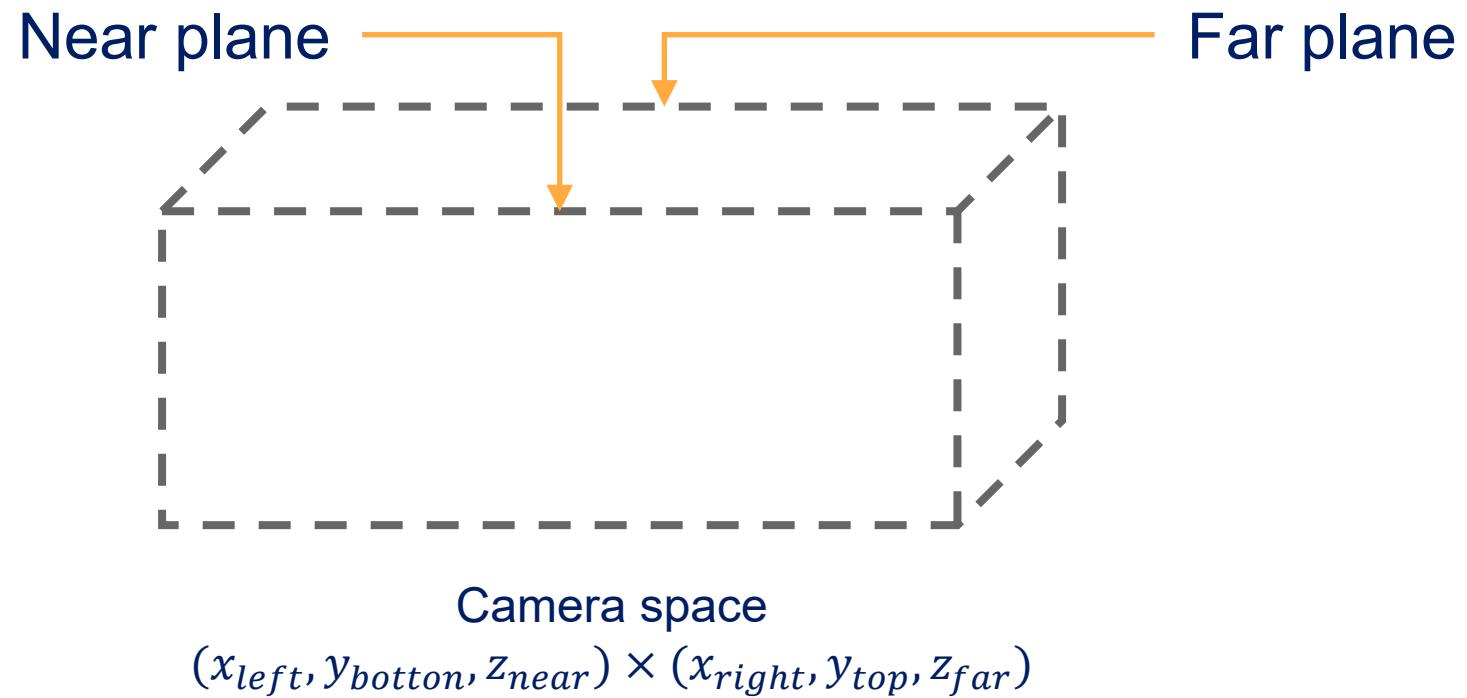
$$\mathbf{M}_{world2camera} = \begin{pmatrix} \mathbf{u} \times \mathbf{g} & \mathbf{u} & \mathbf{d} & \mathbf{e} \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1}$$

Orthographic and perspective projections

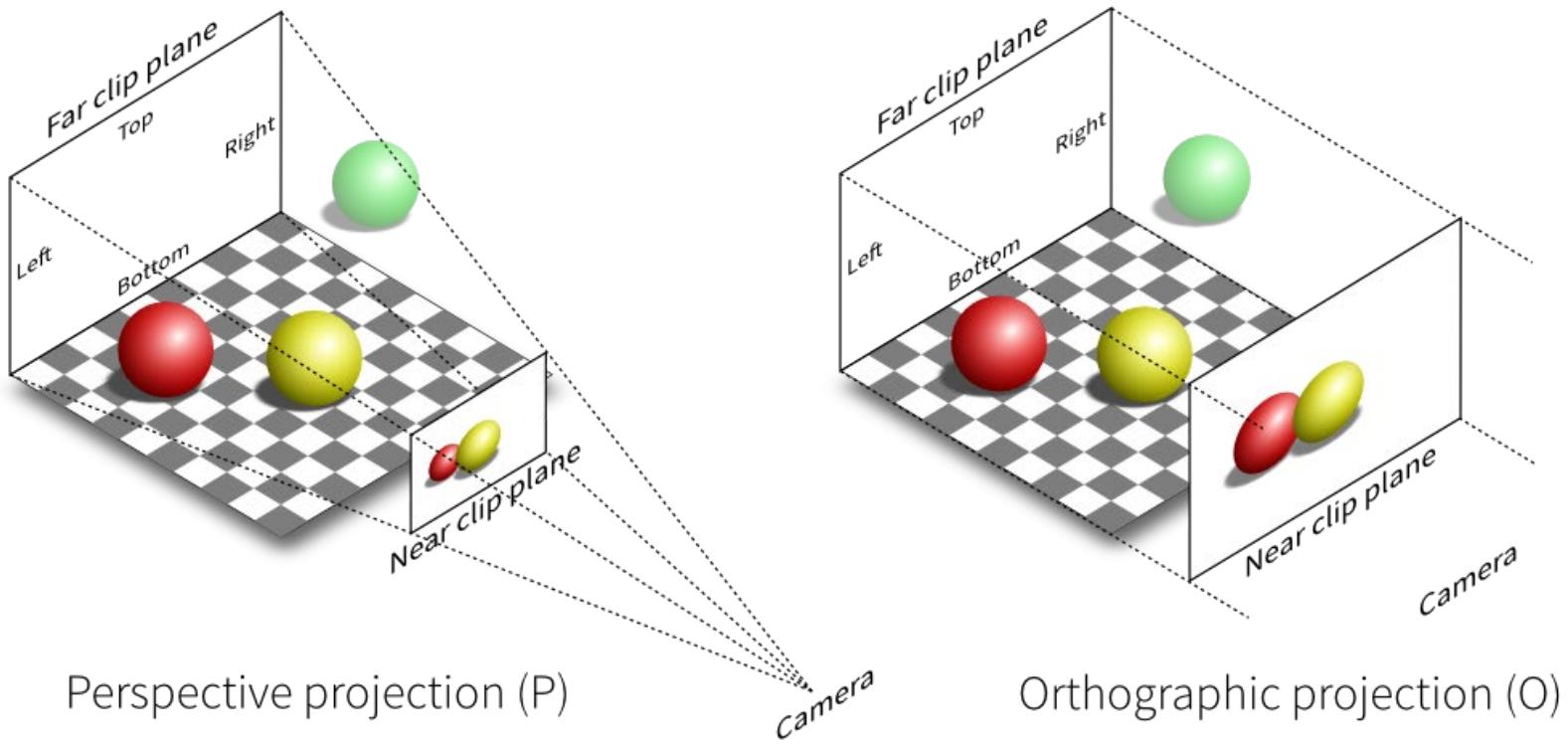


https://en.wikipedia.org/wiki/File:Various_projections_of_cube_above_plane.svg

View frustum



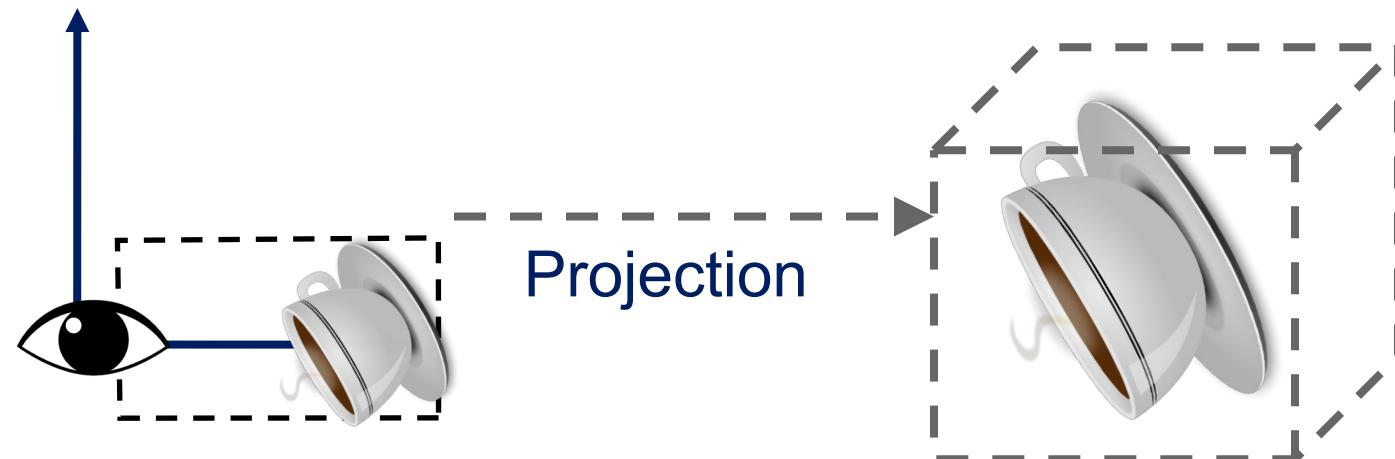
View frustum



Perspective projection (P)

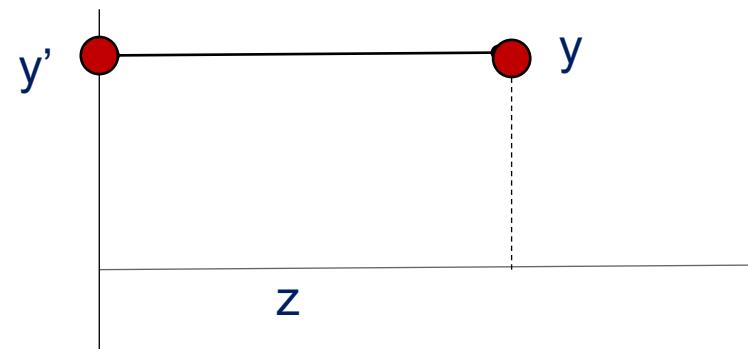
Orthographic projection (O)

Orthographic transformation



$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{2}{(right - left)} & 0 & 0 & -x_{mid} \\ 0 & \frac{2}{(top - bottom)} & 0 & -y_{mid} \\ 0 & 0 & \frac{-2}{(far - near)} & -z_{mid} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

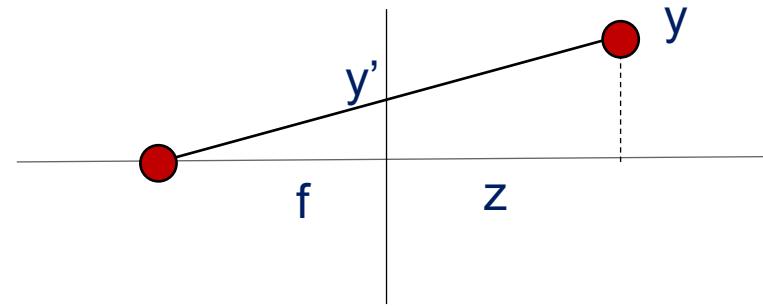
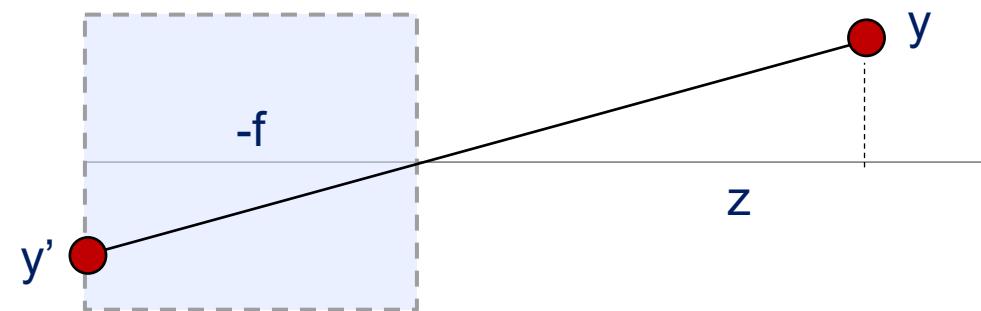
Basic orthographic projection



Projection plane
($z = 0$)

$$\begin{array}{c} \xrightarrow{\quad \text{---} \quad} \\ \boxed{x' = x, y' = y} \end{array}$$
$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Basic perspective projection



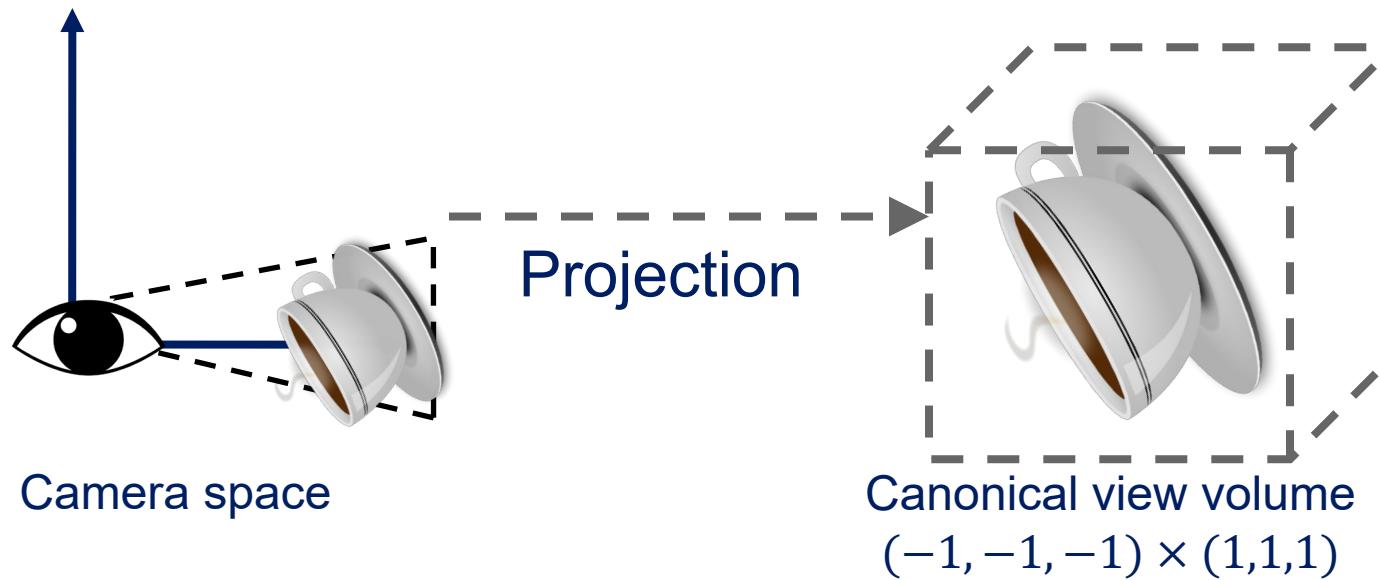
$$x' = f \frac{x}{z}, y' = f \frac{y}{z}$$

if $f = 1$:

$$x' = \frac{x}{z}, y' = \frac{y}{z}, f = 1$$

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ z \\ z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Perspective transformation

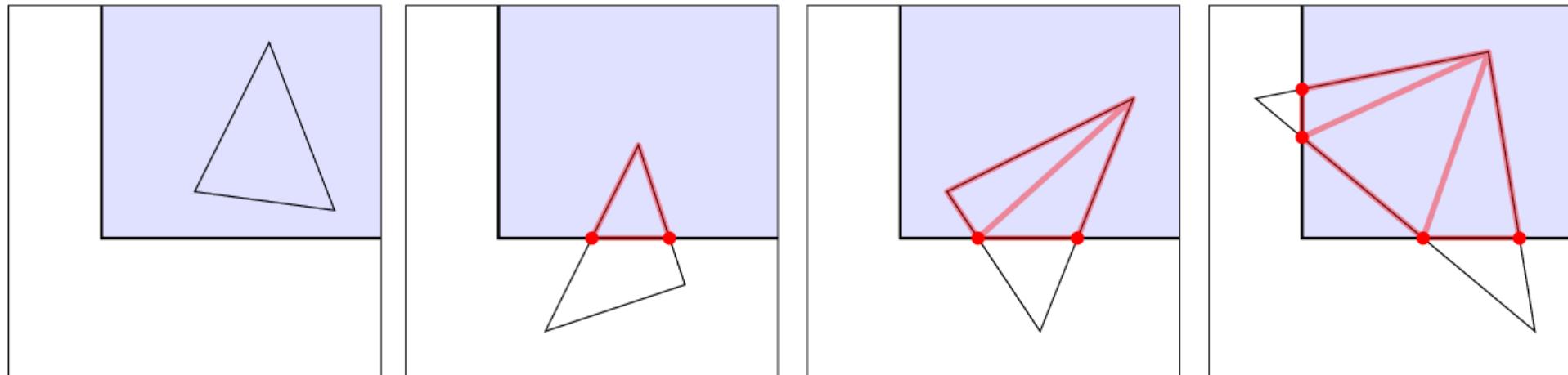


$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{f}{(aspect)} & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & \frac{near + far}{(near - far)} & \frac{2 * far * near}{(near - far)} \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

$f = \frac{1}{\tan(\frac{fov}{2})}$, $aspect = \frac{\text{width}}{\text{height}}$

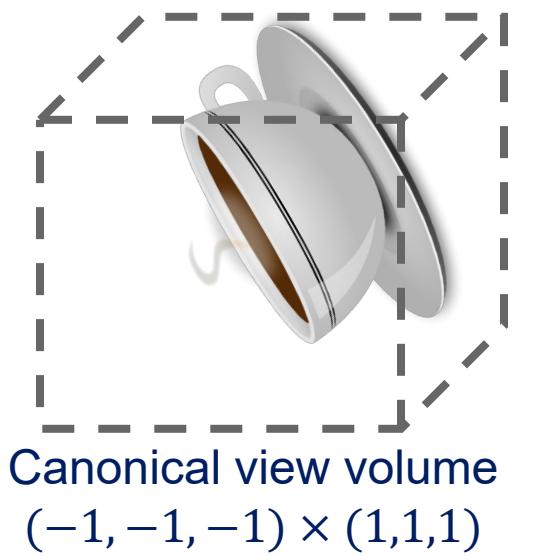
Canonical view volume

- Why?
 - Makes clipping much easier! GL can quickly discard geometry outside $-1, 1$.



From: <https://paroj.github.io/gltut/>

Viewport transformation



Viewport



$$\begin{pmatrix} x_{screen} \\ y_{screen} \\ z_{depth} \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{width}{2} & 0 & 0 & \frac{width}{2} \\ 0 & \frac{height}{2} & 0 & \frac{height}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

`gl.viewport(0, 0, width, height);`



COMPUTER SCIENCE

28

Lab

- Draw a cube on screen with size $(-1, -1, -1) \times (1, 1, 1)$.
 - Apply a model transformation that scales it by 0.5, and tilts it slightly.
 - Apply a projection matrix (orthographic and perspective).