Cameras and Projection

CS 355: Introduction to Graphics and Image Processing

3D Rendering

- If I took a picture of a scene with a virtual camera:
  - What point in 3D is visible at each 2D point in the projected image?
  - What color is the light coming from that point as it reaches the camera?

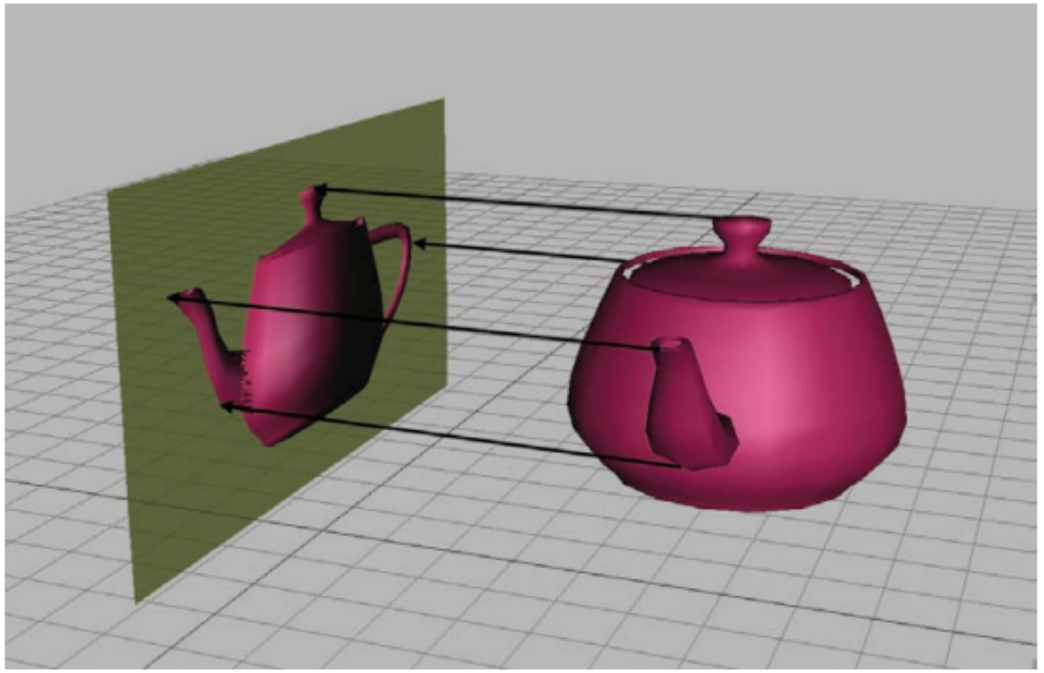
Geometry

Lighting

Projection

- To get 2D pictures of a 3D world, you have to use *projection*
  - Orthographic
  - Perspective

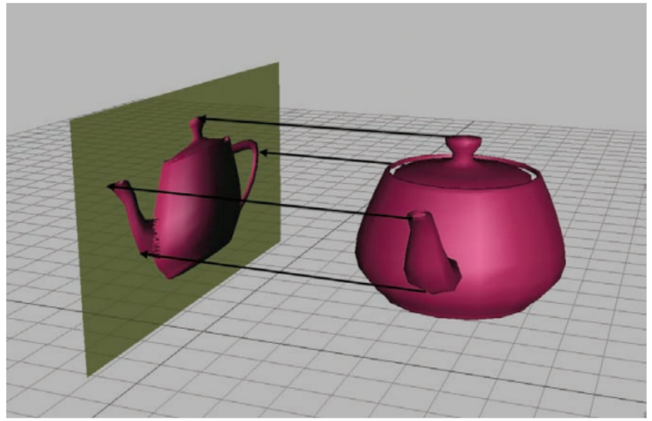
Orthographic Projection



Orthographic projection is just dropping a dimension.

Used in technical drawings, etc.

Orthographic Projection



3D point in homogeneous coordinates

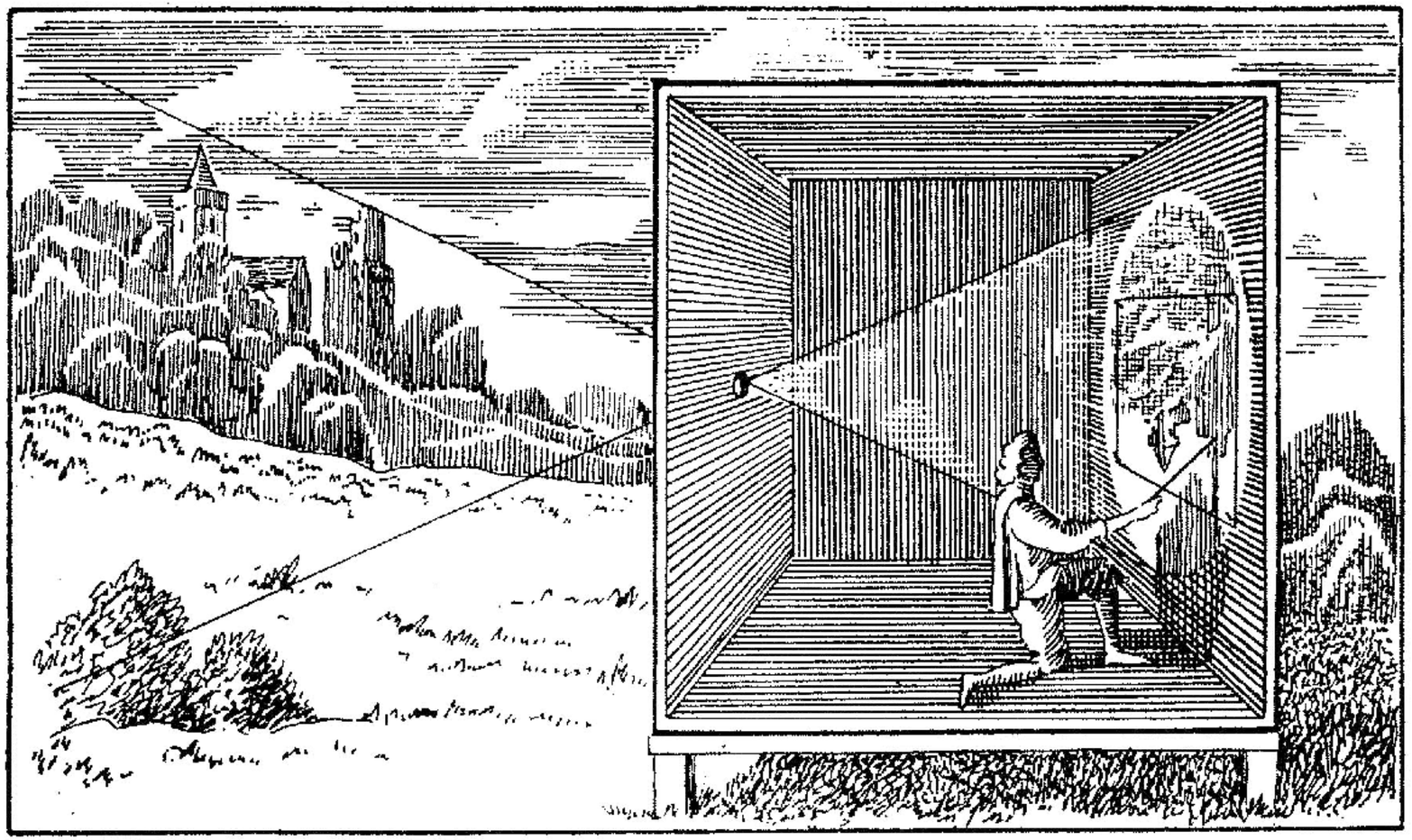
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Orthographic projection is just dropping a dimension.

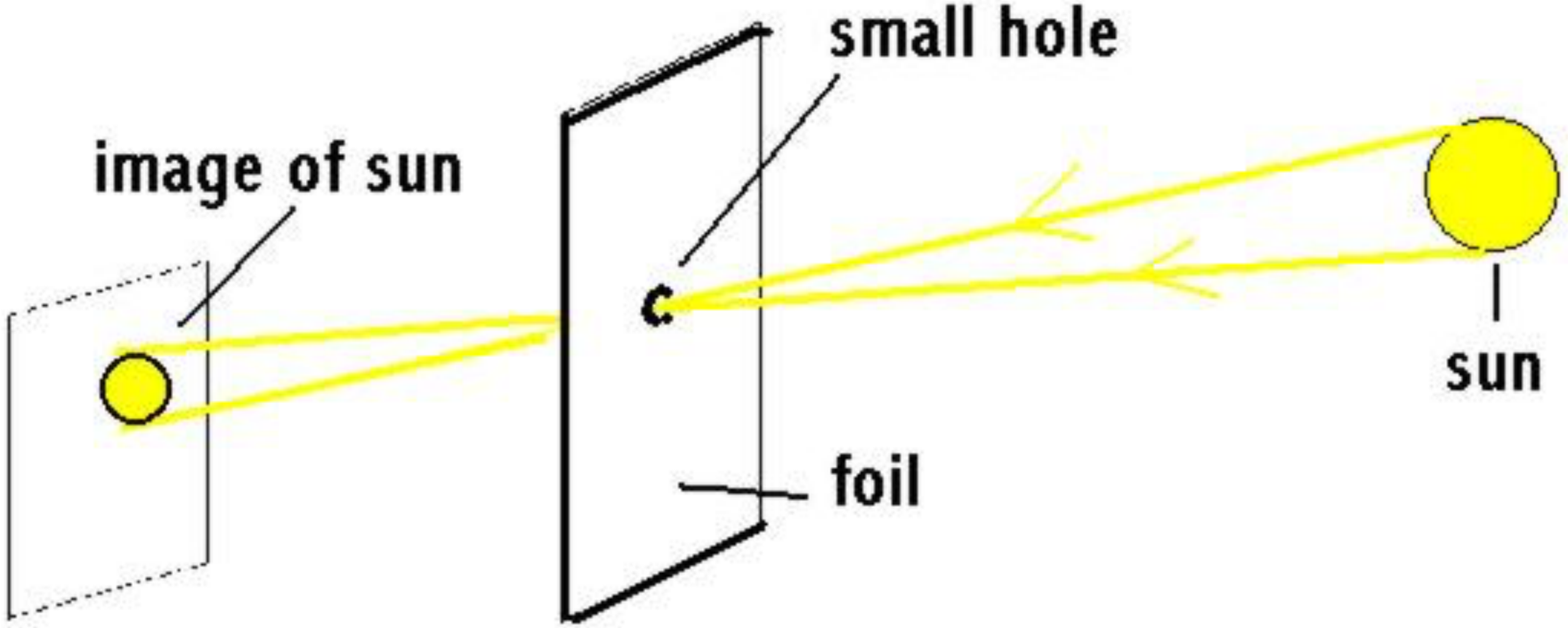
Lacking Perspective

Perspective Projection

Camera Obscura

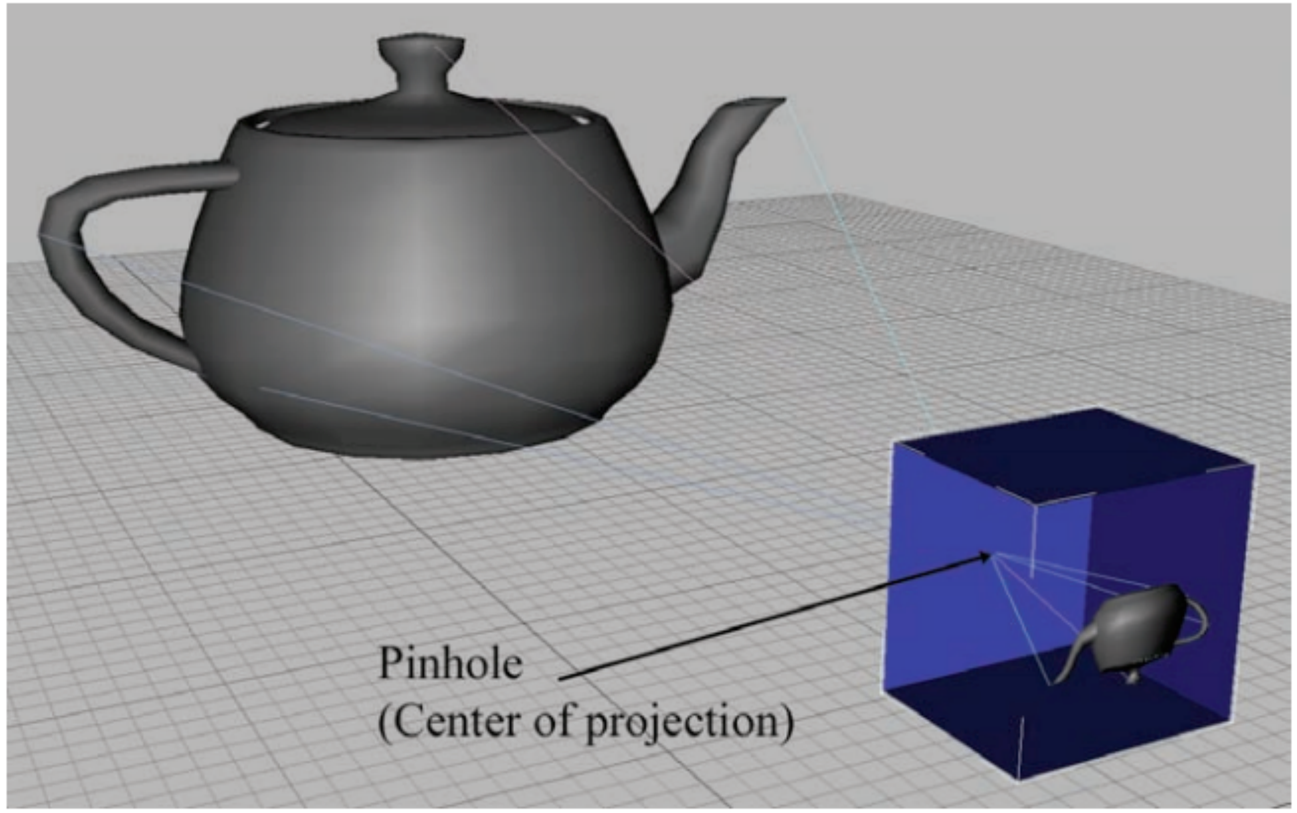


Eclipse Viewing



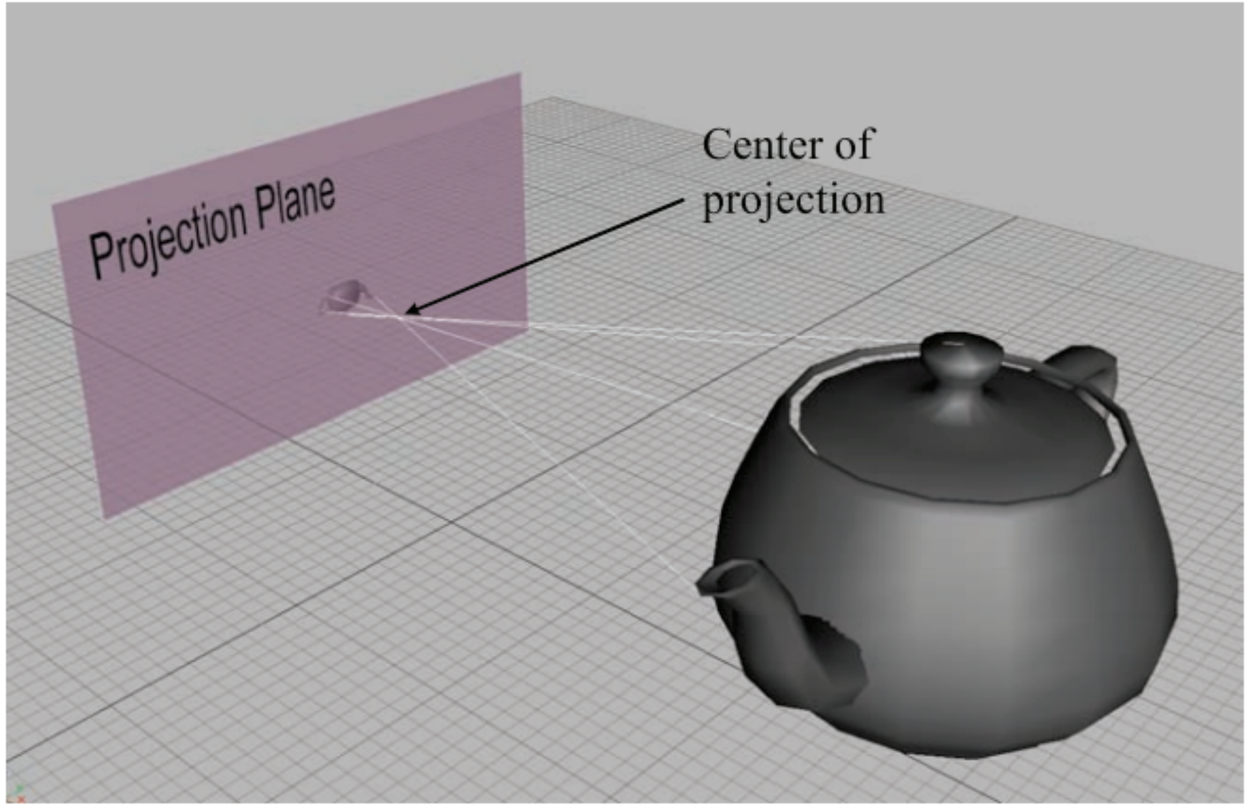
Eclipse Viewing

Perspective Projection



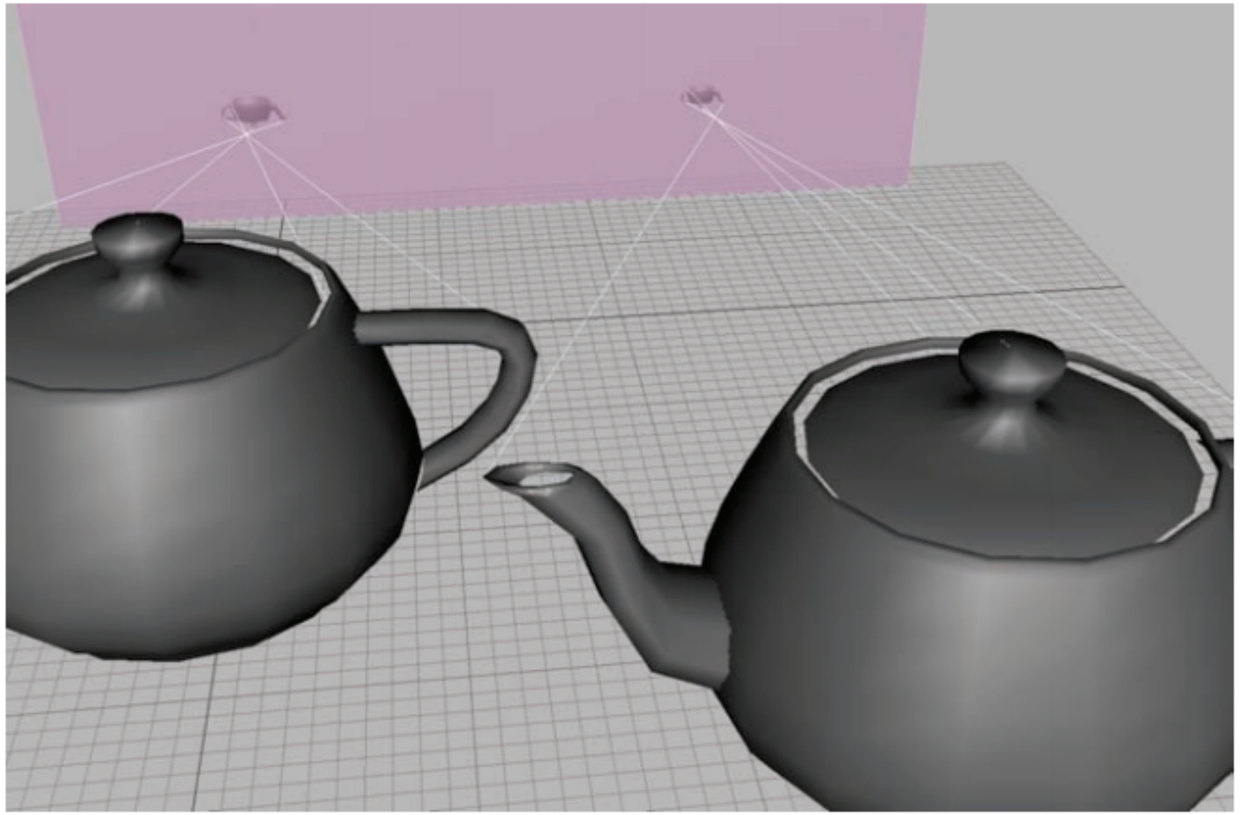
Many graphics systems assume a simple pinhole camera model

Perspective Projection

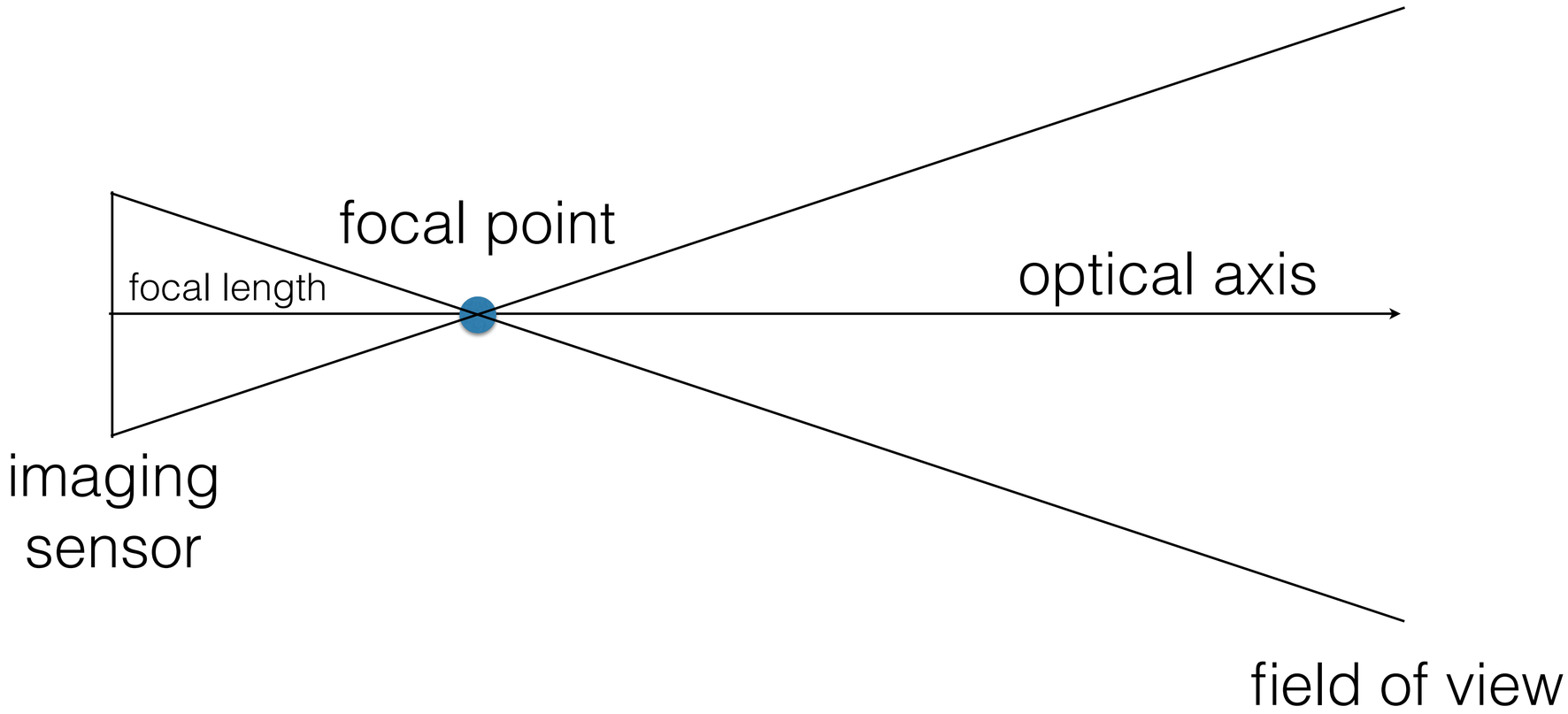


Pinhole camera model

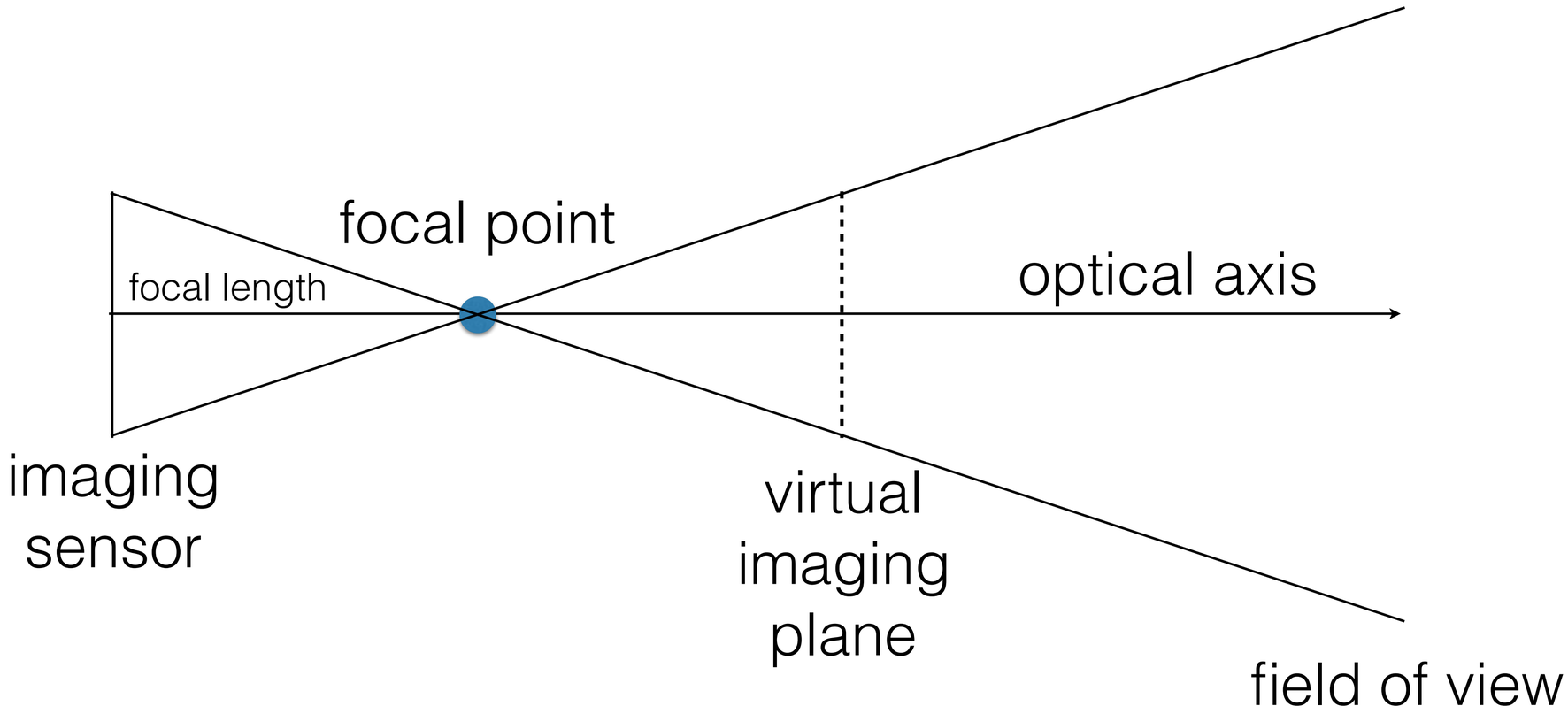
Perspective Projection



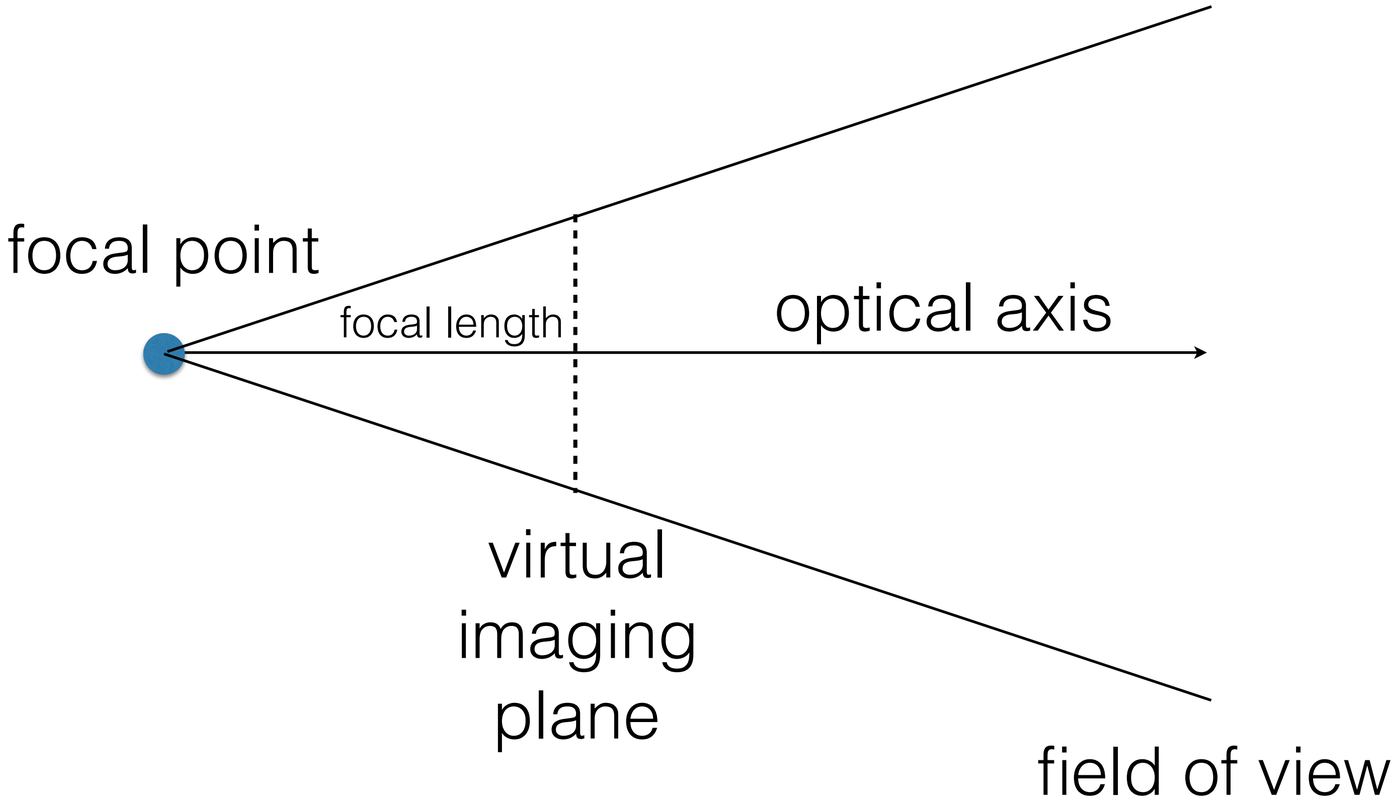
Pinhole Cameras



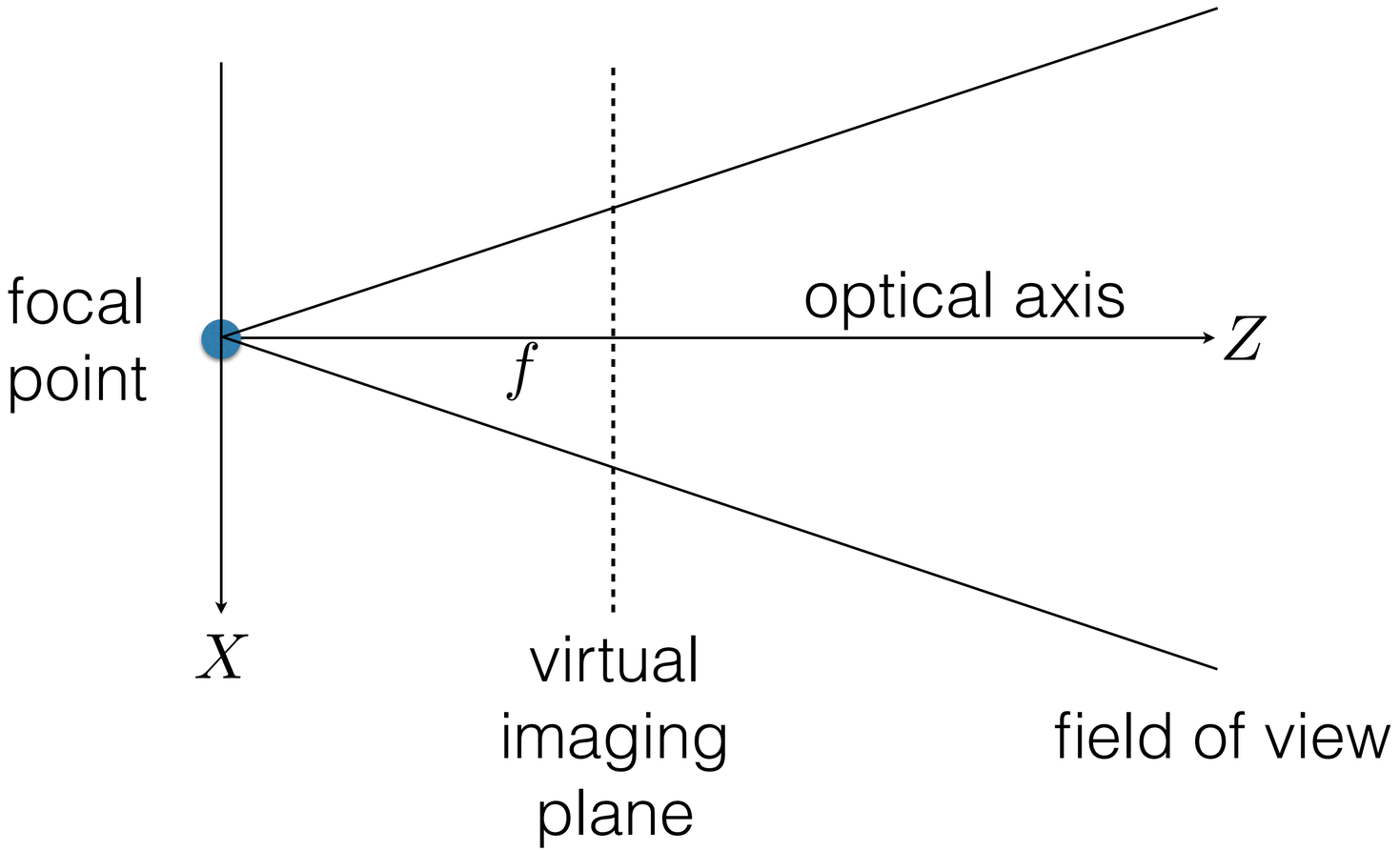
Geometric Model



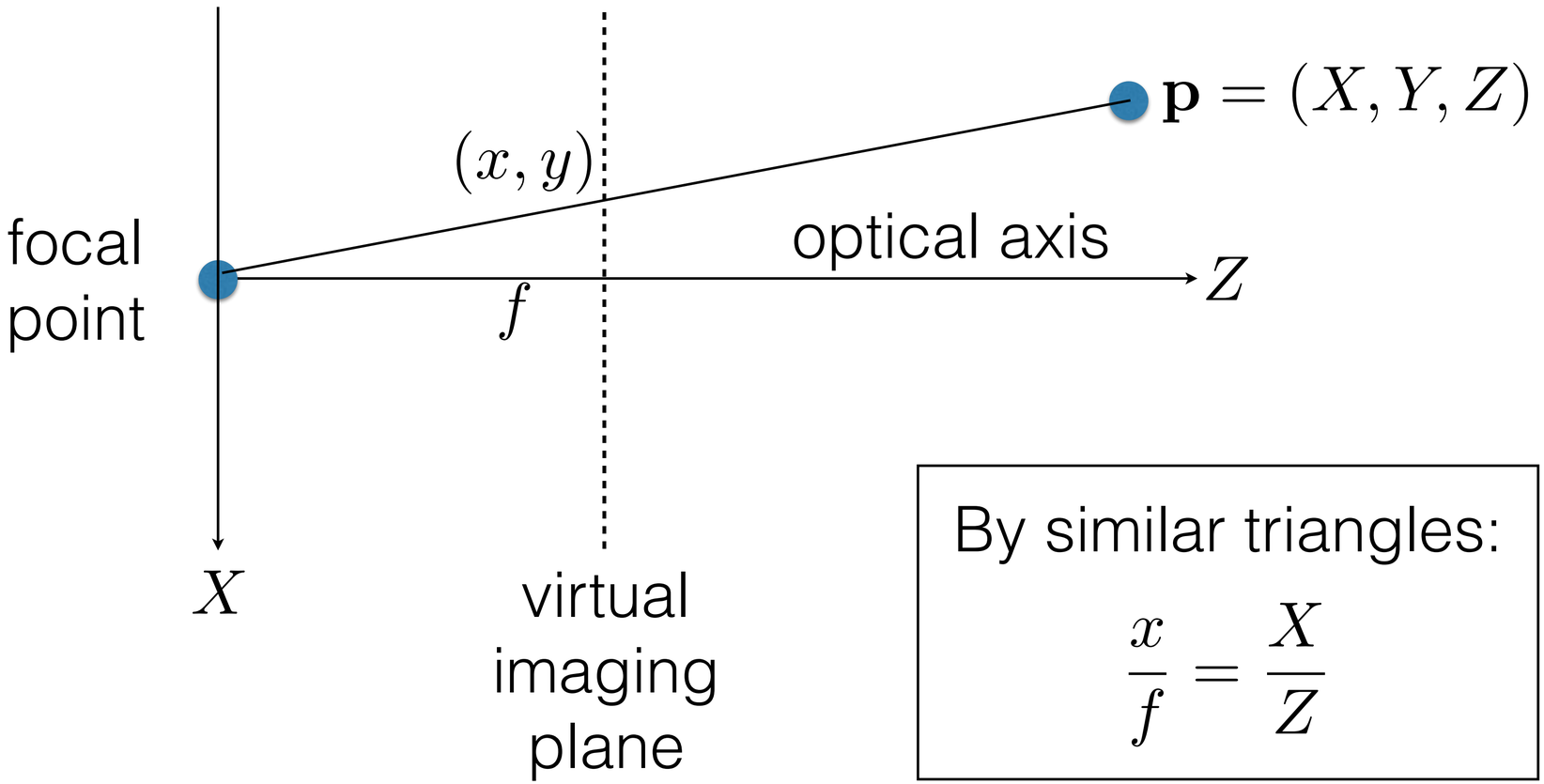
Geometric Model



Camera Coordinates



Projection



By similar triangles:

$$\frac{x}{f} = \frac{X}{Z}$$

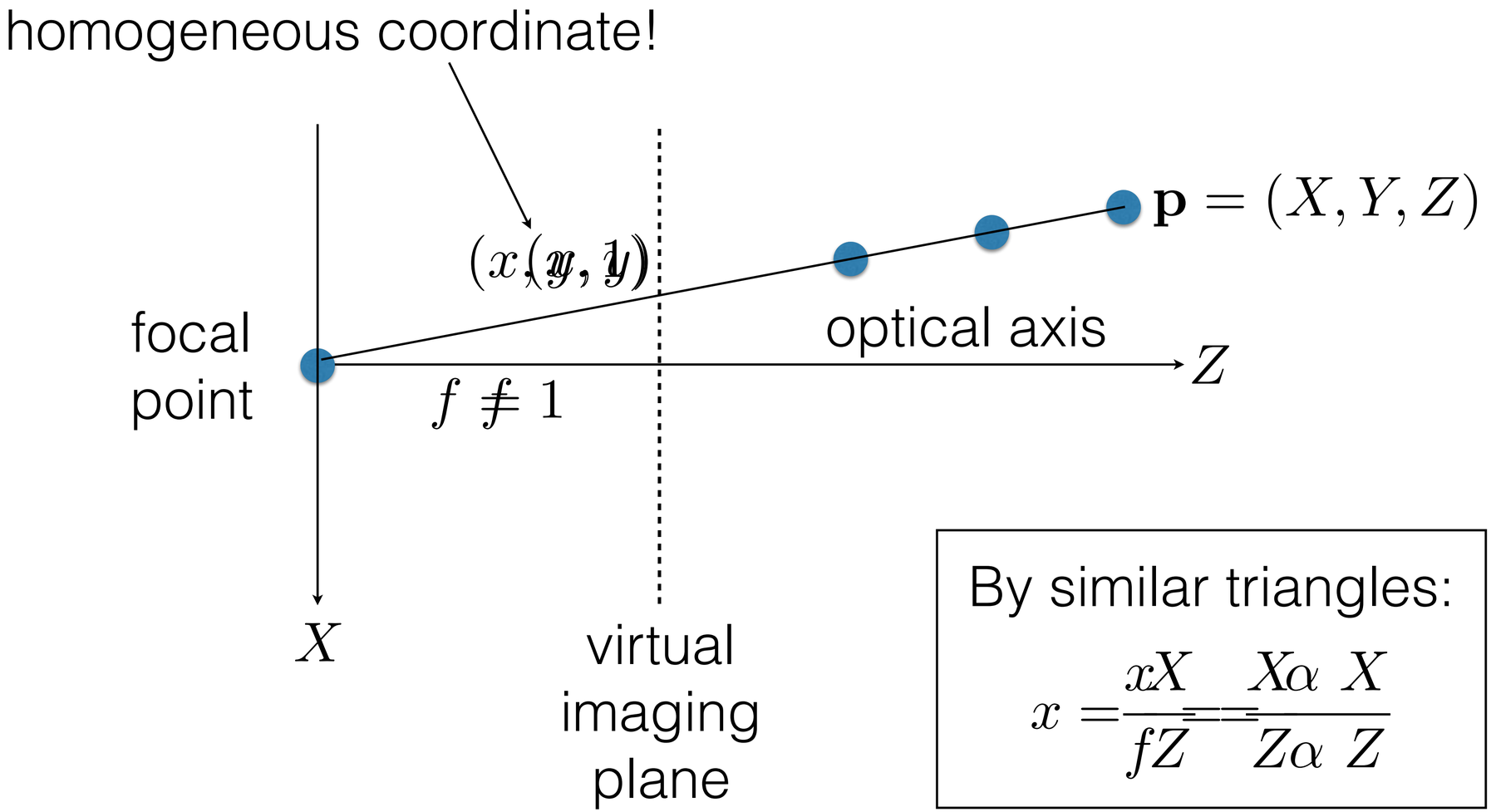
Projection

$$\frac{x}{f} = \frac{X}{Z} \qquad \frac{y}{f} = \frac{Y}{Z}$$

$$(x, y) = \left(\frac{fX}{Z}, \frac{fY}{Z} \right)$$

Note: this is the projected coordinate in real-world units.
To get actual pixel location, have to scale by pixel density and apply offset to image origin (more on this later...)

Projection



By similar triangles:

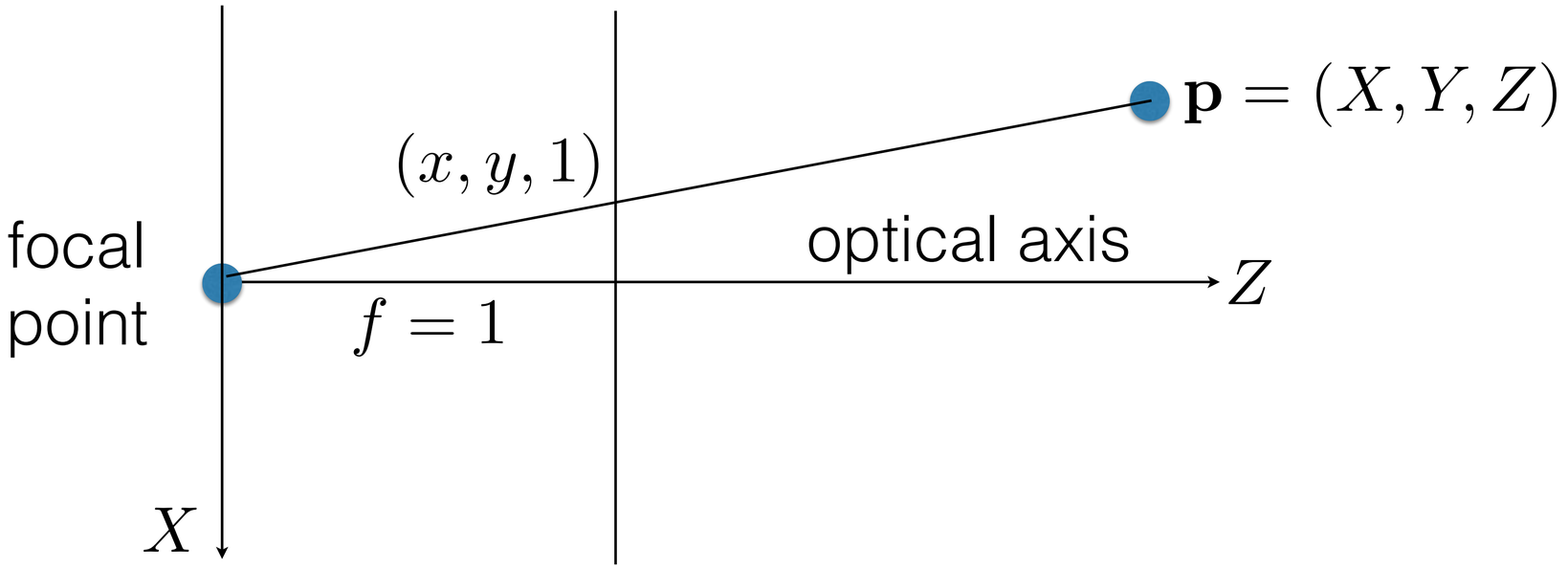
$$x = \frac{xX}{fZ} = \frac{X\alpha}{Z\alpha} = \frac{X}{Z}$$

Homogenous Coordinates

- Homogeneous coordinates are used to represent all 3D points along the ray that falls on the same 2D projection:

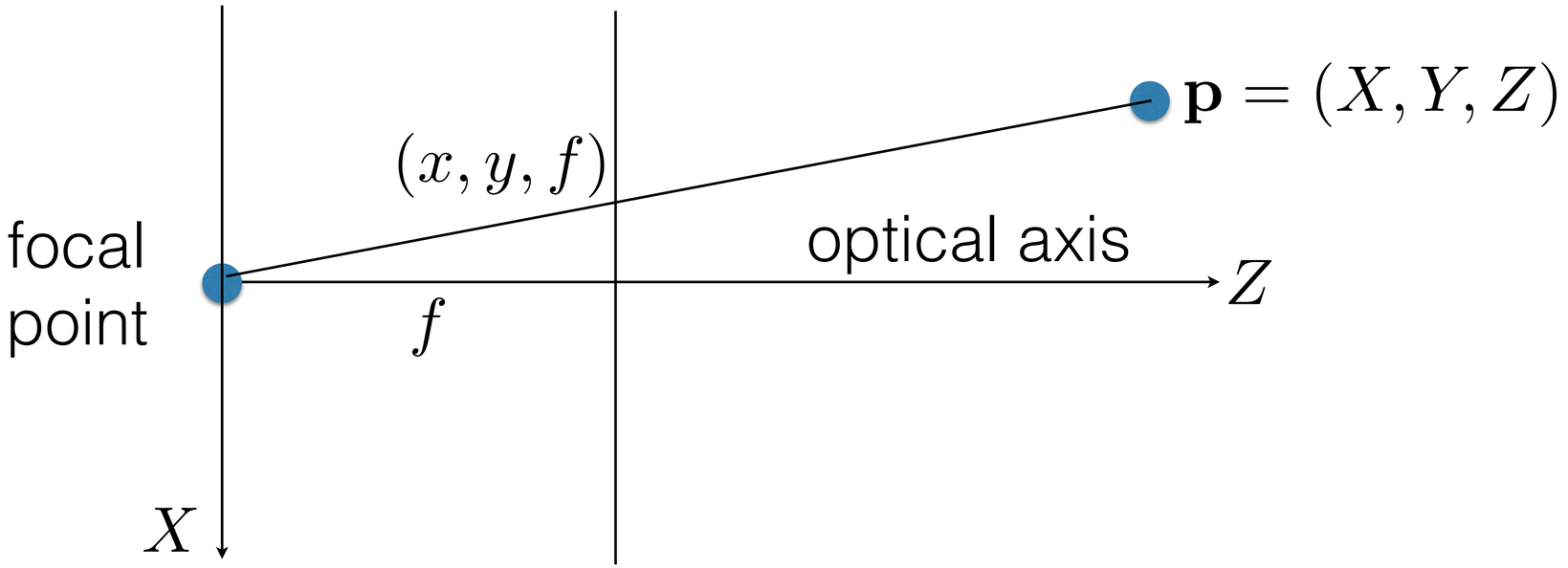
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} \alpha\ x \\ \alpha\ y \\ \alpha \end{bmatrix}$$

Perspective Projection



$$\begin{bmatrix} \boxed{\begin{matrix} x \\ y \end{matrix}} \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} X/Z \\ Y/Z \\ 1 \\ 1 \end{bmatrix} \sim \begin{bmatrix} X \\ Y \\ Z \\ Z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \boxed{\begin{matrix} X \\ Y \\ Z \end{matrix}} \\ 1 \end{bmatrix}$$

Perspective Projection



$$\begin{bmatrix} \boxed{x} \\ \boxed{y} \\ f \\ 1 \end{bmatrix} = \begin{bmatrix} fX/Z \\ fY/Z \\ f \\ 1 \end{bmatrix} \sim \begin{bmatrix} X \\ Y \\ Z \\ Z/f \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \boxed{1/f} & 0 \end{bmatrix} \begin{bmatrix} \boxed{X} \\ \boxed{Y} \\ \boxed{Z} \\ \boxed{1} \end{bmatrix}$$

Alternative Form

One way (some implementation advantages):

$$\begin{bmatrix} x \\ y \\ f \\ 1 \end{bmatrix} = \begin{bmatrix} fX/Z \\ fY/Z \\ f \\ 1 \end{bmatrix} \sim \begin{bmatrix} X \\ Y \\ Z \\ Z/f \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Another way (some conceptual advantages):

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} fX/Z \\ fY/Z \\ 1 \end{bmatrix} \sim \begin{bmatrix} X \\ Y \\ Z/f \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Coming up...

- Simple modeling primitives (points, lines, polygons)
- 3D rendering geometry
- Introduction to OpenGL
- Hierarchical transformations
- Visibility
- Lighting