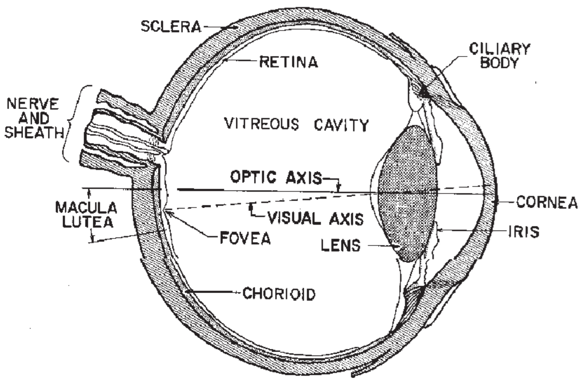
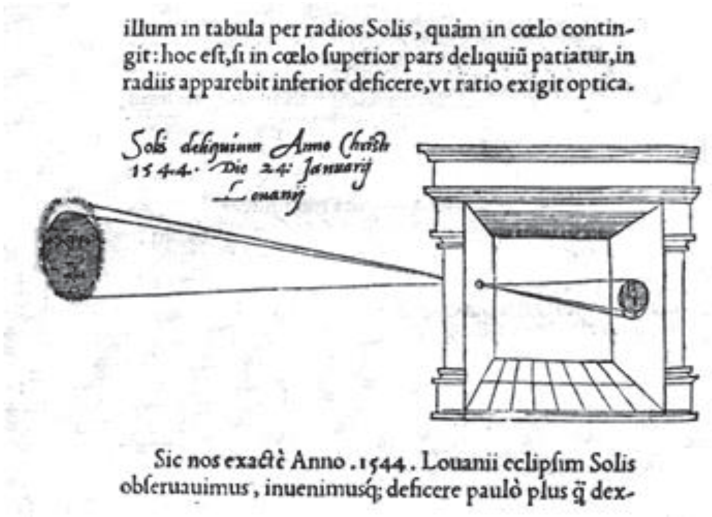
2.2

Geometric Image Formation

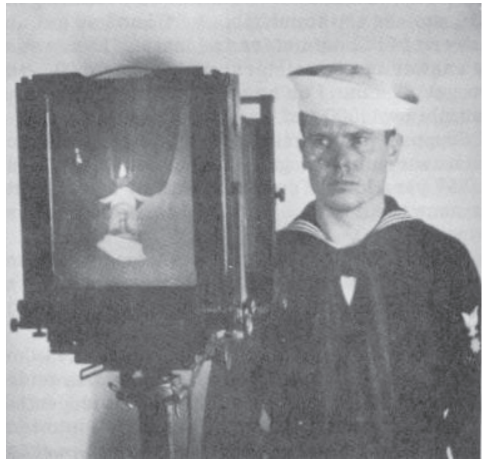
Origins of the Pinhole Camera



Animal Eye:
A long time ago

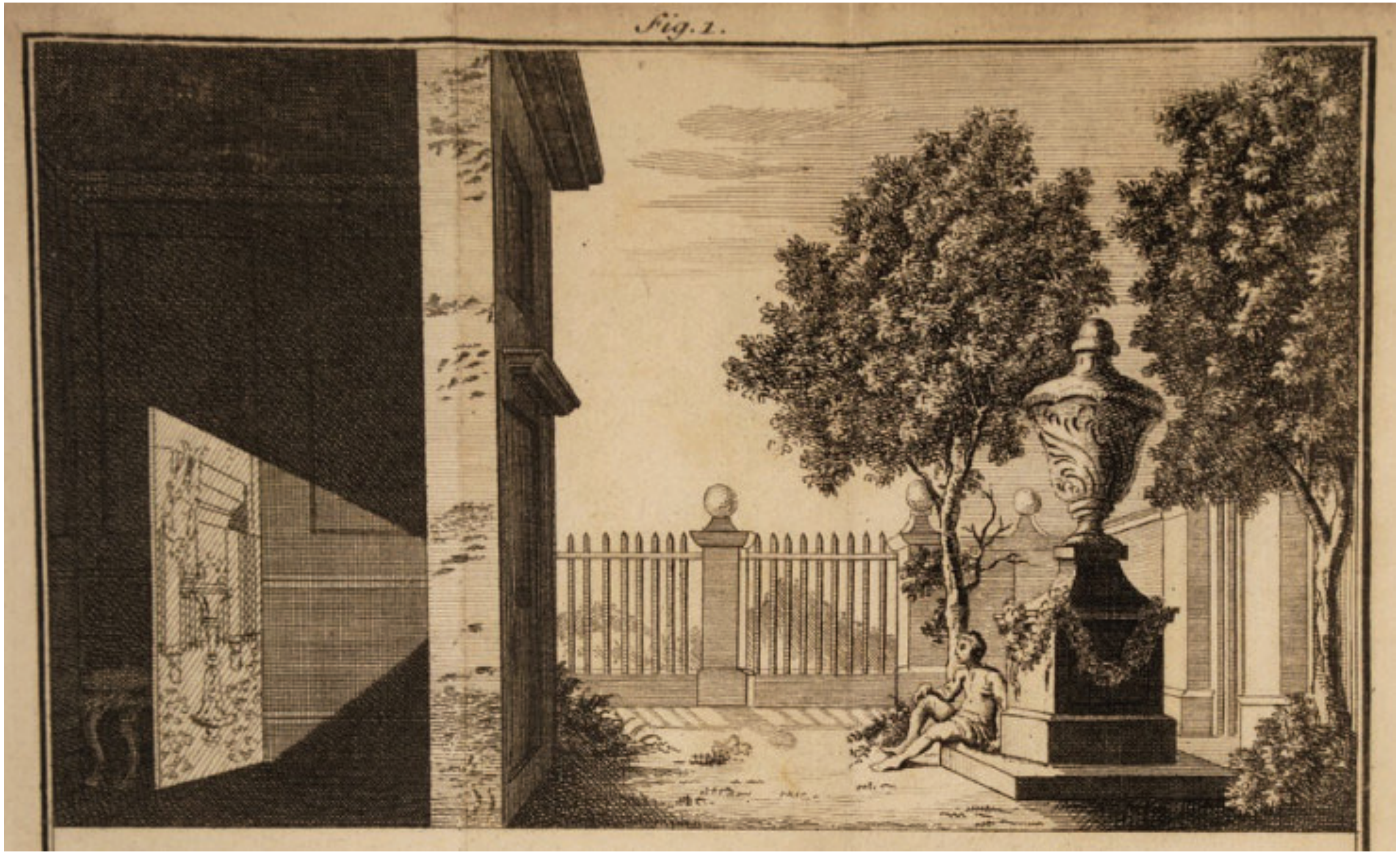


Pinhole Perspective Projection:
Brunelleschi, 15th Century



Photographic Camera:
Nicéphore Niépce, 1816

Origins of the Pinhole Camera



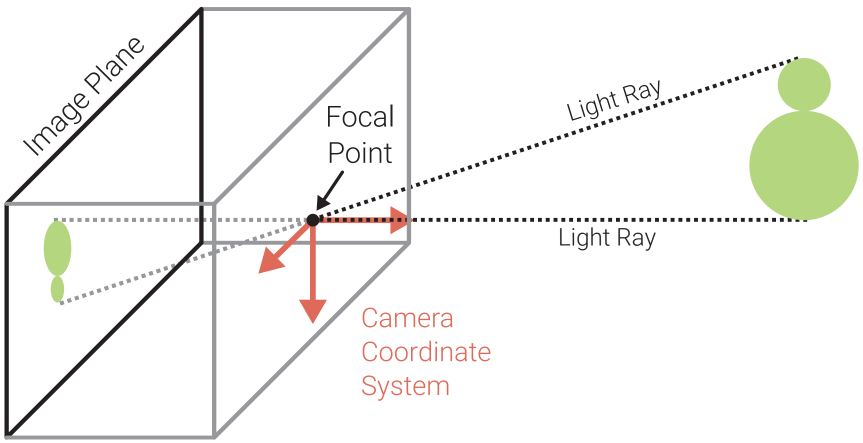
Camera Obscura:
4th Century BC

Origins of the Pinhole Camera

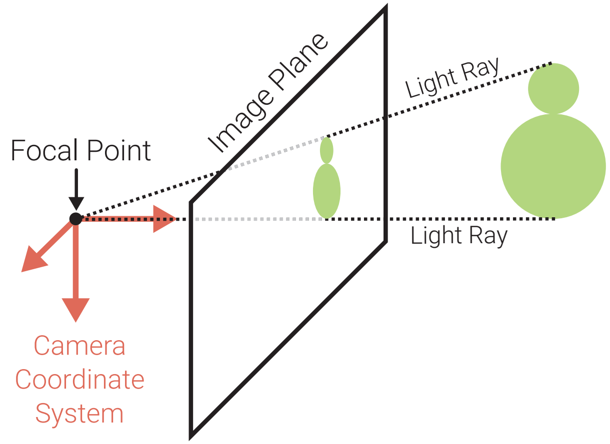
https://www.abelardomorell.net/camera-obscura

Origins of the Pinhole Camera

Physical Camera Model



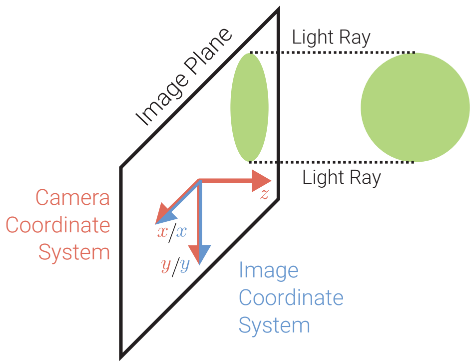
Mathematical Camera Model

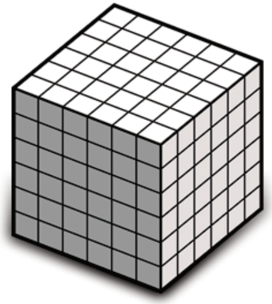


- In a physical pinhole camera the image is projected up-side down onto the image plane which is located **behind** the focal point
- When modeling perspective projection, we assume the image plane **in front**
- Both models are **equivalent,** with appropriate change of image coordinates

Projection Models

Orthographic Projection

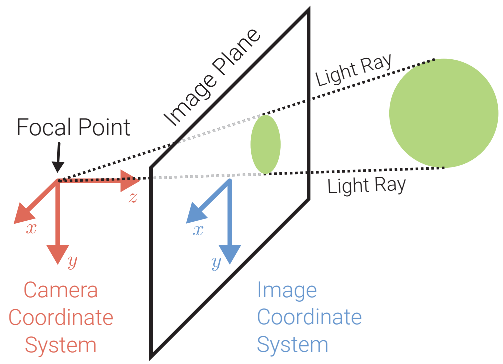


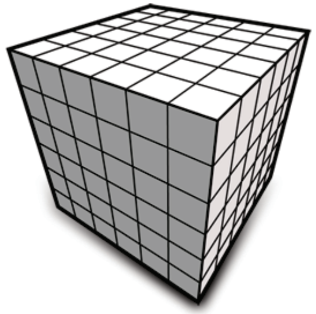


Opto Engineering Telecentric Lens

Canon 800mm Telephoto Lens

Perspective Projection





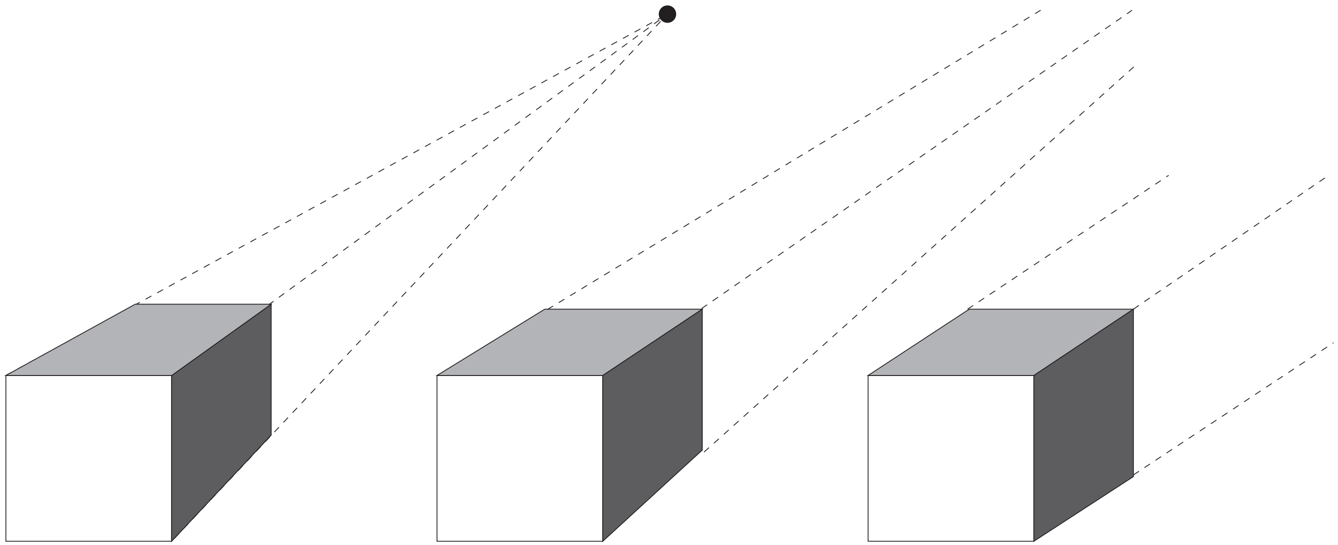
Nikon AF-S Nikkor 50mm

Sony DSC-RX100 V

Samsung Galaxy S20

- These two are the most important projections, see Szeliski Ch. 2.1.4 for others

Projection Models

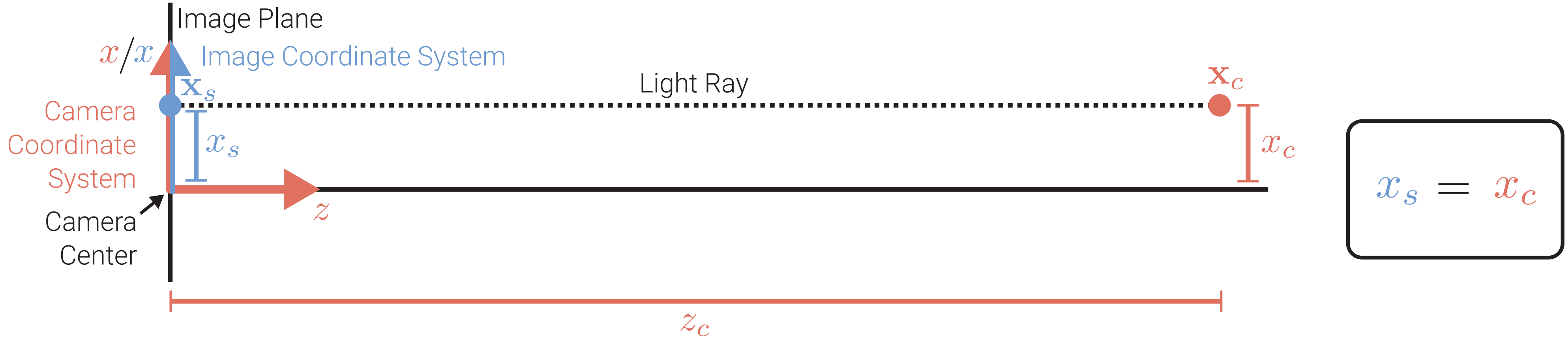


Perspective

Weak Perspective /Orthographic

Increasing Focal Length / Distance from Camera

Orthographic Projection



Orthographic projection of a 3D point $\mathbf{x}_c \in \mathbb{R}^3$ to pixel coordinates $\mathbf{x}_s \in \mathbb{R}^2$:

- The x and y axes of the camera and image coordinate systems are shared
- Light rays are parallel to the z-coordinate of the camera coordinate system
- During projection, the z-coordinate is dropped, x and y remain the same
- Remark: the y coordinate is not shown here for clarity, but behaves similarly

Orthographic Projection

An **orthographic projection** simply **drops the z component** of the 3D point in camera coordinates $\mathbf{x}_c$ to obtain the corresponding 2D point on the image plane (= screen) $\mathbf{x}_s$.

$$\mathbf{x}_s = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{x}_c \quad \Leftrightarrow \quad \bar{\mathbf{x}}_s = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \bar{\mathbf{x}}_c$$

Orthography is exact for telecentric lenses and an approximation for telephoto lenses. After projection the distance of the 3D point from the image can't be recovered.

Scaled Orthographic Projection

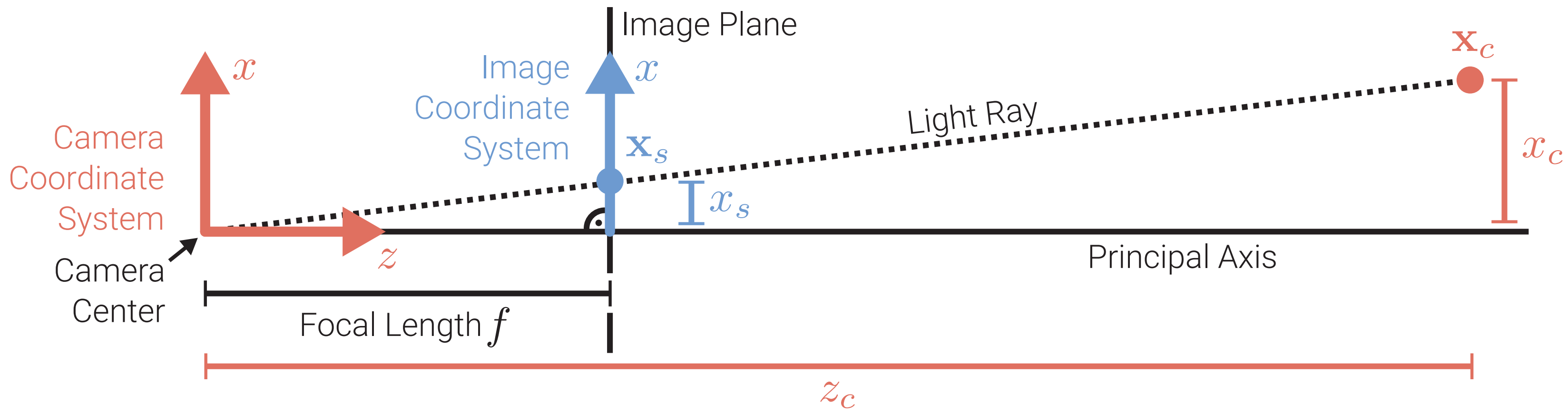
In practice, world coordinates (which may measure dimensions in meters) must be scaled to fit onto an image sensor (measuring in pixels) $\Rightarrow$ **scaled orthography:**

$$\mathbf{x}_s = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \end{bmatrix} \mathbf{x}_c \quad \Leftrightarrow \quad \bar{\mathbf{x}}_s = \begin{bmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \bar{\mathbf{x}}_c$$

Remark: The unit for s is px/m or px/mm to convert metric 3D points into pixels.

Under orthography, structure and motion can be estimated simultaneously using factorization methods (e.g., via singular value decomposition).

Perspective Projection



$$\frac{x_s}{f} = \frac{x_c}{z_c}$$

Perspective projection of a 3D point $\mathbf{x}_c \in \mathbb{R}^3$ to pixel coordinates $\mathbf{x}_s \in \mathbb{R}^2$:

- The light ray passes through the camera center, the pixel $\mathbf{x}_s$ and the point $\mathbf{x}_c$
- Convention: the principal axis (orthogonal to image plane) aligns with the z-axis
- Remark: the y coordinate is not shown here for clarity, but behaves similarly

Perspective Projection

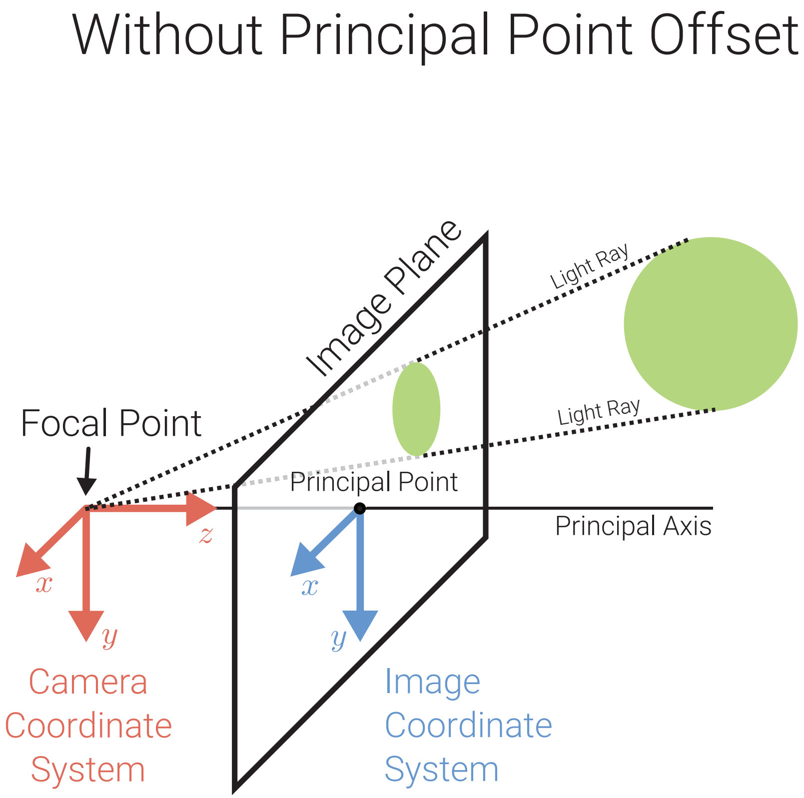
In **perspective projection,** 3D points in camera coordinates are mapped to the image plane by **dividing** them **by their z component** and multiplying with the focal length:

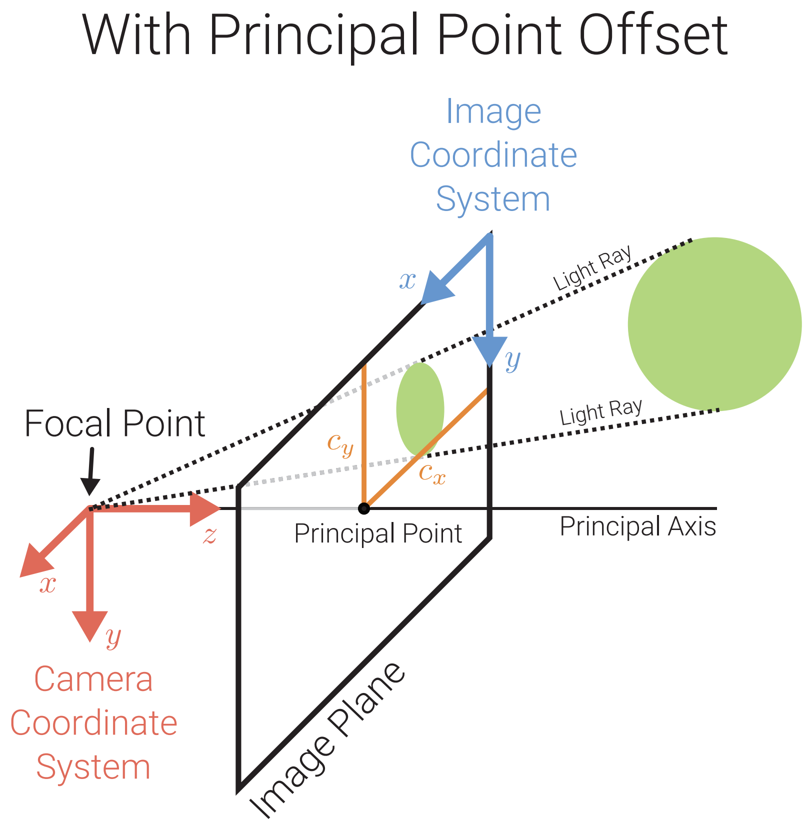
$$\begin{pmatrix} x_s \\ y_s \end{pmatrix} = \begin{pmatrix} f x_c / z_c \\ f y_c / z_c \end{pmatrix} \quad \Leftrightarrow \quad \tilde{\mathbf{x}}_s = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \bar{\mathbf{x}}_c$$

Note that this projection is **linear** when using **homogeneous coordinates.** After the projection it is not possible to recover the distance of the 3D point from the image.

Remark: The unit for f is px (=pixels) to convert metric 3D points into pixels.

Perspective Projection





- To ensure positive pixel coordinates, a **principal point offset $\mathbf{c}$** is usually added
- This moves the image coordinate system to the corner of the image plane

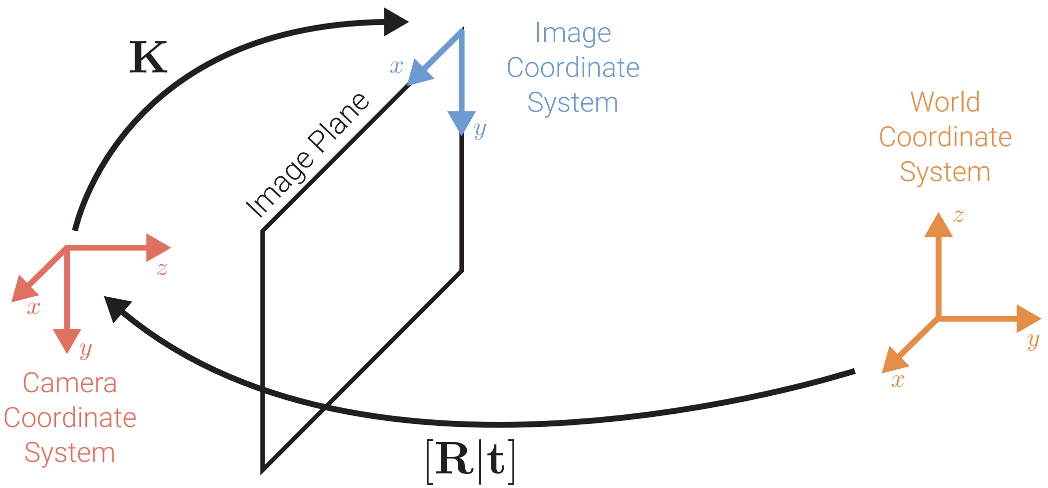
Perspective Projection

The **complete perspective projection model** is given by:

$$\begin{pmatrix} x_s \\ y_s \end{pmatrix} = \begin{pmatrix} f_x\, x_c/z_c + s\, y_c/z_c + c_x \\ f_y\, y_c/z_c + c_y \end{pmatrix} \quad \Leftrightarrow \quad \tilde{\mathbf{x}}_s = \begin{bmatrix} f_x & s & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \bar{\mathbf{x}}_c$$

- The left 3×3 submatrix of the projection matrix is called **calibration matrix** $\mathbf{K}$
- The parameters of $\mathbf{K}$ are called camera intrinsics (as opposed to extrinsic pose)
- Here, f_x and f_y are independent, allowing for different pixel aspect ratios
- The skew s arises due to the sensor not mounted perpendicular to the optical axis
- In practice, we often set $f_x = f_y$ and $s = 0$, but model $\mathbf{c} = (c_x, c_y)^\top$

Chaining Transformations



Let $\mathbf{K}$ be the calibration matrix (intrinsics) and $[\mathbf{R}|\mathbf{t}]$ the camera pose (extrinsics).
We **chain both transformations** to project a point in world coordinates to the image:

$$\tilde{\mathbf{x}}_s = \begin{bmatrix} \mathbf{K} & \mathbf{0} \end{bmatrix} \bar{\mathbf{x}}_c = \begin{bmatrix} \mathbf{K} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \bar{\mathbf{x}}_w = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \bar{\mathbf{x}}_w = \mathbf{P}\, \bar{\mathbf{x}}_w$$

Remark: The 3×4 projection matrix $\mathbf{P}$ can be pre-computed.

Full Rank Representation

It is sometimes preferable to use a **full rank** 4×4 projection matrix:

$$\tilde{\mathbf{x}}_s = \begin{bmatrix} \mathbf{K} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \bar{\mathbf{x}}_w = \tilde{\mathbf{P}}\,\bar{\mathbf{x}}_w$$

Now, the homogeneous vector $\tilde{\mathbf{x}}_s$ is a 4D vector and must be normalized wrt. its 3rd entry to obtain inhomogeneous image pixels:

$$\bar{\mathbf{x}}_s = \tilde{\mathbf{x}}_s / z_s = (x_s/z_s, y_s/z_s, 1, 1/z_s)^\top$$

Note that the 4th component of the inhomogeneous 4D vector is the **inverse depth.** If the inverse depth is known, a 3D point can be retrieved from its pixel coordinates via $\tilde{\mathbf{x}}_w = \tilde{\mathbf{P}}^{-1}\,\bar{\mathbf{x}}_s$ and subsequent normalization of $\tilde{\mathbf{x}}_w$ wrt. its 4th entry.

Lens Distortion

The assumption of linear projection (straight lines remain straight) is violated in practice due to the properties of the camera lens which introduces distortions.

Both **radial and tangential distortion** effects can be modeled relatively easily:

Let $x = x_c/z_c$, $y = y_c/z_c$ and $r^2 = x^2 + y^2$. The distorted point is obtained as:

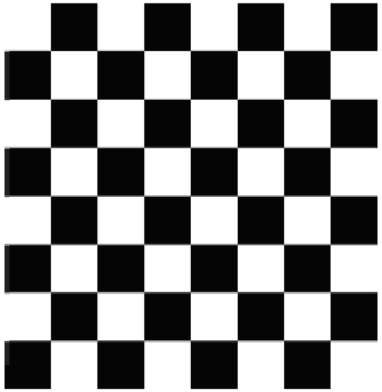
$$\mathbf{x}' = \underbrace{(1 + \kappa_1 r^2 + \kappa_2 r^4)}_{\text{Radial Distortion}} \begin{pmatrix} x \\ y \end{pmatrix} + \underbrace{\begin{pmatrix} 2\,\kappa_3\,x\,y + \kappa_4(r^2 + 2\,x^2) \\ 2\,\kappa_4\,x\,y + \kappa_3(r^2 + 2\,y^2) \end{pmatrix}}_{\text{Tangential Distortion}}$$

$$\mathbf{x}_s = \begin{pmatrix} f_x\,x' + c_x \\ f_y\,y' + c_y \end{pmatrix}$$

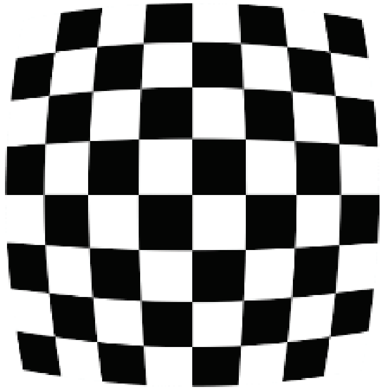
Images can be **undistorted** such that the perspective projection model applies.

More complex distortion models must be used for wide-angle lenses (e.g., fisheye).

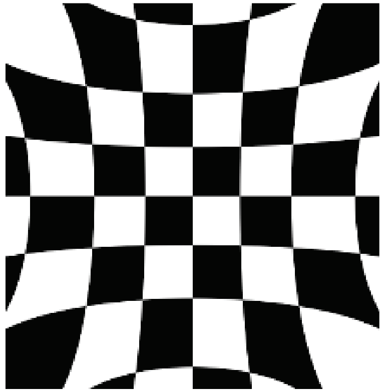
Lens Distortion



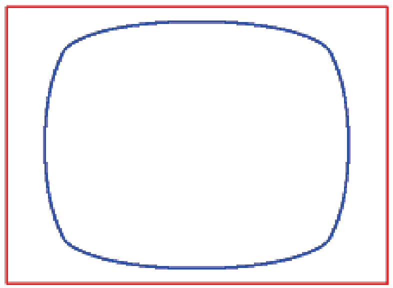
No distortion



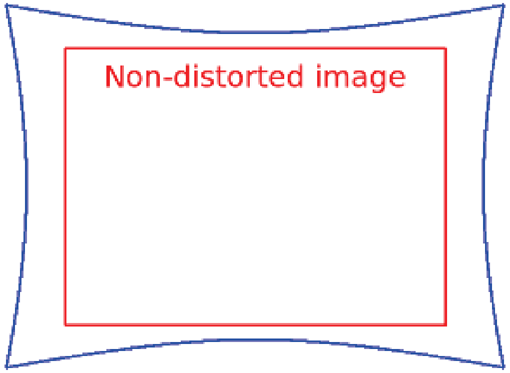
Negative radial distortion
(Barrel distortion)



Positive radial distortion
(Pincushion distortion)



Negative radial distortion (k1=-1.5)
(Barrel distortion)



Positive radial distortion (k1=1.5)
(Pincushion distortion)