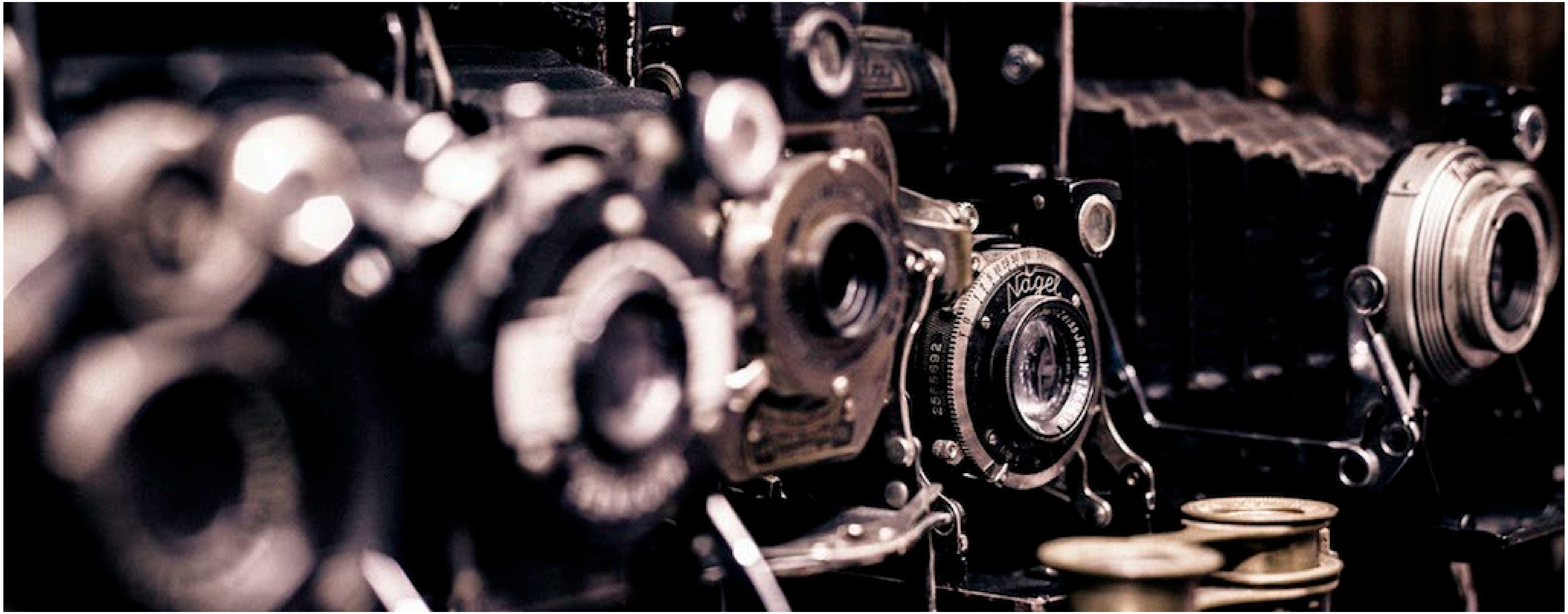
Geometric camera models and calibration



15-463, 15-663, 15-862
Computational Photography
Fall 2022, Lecture 15

http://graphics.cs.cmu.edu/courses/15-463

Course announcements

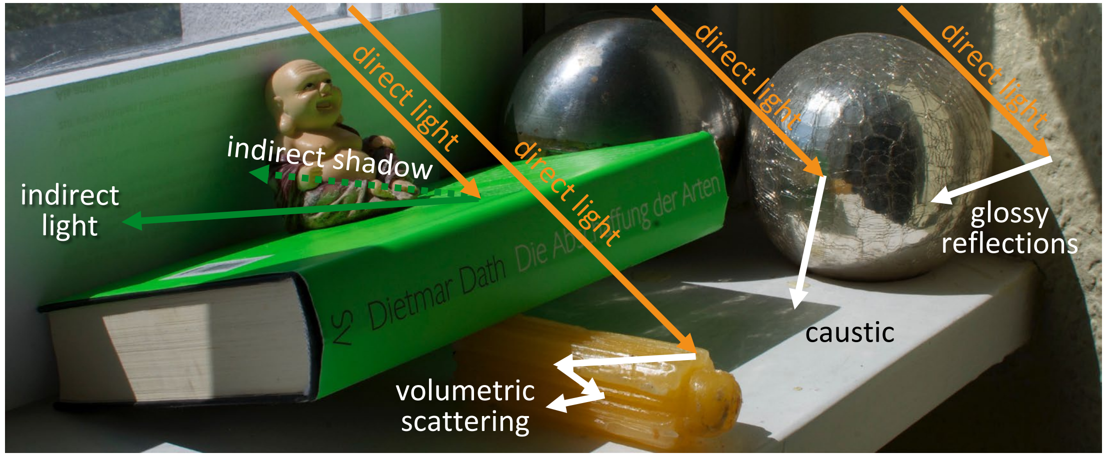
- Homework 5 is due **November 21st**.
  - Any questions?

- Reading group this Friday, 3 – 4:30 pm.

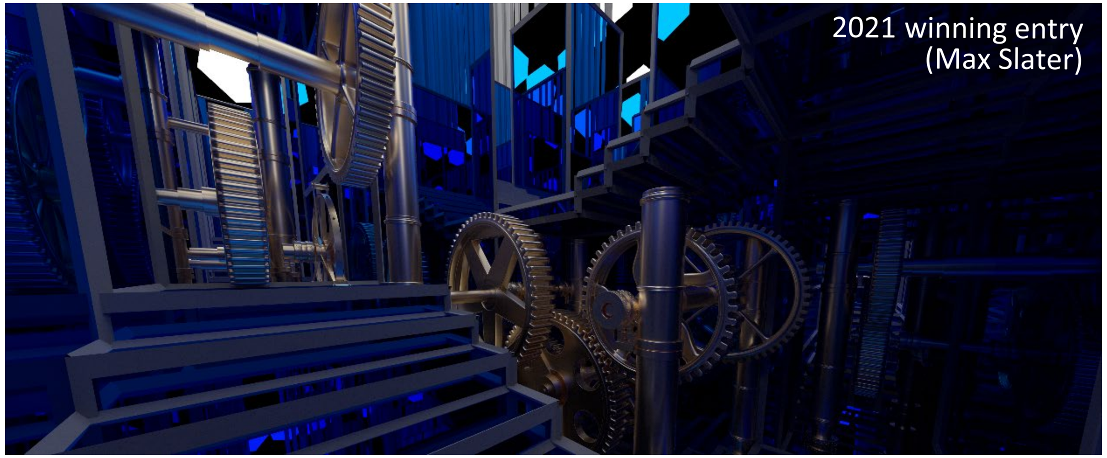
- Extra office hours by Yannis this Friday, 4:30 – 6:30 pm.

15-468/15-668/15-868 Physics-based Rendering

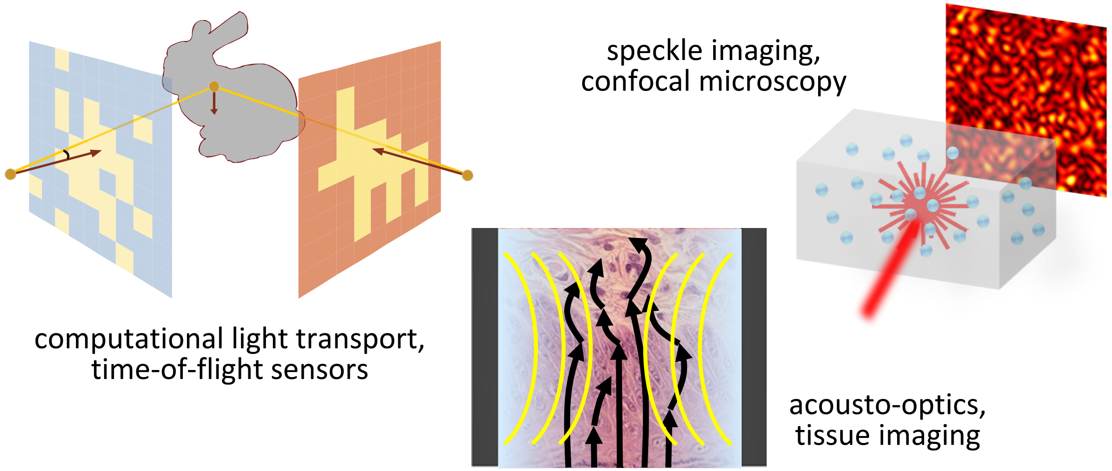
Learn all about modeling, simulating, differentiating, and inverting light!



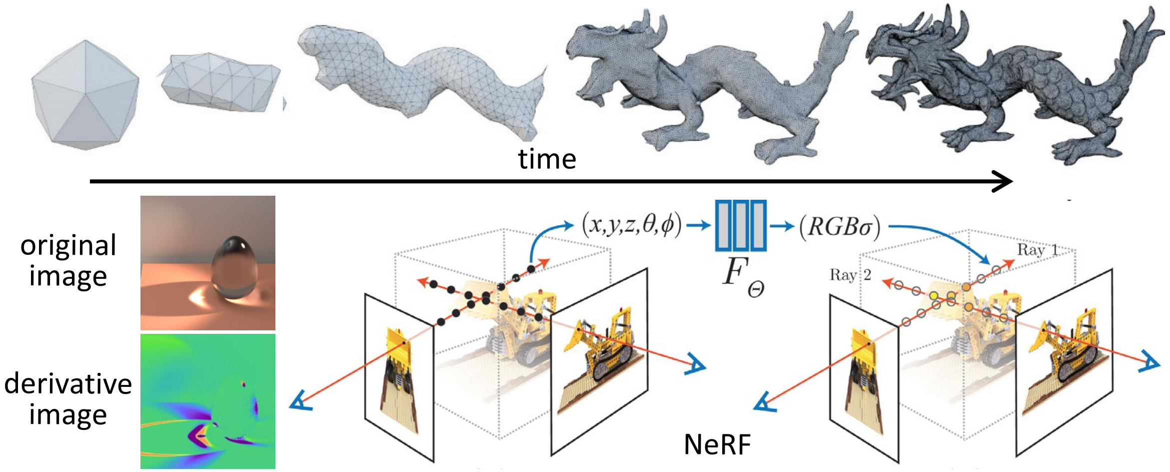
theory and simulation of light transport



rendering competition (win free SIGGRAPH registrations!)



scientific imaging applications



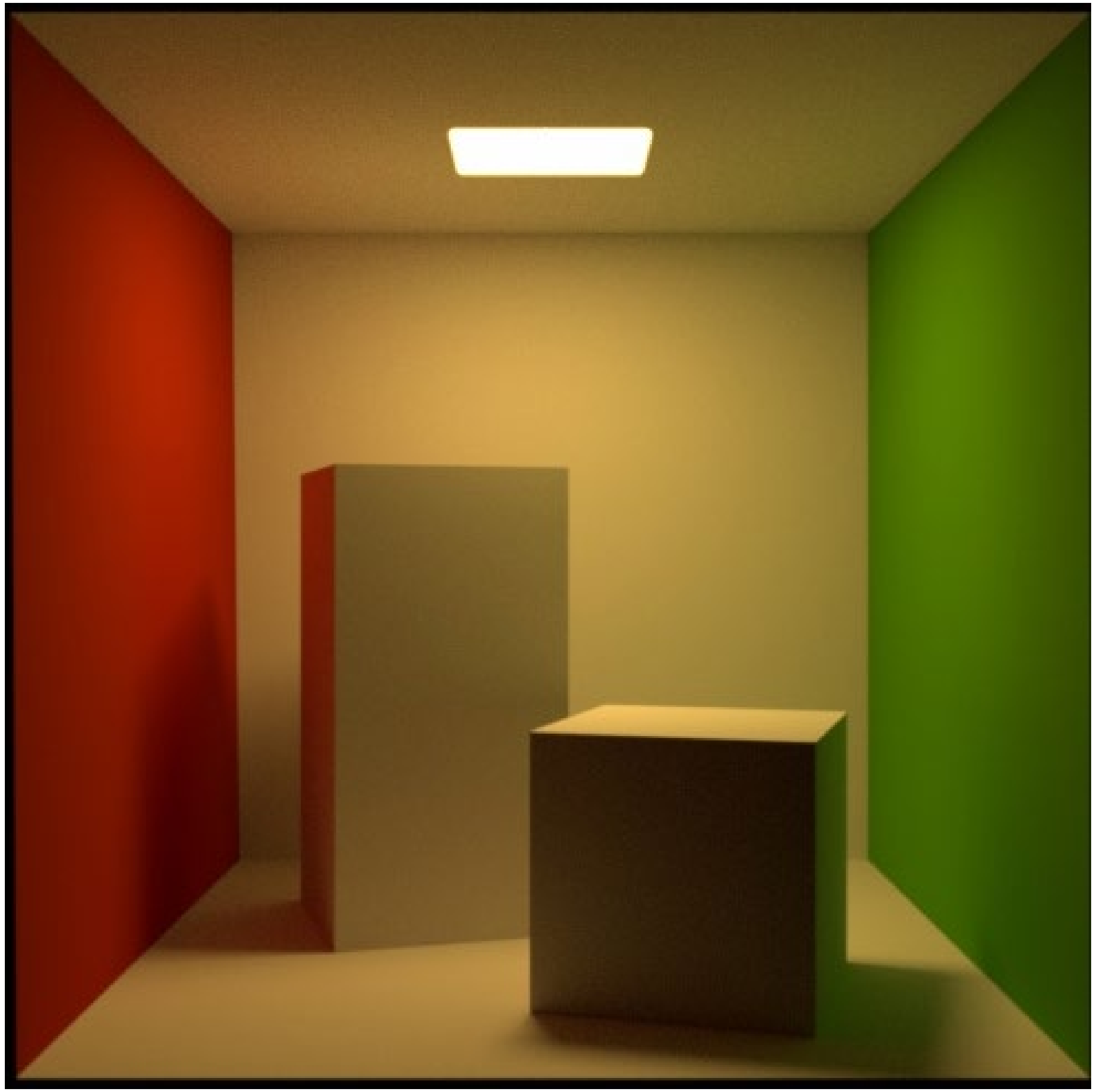
differentiable, inverse, and neural rendering

http://graphics.cs.cmu.edu/courses/15-468/

Topics to be covered

Basics of ray tracing:

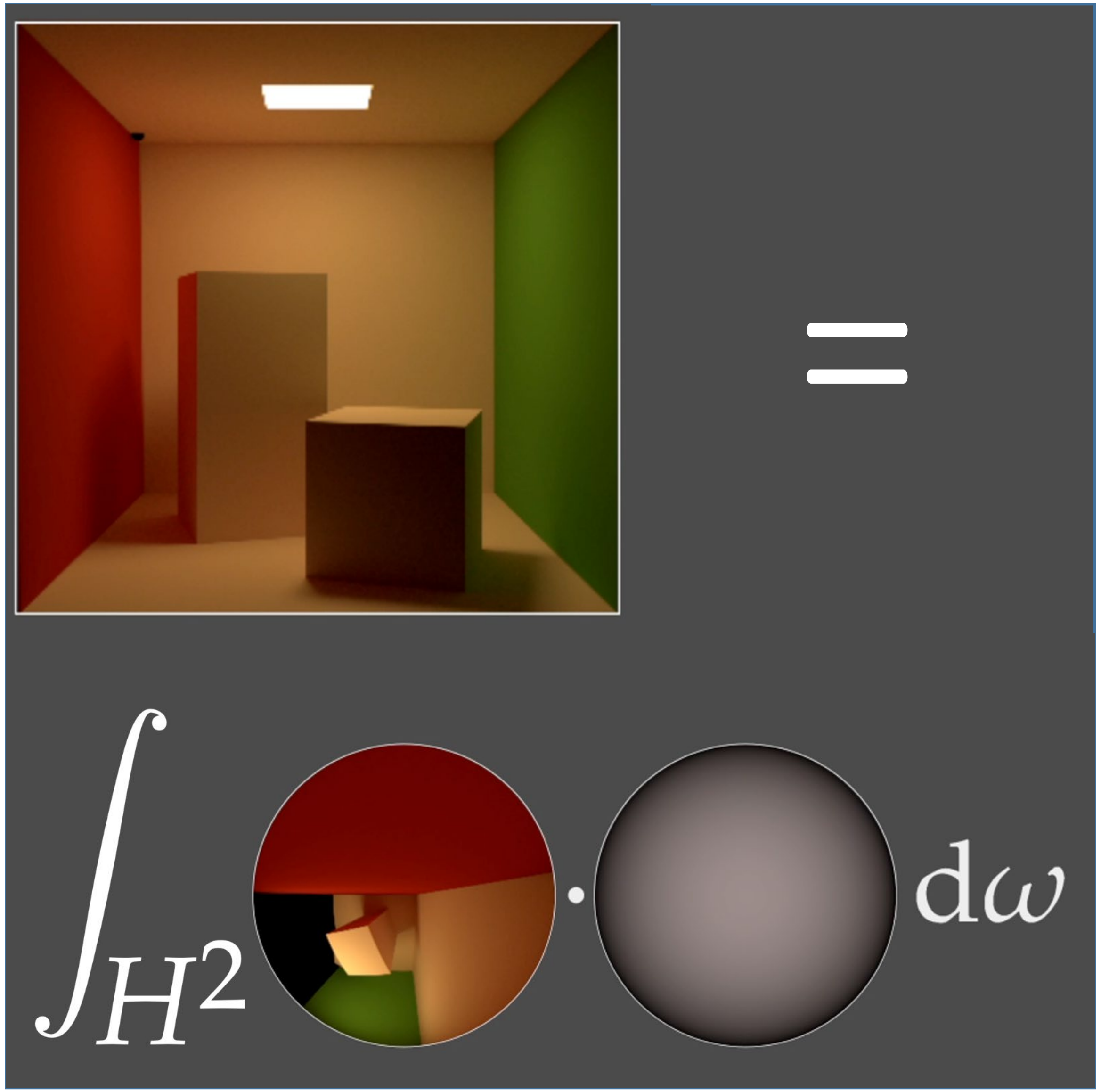
- trace-intersect recursions
- basic camera and illumination models
- shading
- intersection queries
- texture mapping



Topics to be covered

Theory of light transport and materials:

- rendering equation
- radiative transfer equation
- path integral formulations
- microfacet reflectance models
- statistical scattering models



Topics to be covered

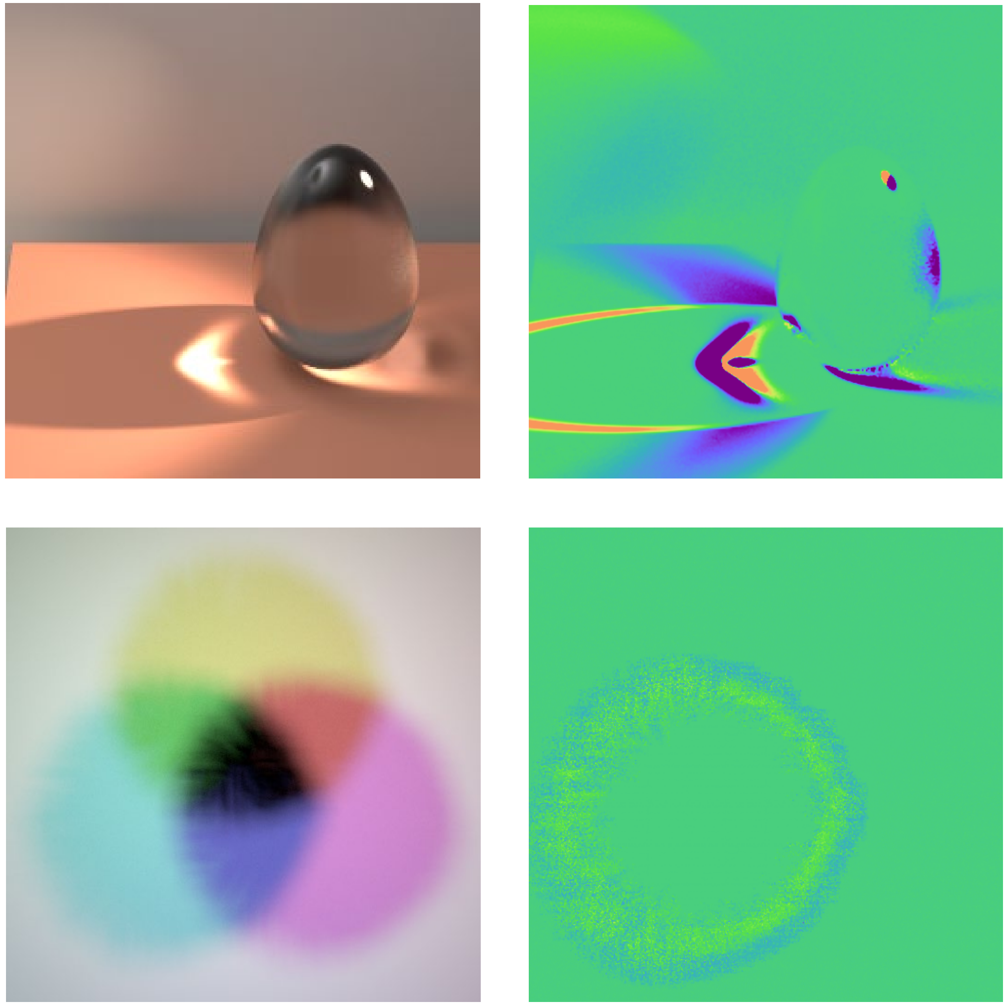
Monte Carlo rendering algorithms:

- unidirectional and bidirectional estimators
- Markov chain Monte Carlo techniques
- volumetric rendering
- importance sampling techniques
- quasi-Monte Carlo techniques

Topics to be covered

Advanced topics:

- differentiable rendering
- neural rendering
- rendering wave-optics effects
- rendering specular transport effects
- rendering eikonal transport effects



Overview of today’s lecture

- Pinholes and lenses.
- Pinhole camera.
- Accidental pinholes.
- Camera matrix.
- Perspective.
- Other camera models.
- Pose estimation.

Slide credits

Most of these slides were adapted from:

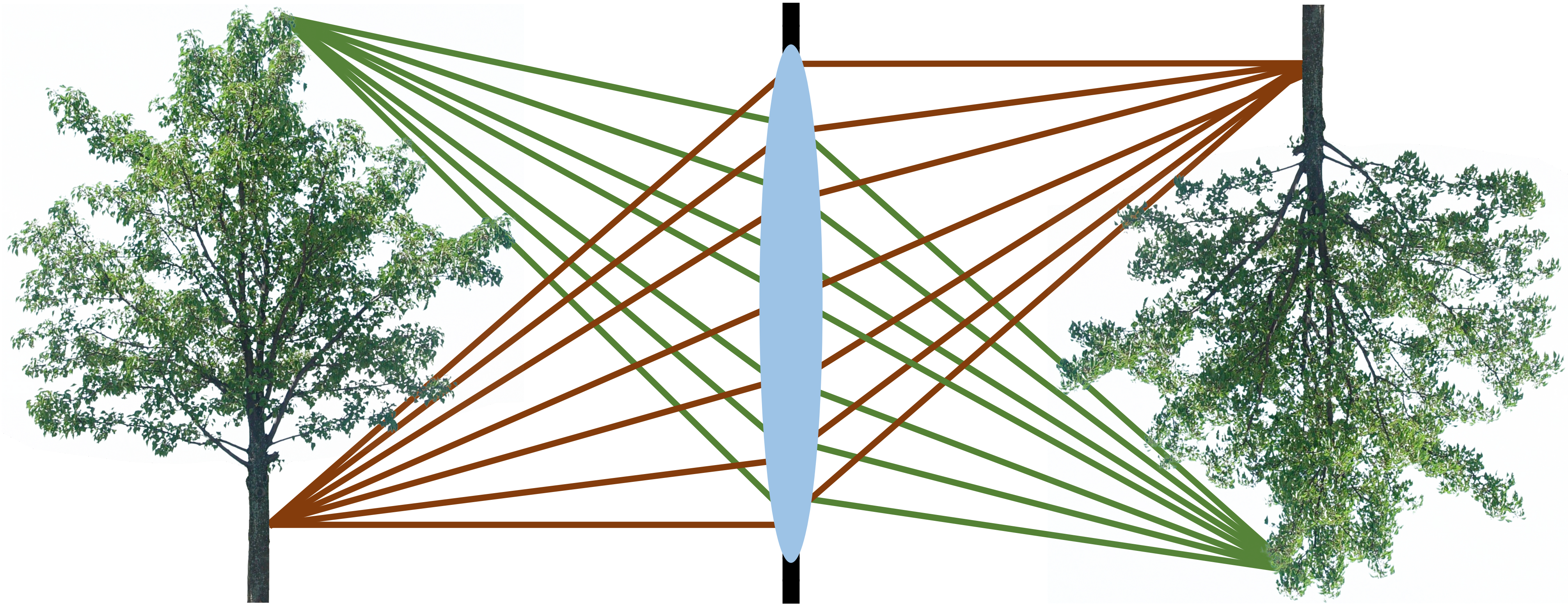
- Kris Kitani (15-463, Fall 2016).

Some slides inspired from:

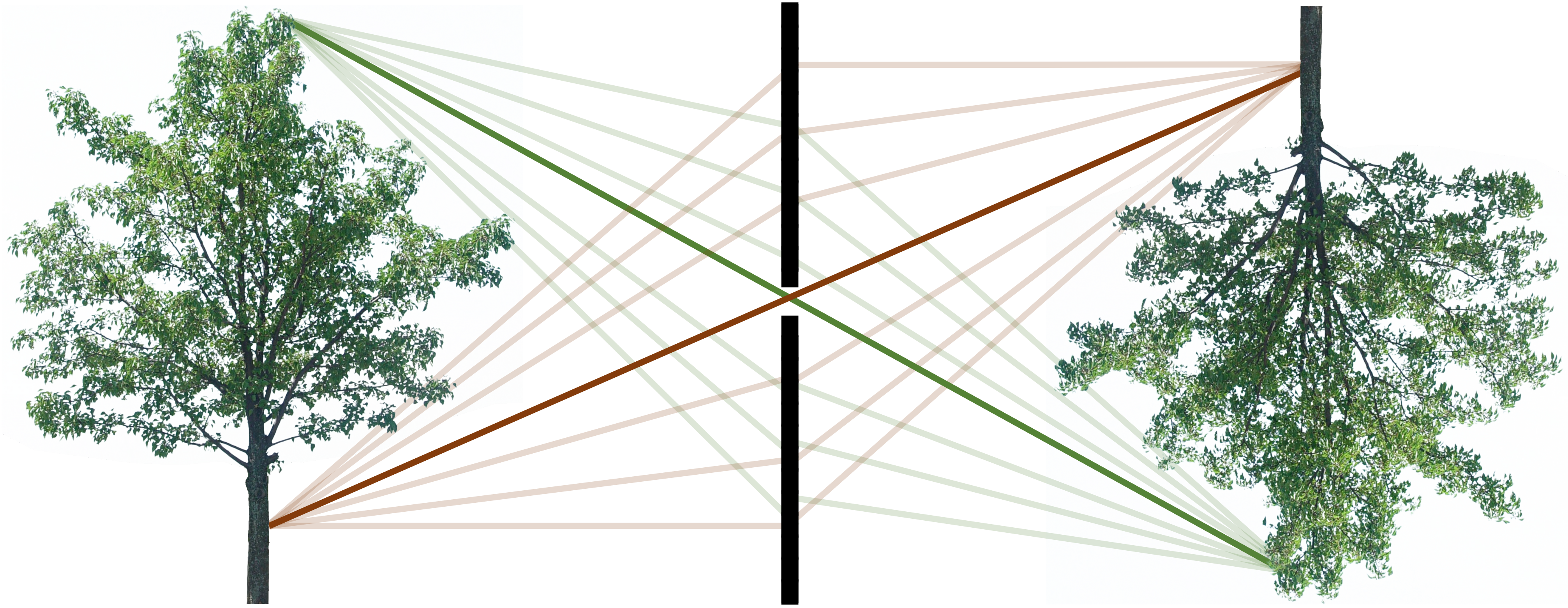
- Fredo Durand (MIT).

Pinhole and lens cameras

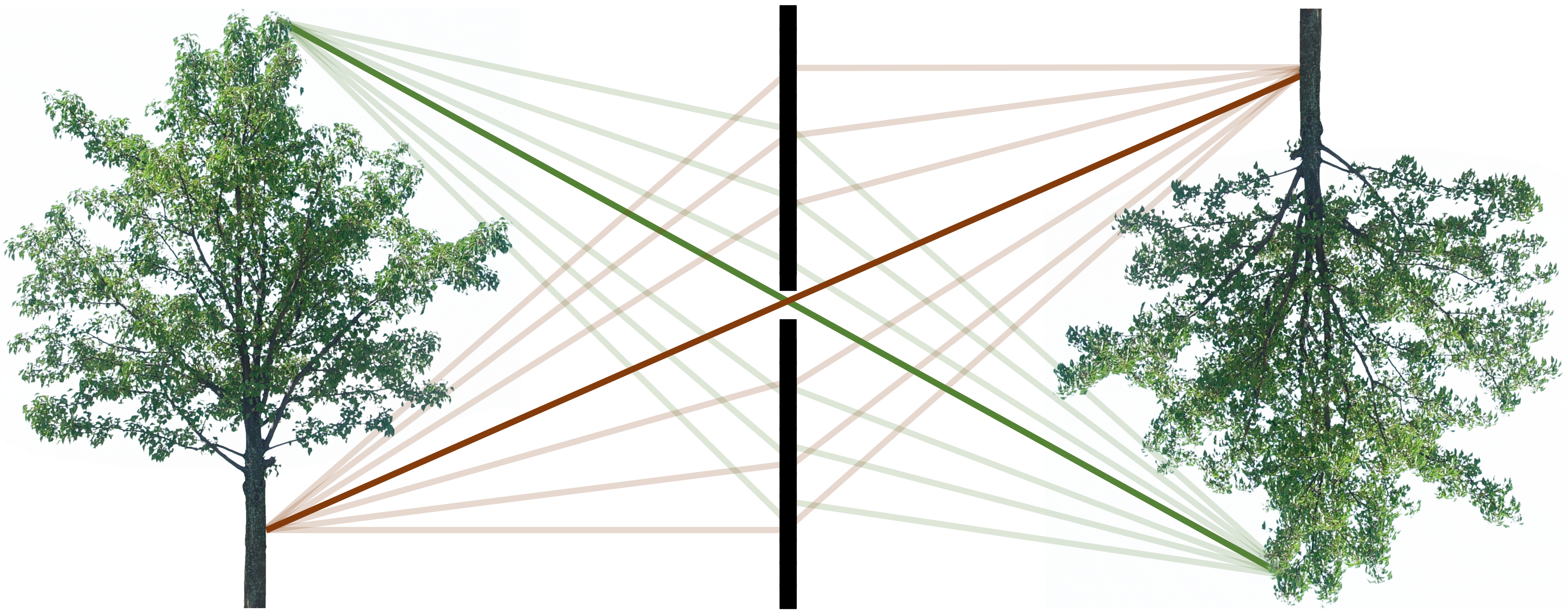
The lens camera



The pinhole camera

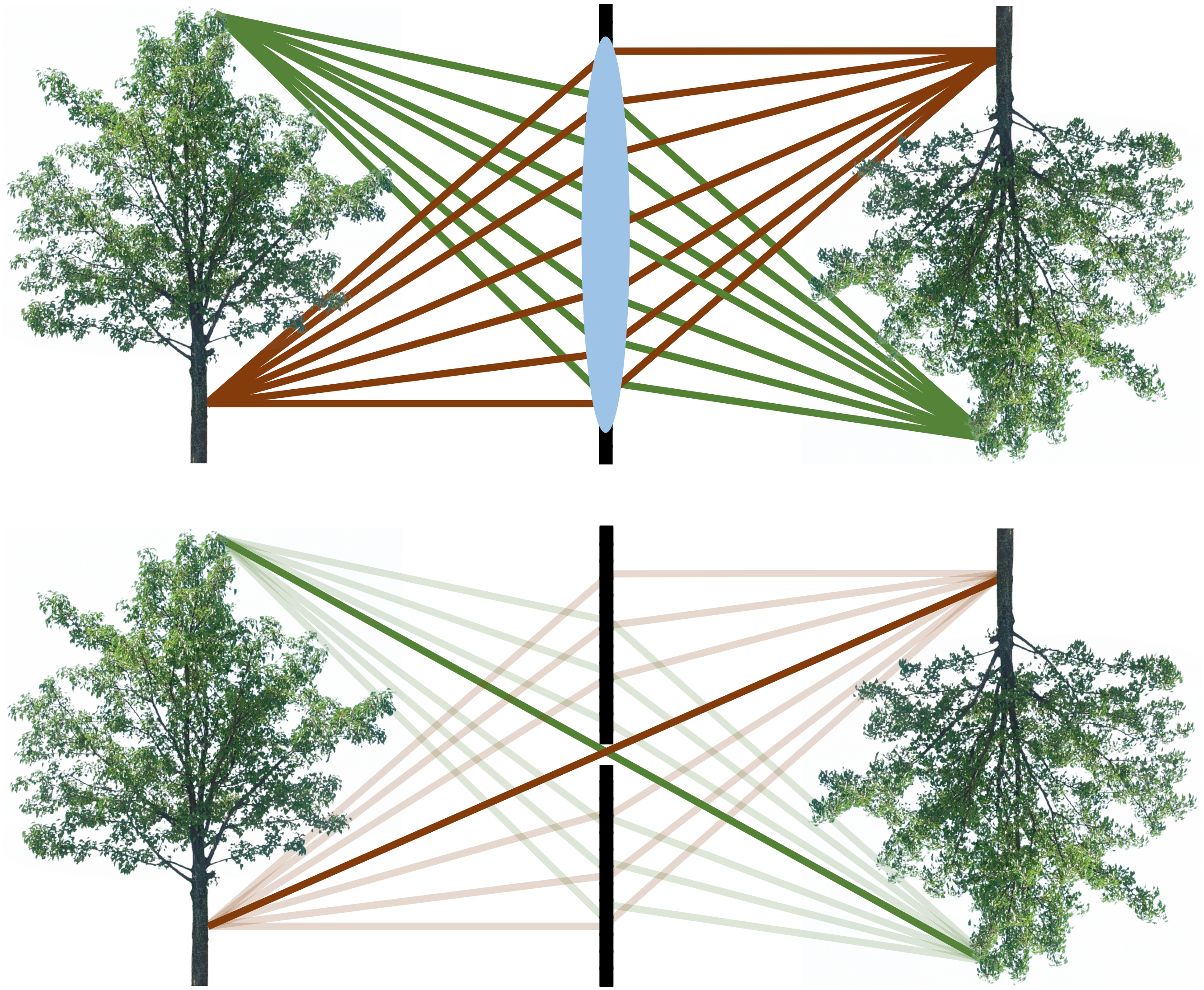


The pinhole camera



Central rays propagate in the same way for both models!

Describing both lens and pinhole cameras

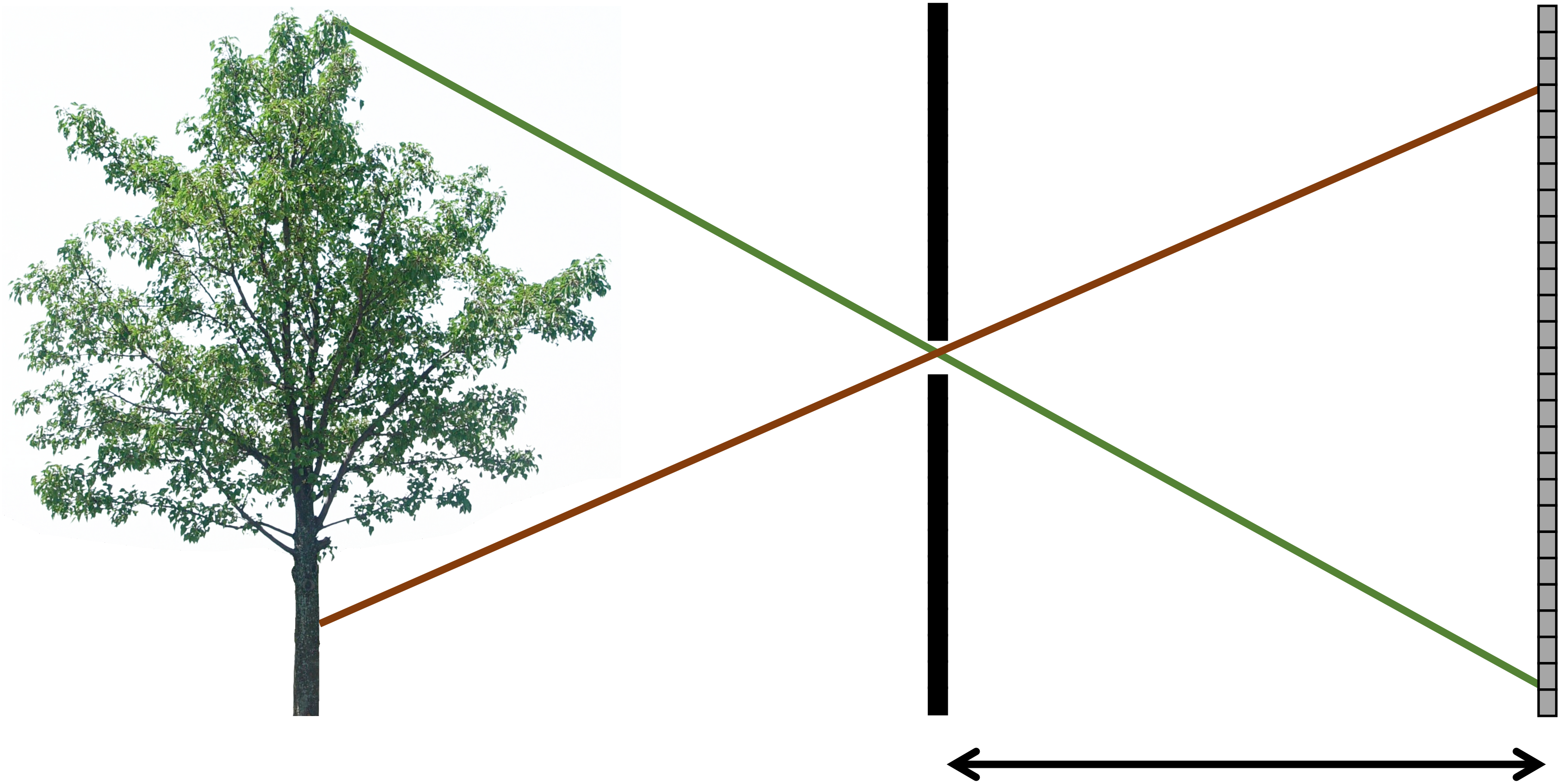


We can derive properties and descriptions that hold for both camera models if:

- We use only central rays.
- We assume the lens camera is in focus.

Important difference: focal length

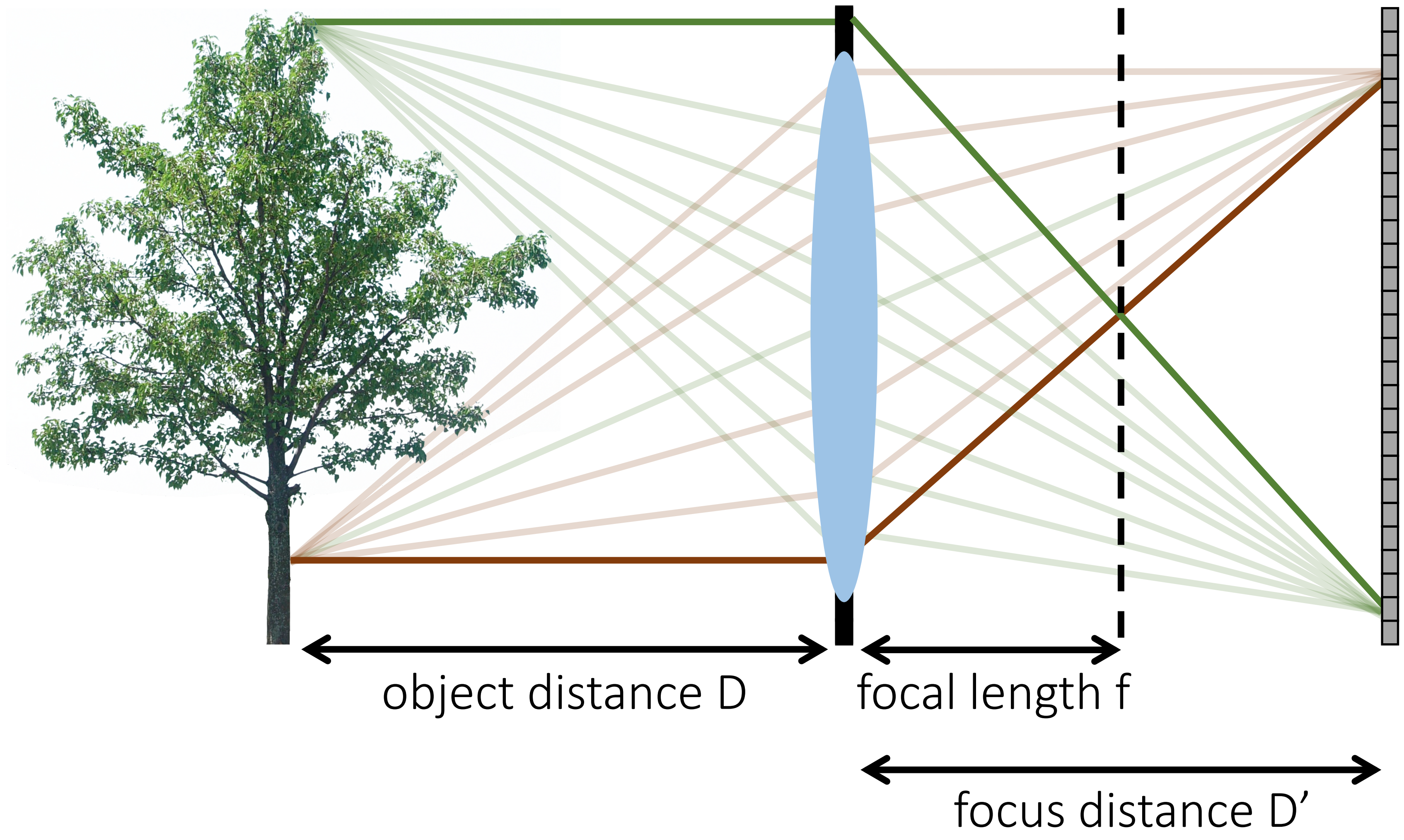
In a pinhole camera, focal length is distance between aperture and sensor



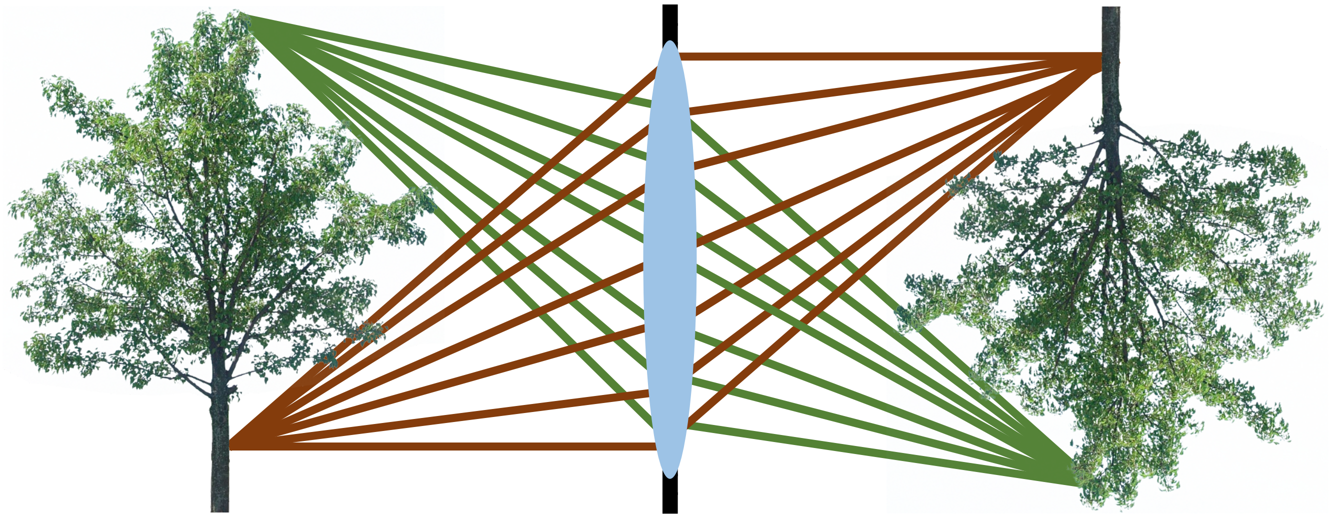
focal length f

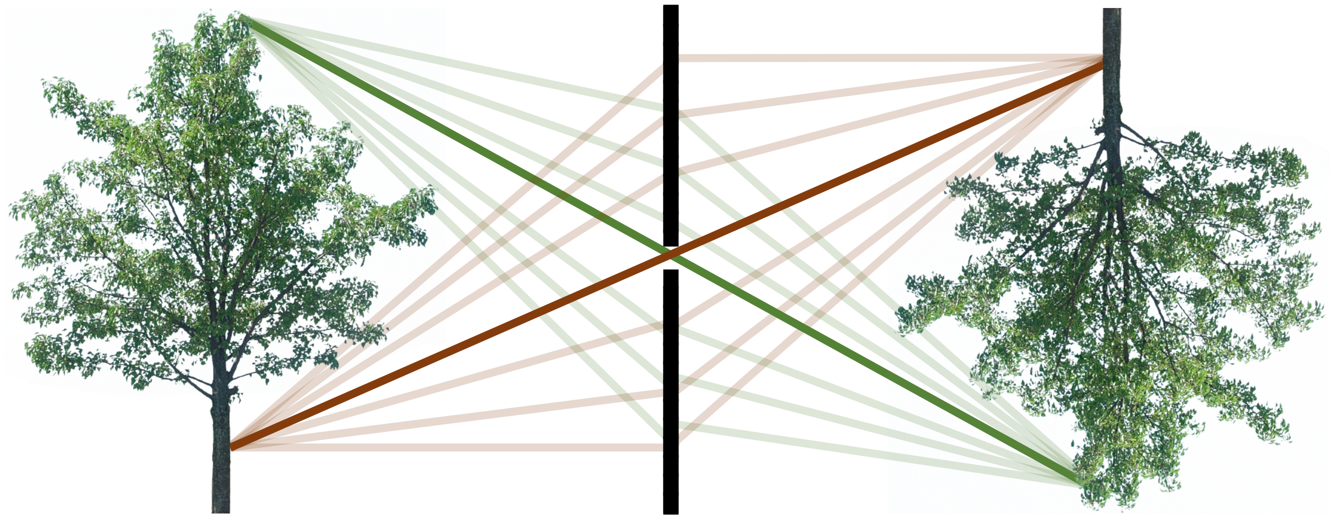
Important difference: focal length

In a lens camera, focal length is distance where parallel rays intersect



Describing both lens and pinhole cameras





We can derive properties and descriptions that hold for both camera models if:

- We use only central rays.
- We assume the lens camera is in focus.
- We assume that the focus distance of the lens camera is equal to the focal length of the pinhole camera.

Remember: *focal length* f refers to different things for lens and pinhole cameras.

- In this lecture, we use it to refer to the aperture-sensor distance, as in the pinhole camera case.

Camera matrix

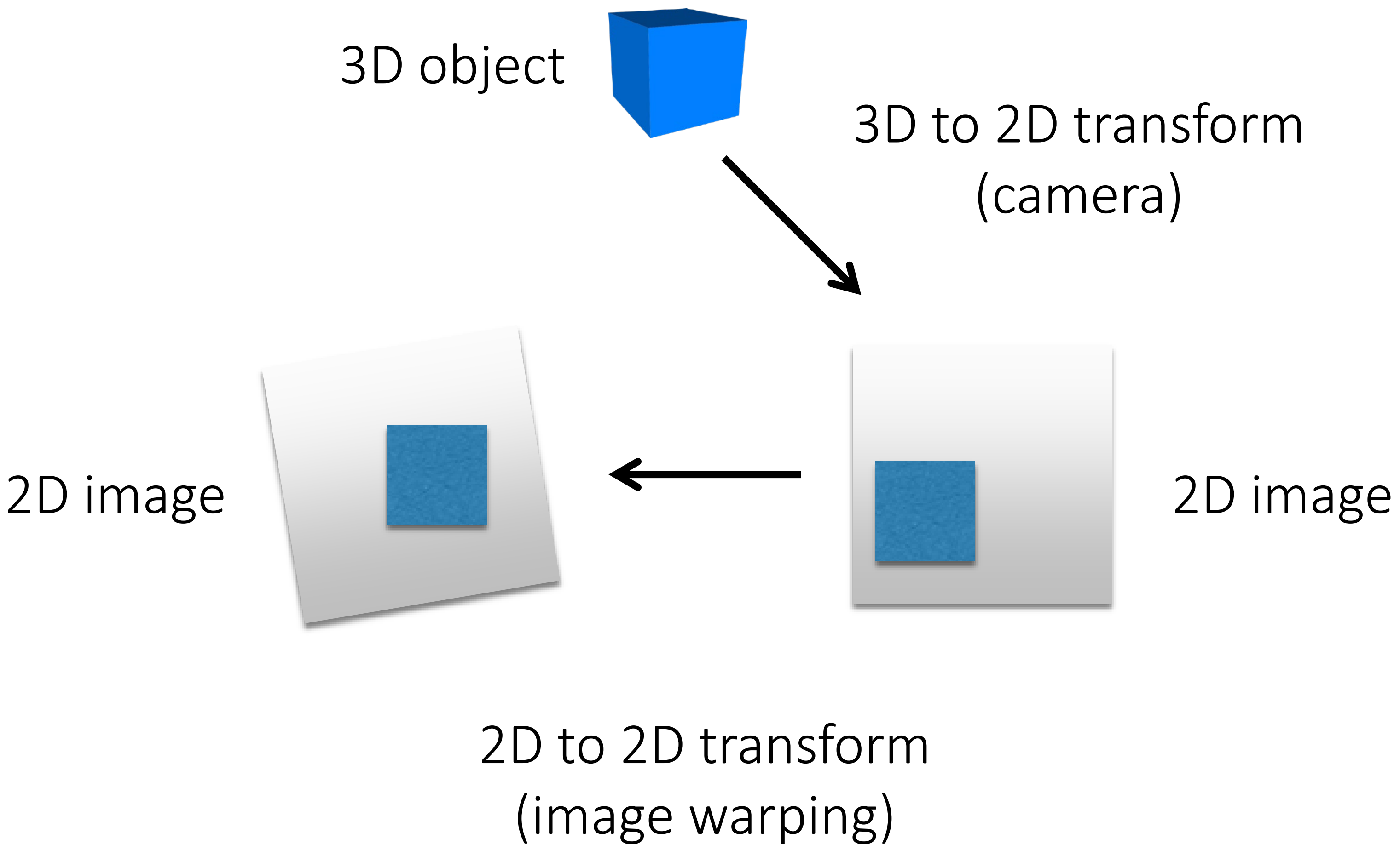
The camera as a coordinate transformation

A camera is a mapping from:

the 3D world

to:

a 2D image



The camera as a coordinate transformation

A camera is a mapping from:

the 3D world

to:

a 2D image

homogeneous coordinates

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

2D image point

camera matrix

3D world point

What are the dimensions of each variable?

Reminder: 2D homogeneous coordinates

heterogeneous coordinates		homogeneous coordinates	
$\begin{bmatrix} x \\ y \end{bmatrix}$	$\Rightarrow$	$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	add a 1 here

- Represent 2D point with a 3D vector

Reminder: 2D homogeneous coordinates

heterogeneous coordinates | homogeneous coordinates

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \stackrel{\text{def}}{=} \begin{bmatrix} ax \\ ay \\ a \end{bmatrix}$$

- Represent 2D point with a 3D vector
- 3D vectors are only defined up to scale

Reminder: 2D homogeneous coordinates

Conversion:

- heterogeneous → homogeneous

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- homogeneous → heterogeneous

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \begin{bmatrix} x/z \\ y/z \end{bmatrix}$$

Scale invariance:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = a \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Special points:

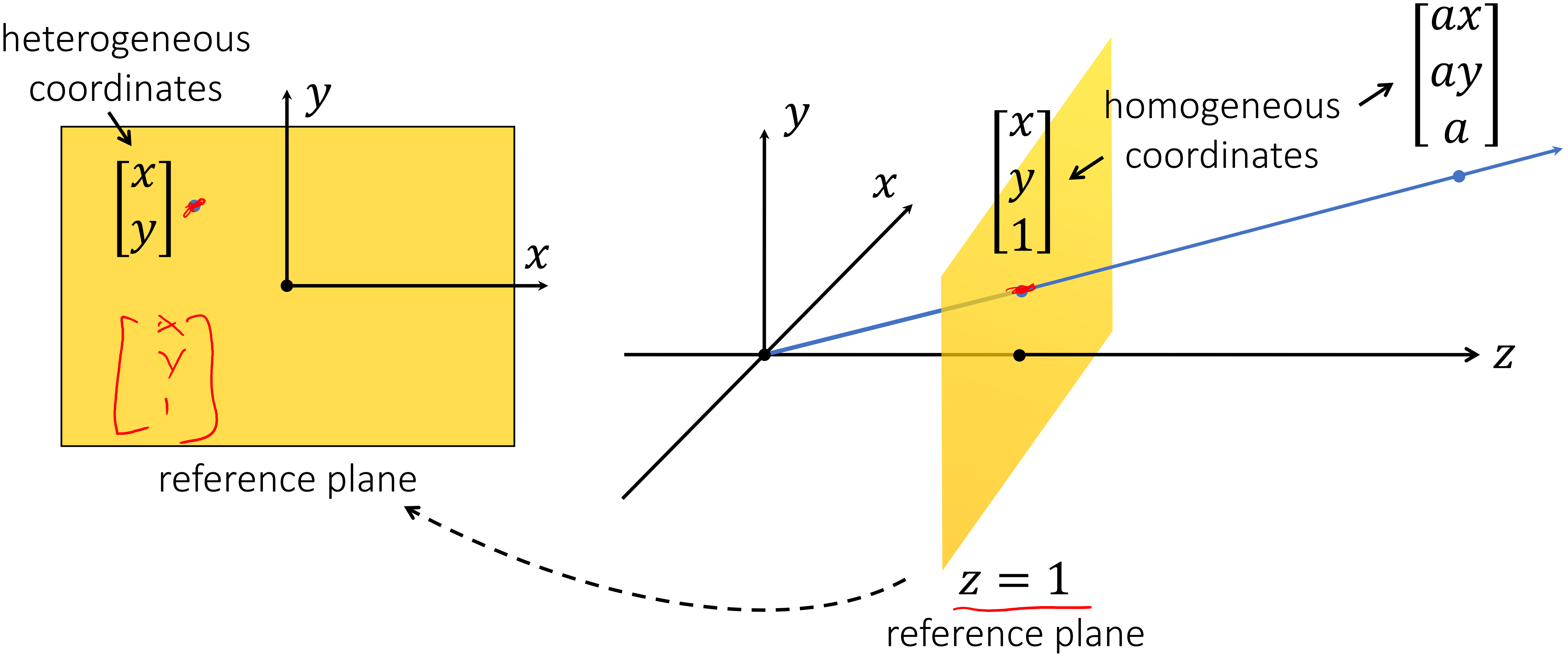
- point at infinity

$$\begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$$

- undefined

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Reminder: 2D projective geometry



Through the scale invariance property, homogeneous coordinates map all points on a line passing through the origin to the point where this line intersects the reference plane.

Reminder: 3D homogeneous coordinates

heterogeneous coordinates | homogeneous coordinates

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \Rightarrow \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \stackrel{\text{def}}{=} \begin{bmatrix} aX \\ aY \\ aZ \\ a \end{bmatrix}$$

- Represent 3D point with a 4D vector
- 4D vectors are only defined up to scale

Reminder: notation

	heterogeneous coordinates	homogeneous coordinates
2D coordinates	2D vector $\tilde{\boldsymbol{x}} = \begin{bmatrix} x \\ y \end{bmatrix}$	3D vector $\boldsymbol{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
3D coordinates	3D vector $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$	4D vector $\boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$

The camera as a coordinate transformation

A camera is a mapping from:

the 3D world

to:

a 2D image

homogeneous coordinates

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

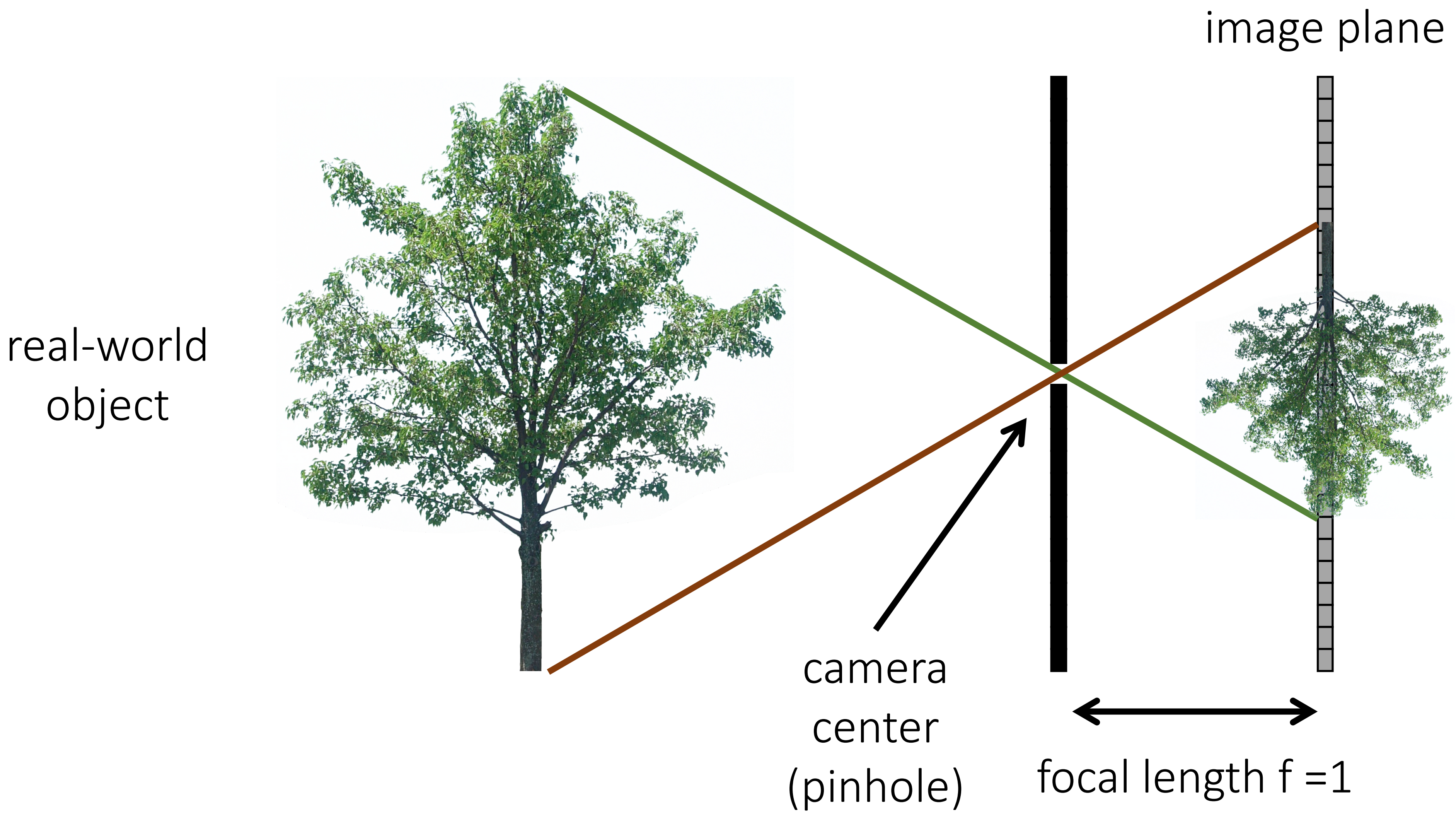
2D image point

camera matrix

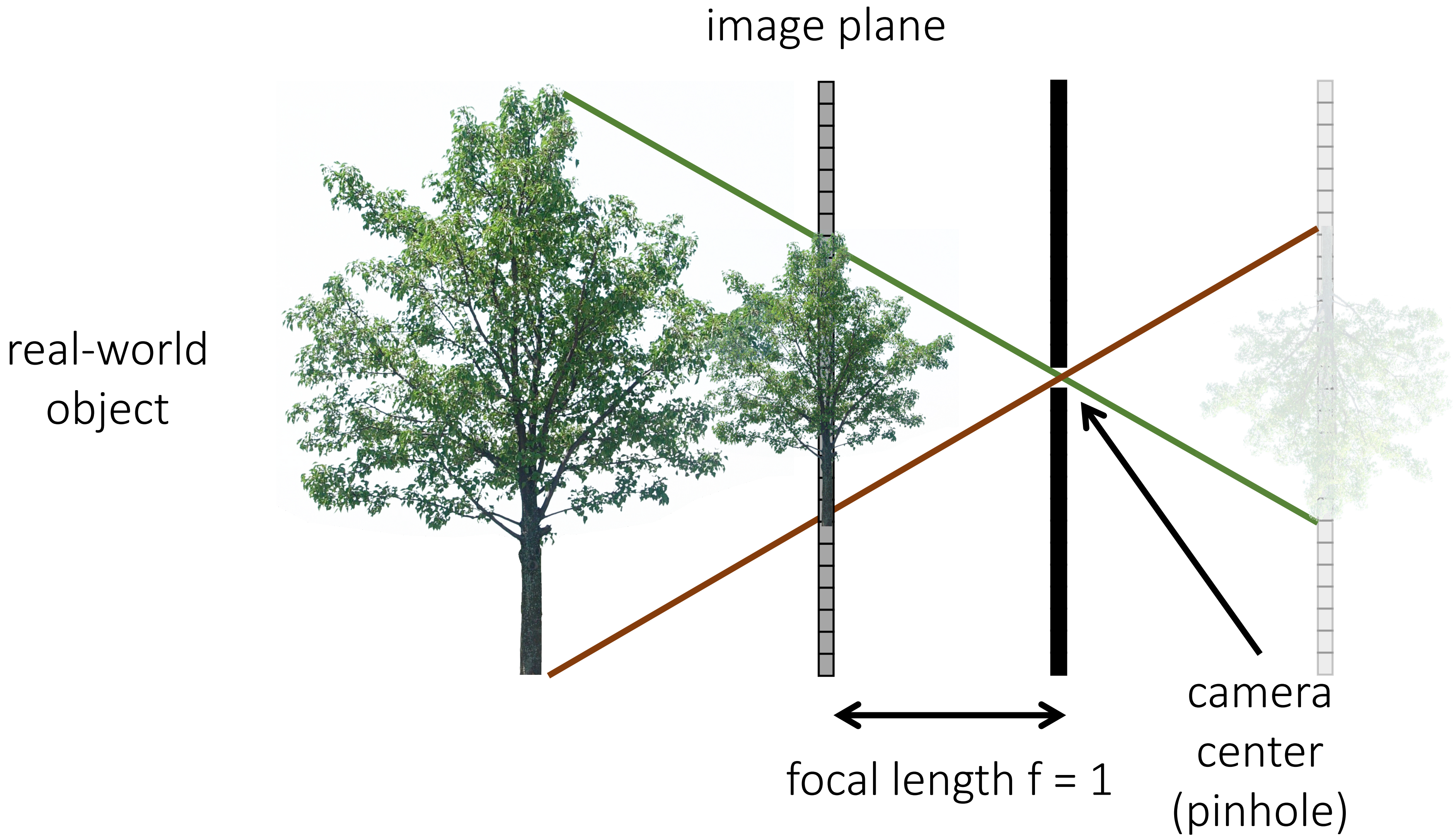
3D world point

What does this transformation look like?

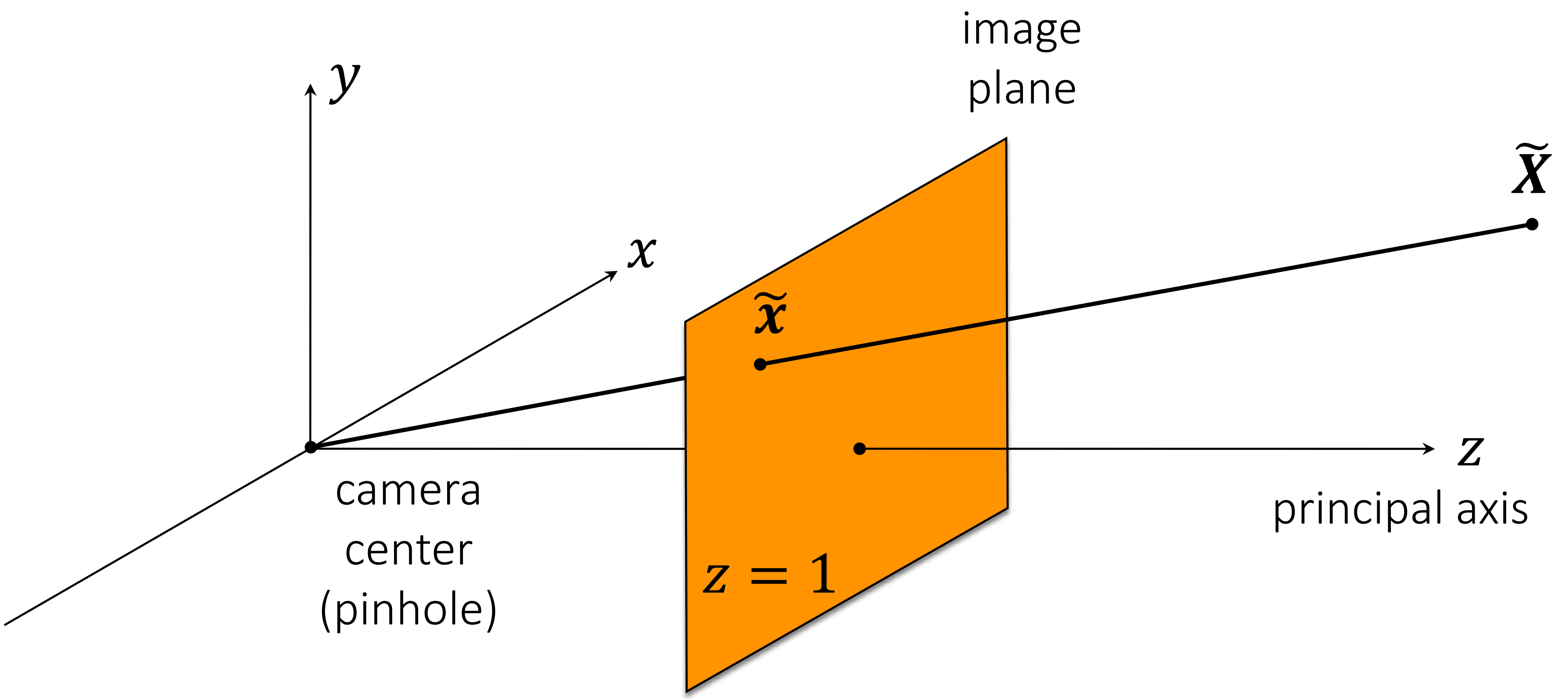
The pinhole camera



The (rearranged) pinhole camera

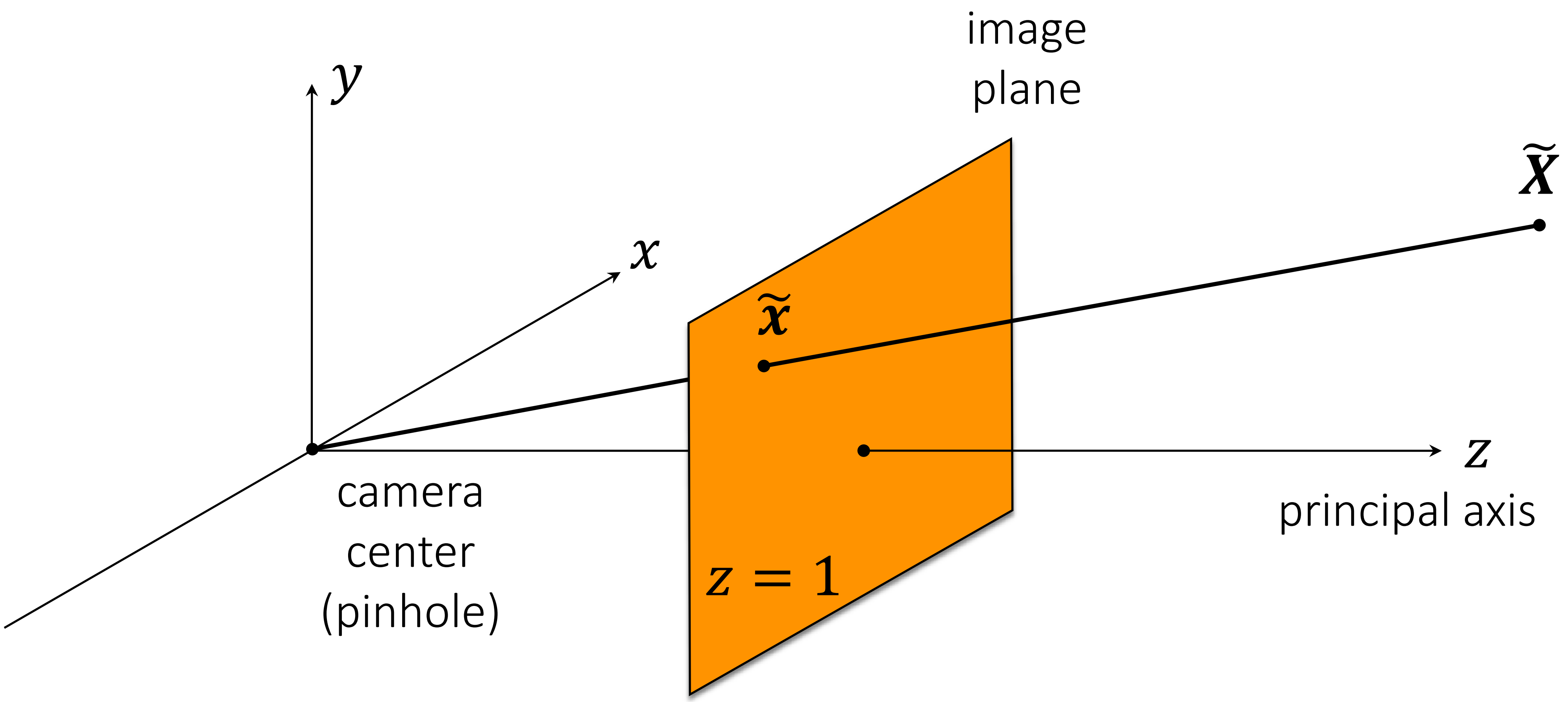


The (rearranged) pinhole camera



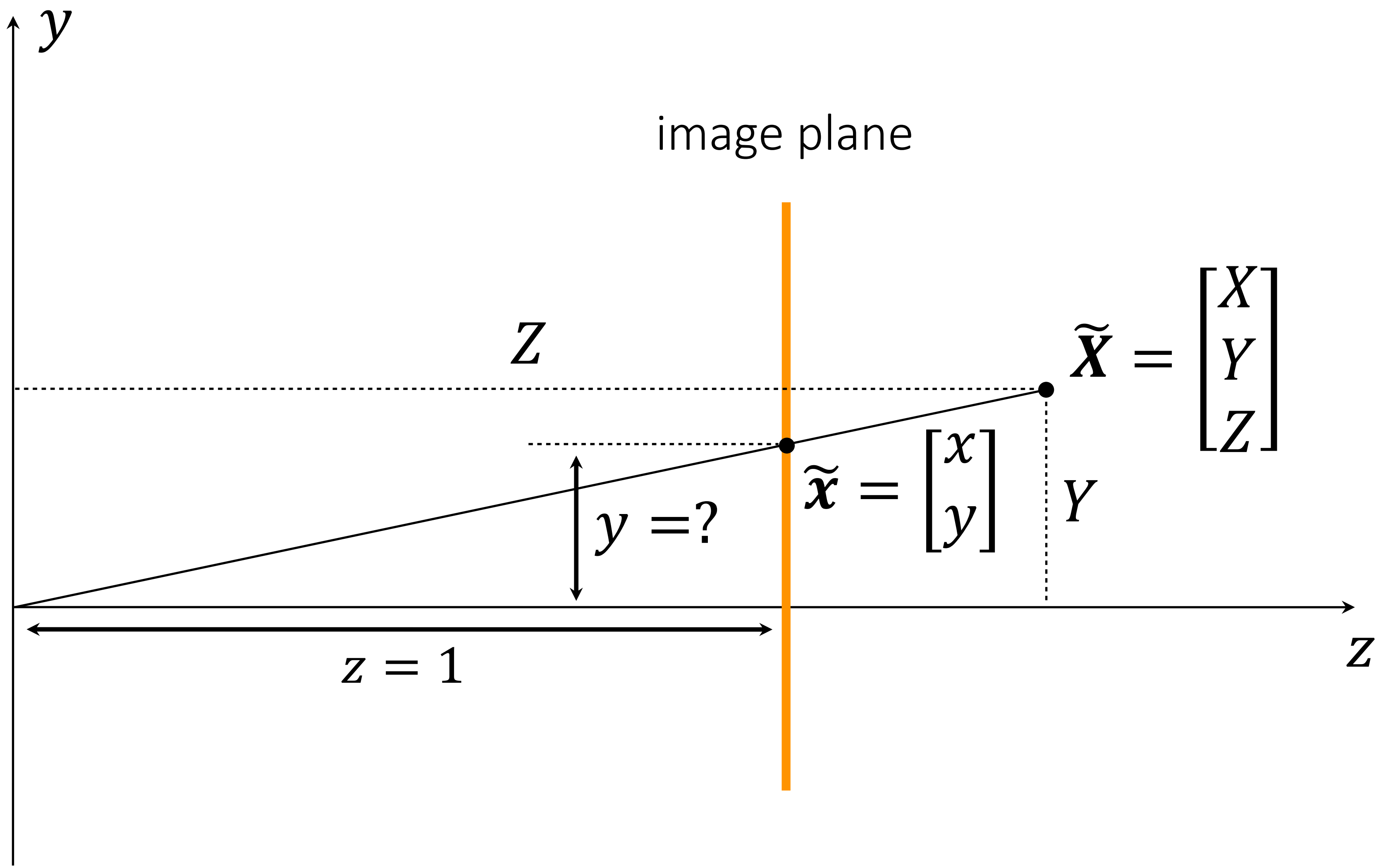
Where did we see a similar picture?

The (rearranged) pinhole camera



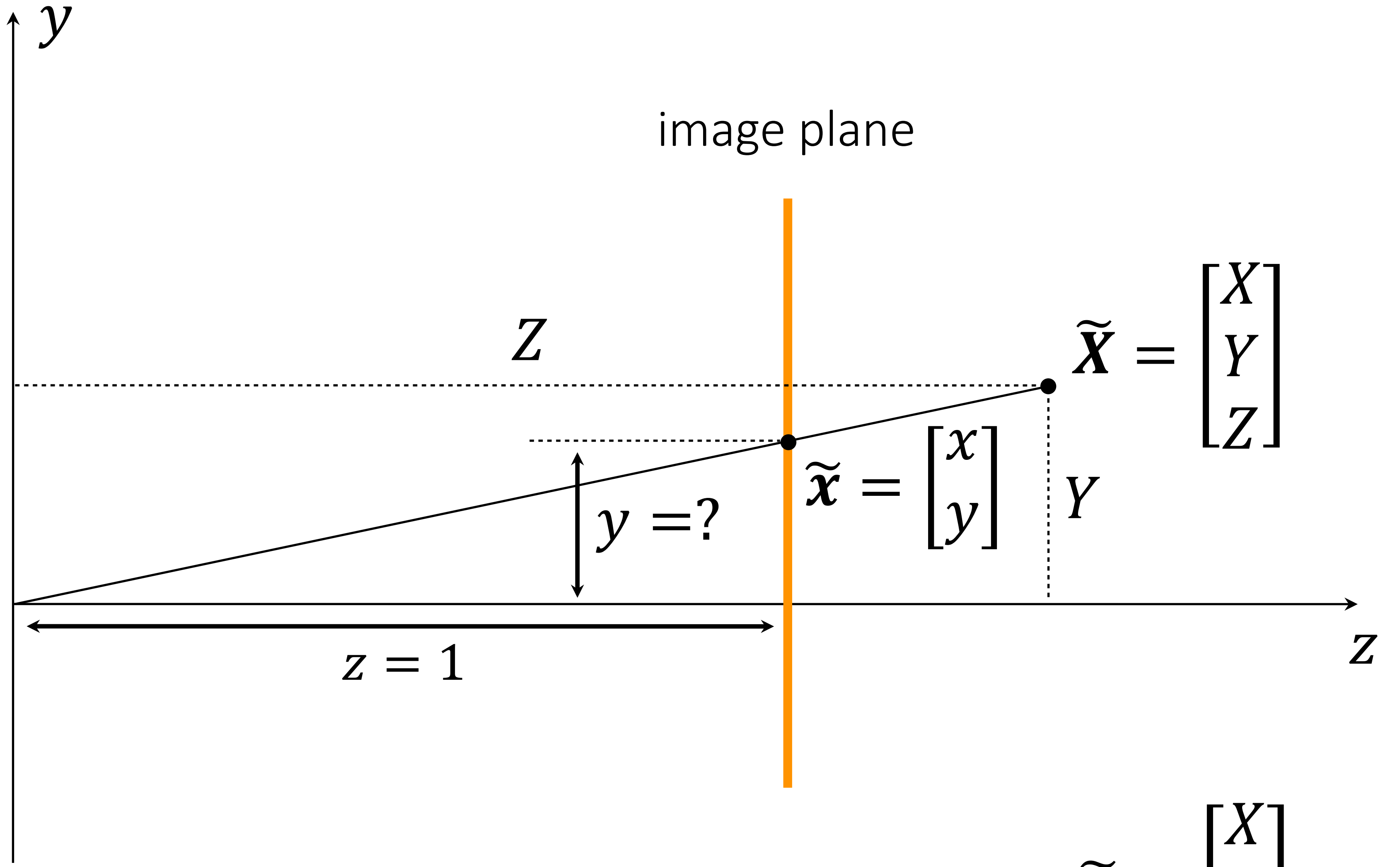
What is the equation for image coordinate $\tilde{\boldsymbol{x}}$ in terms of $\tilde{\boldsymbol{X}}$?

The 2D view of the (rearranged) pinhole camera



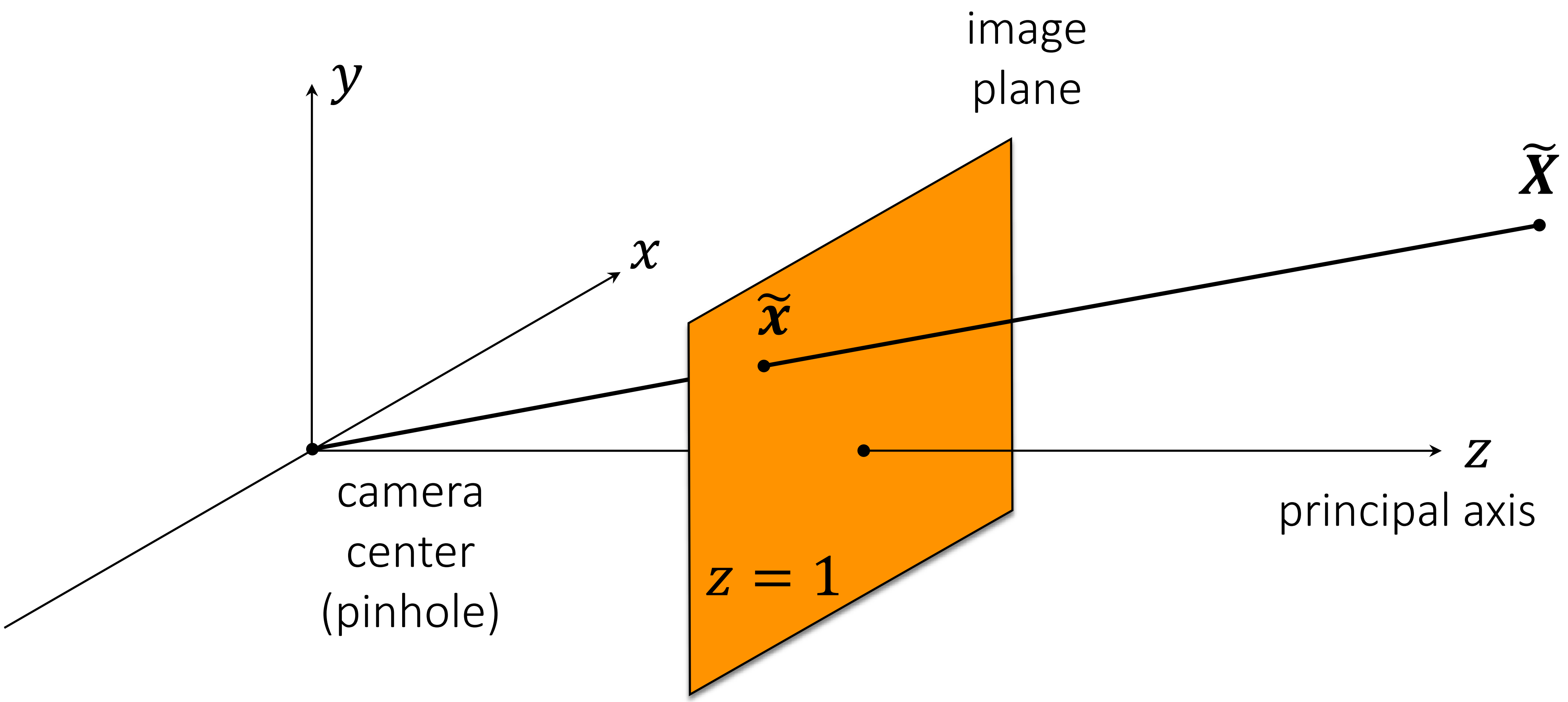
What is the equation for image coordinate $\tilde{\boldsymbol{x}}$ in terms of $\tilde{\boldsymbol{X}}$?

The 2D view of the (rearranged) pinhole camera



Using similar triangles: $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$

The (rearranged) pinhole camera



What is the camera matrix $\boldsymbol{P}$ for a pinhole camera?

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

The pinhole camera matrix

Camera projection relationship expressed:

- in *heterogeneous coordinates*

$$\widetilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \widetilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$$

- in *homogeneous coordinates*

$$\boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \boldsymbol{x} = ?$$

The pinhole camera matrix

Camera projection relationship expressed:

- in *heterogeneous coordinates*

$$\widetilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \widetilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$$

- in *homogeneous coordinates*

$$\boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \boldsymbol{x} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

General camera model in *homogeneous coordinates*:

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

What does the pinhole camera projection look like?

$$\boldsymbol{P} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$$

The pinhole camera matrix

Camera projection relationship expressed:

- in *heterogeneous coordinates*

$$\widetilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \widetilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$$

- in *homogeneous coordinates*

$$\boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \boldsymbol{x} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

General camera model in *homogeneous coordinates*:

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

What does the pinhole camera projection look like?

The perspective projection matrix

$$\boldsymbol{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The pinhole camera matrix

Camera projection relationship expressed:

- in *heterogeneous coordinates*

$$\widetilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \widetilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$$

- in *homogeneous coordinates*

$$\boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \boldsymbol{x} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

General camera model in *homogeneous coordinates*:

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

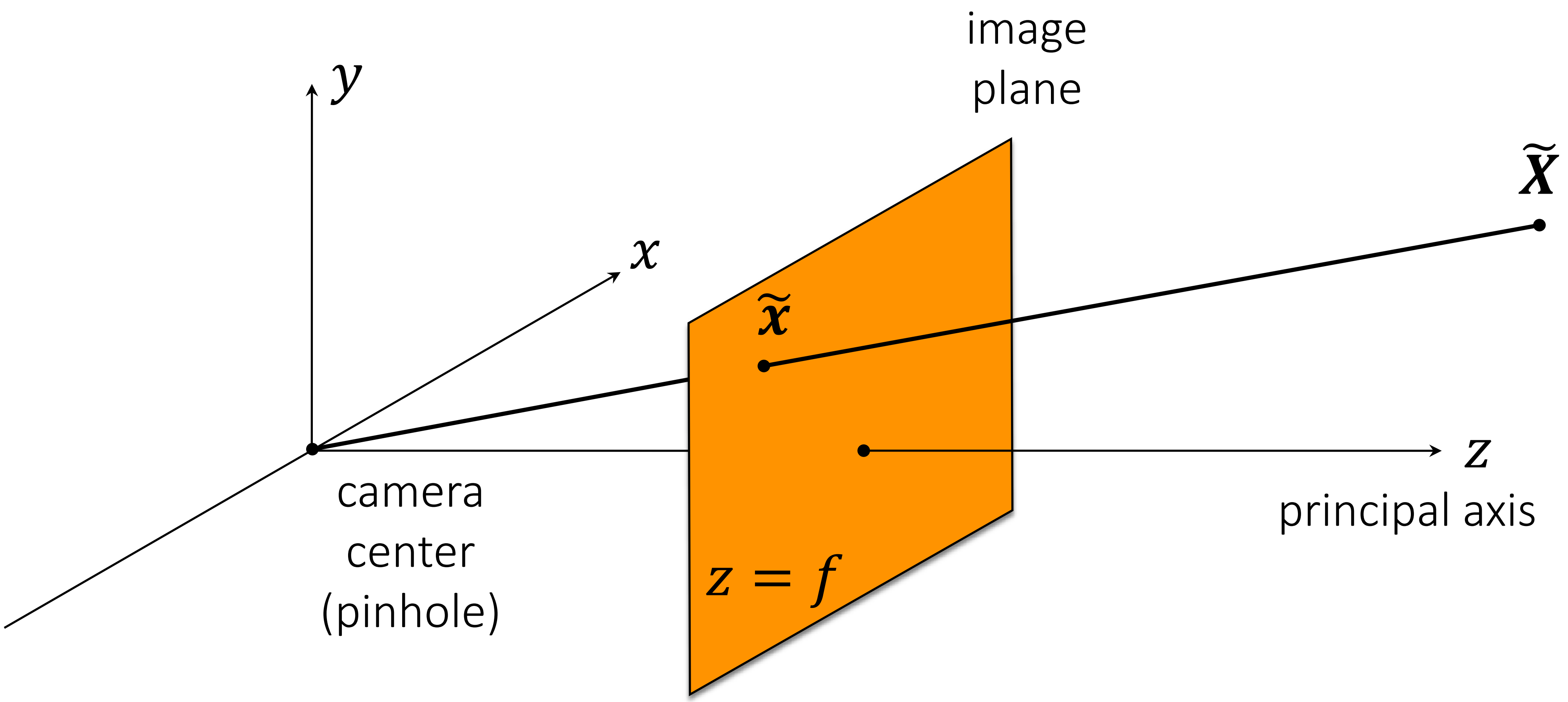
What does the pinhole camera projection look like?

The *perspective projection matrix*

$$\boldsymbol{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = [\boldsymbol{I} \mid \boldsymbol{0}]$$

alternative way to write the same thing

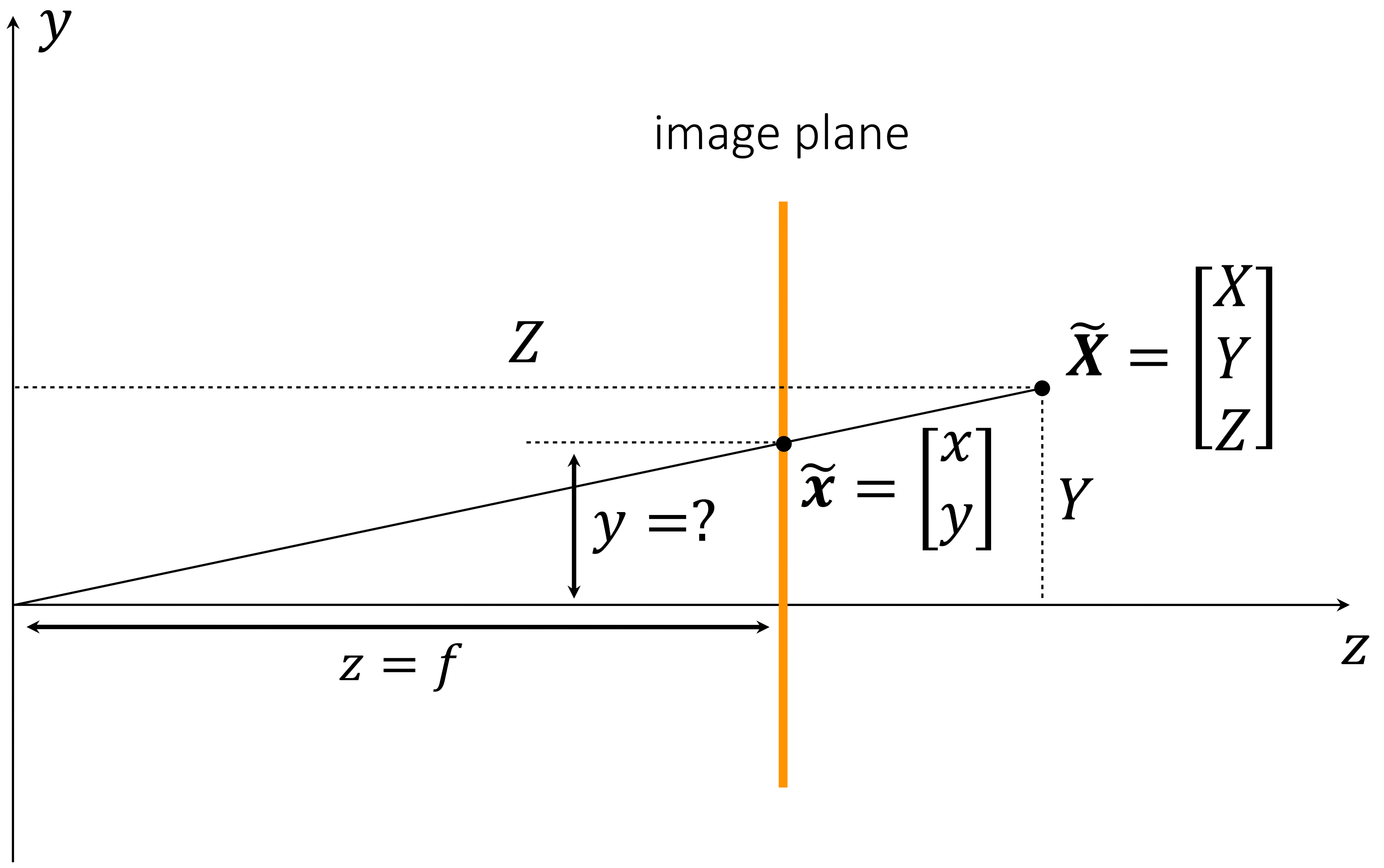
More general case: arbitrary focal length



What is the camera matrix $\boldsymbol{P}$ for a pinhole camera?

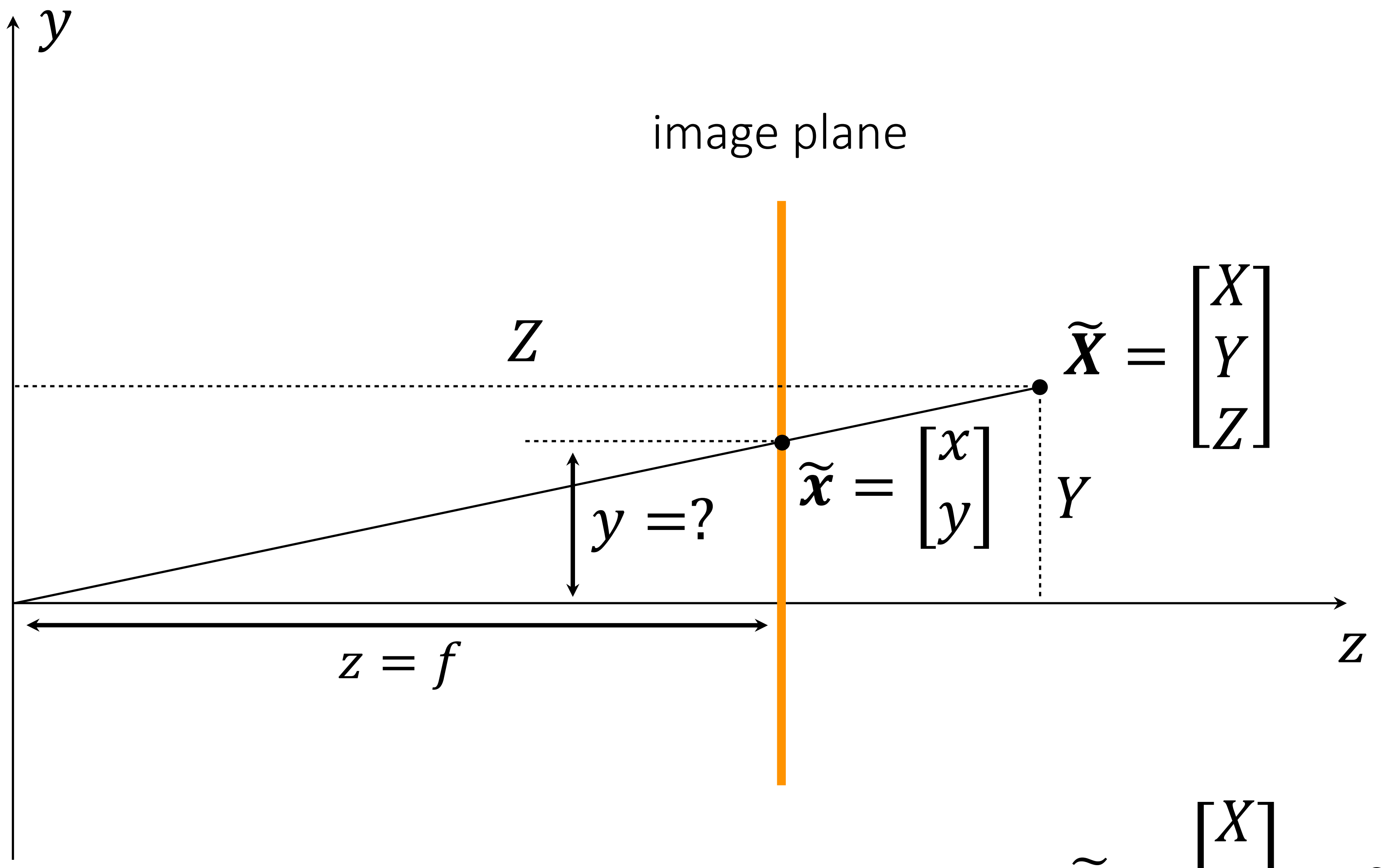
$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

More general (2D) case: arbitrary focal length



What is the equation for image coordinate $\tilde{\boldsymbol{x}}$ in terms of $\tilde{\boldsymbol{X}}$?

More general (2D) case: arbitrary focal length



Using similar triangles: $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} fX/Z \\ fY/Z \end{bmatrix}$

The pinhole camera matrix for arbitrary focal length

Camera projection relationship expressed:

- in *heterogeneous coordinates*

$$\widetilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \widetilde{\boldsymbol{x}} = \begin{bmatrix} fX/Z \\ fY/Z \end{bmatrix}$$

- in *homogeneous coordinates*

$$\boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \boldsymbol{x} = \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix}$$

General camera model in *homogeneous coordinates*:

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

What does the pinhole camera projection look like?

$$\boldsymbol{P} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$$

The pinhole camera matrix for arbitrary focal length

Camera projection relationship expressed:

- in *heterogeneous coordinates*

$$\widetilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \widetilde{\boldsymbol{x}} = \begin{bmatrix} fX/Z \\ fY/Z \end{bmatrix}$$

- in *homogeneous coordinates*

$$\boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \boldsymbol{x} = \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix}$$

General camera model in *homogeneous coordinates*:

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

What does the pinhole camera projection look like?

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The pinhole camera matrix for arbitrary focal length

Camera projection relationship expressed:

- in *heterogeneous coordinates*

$$\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} fX/Z \\ fY/Z \end{bmatrix}$$

- in *homogeneous coordinates*

$$\boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \boldsymbol{x} = \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix}$$

General camera model in *homogeneous coordinates*:

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}$$

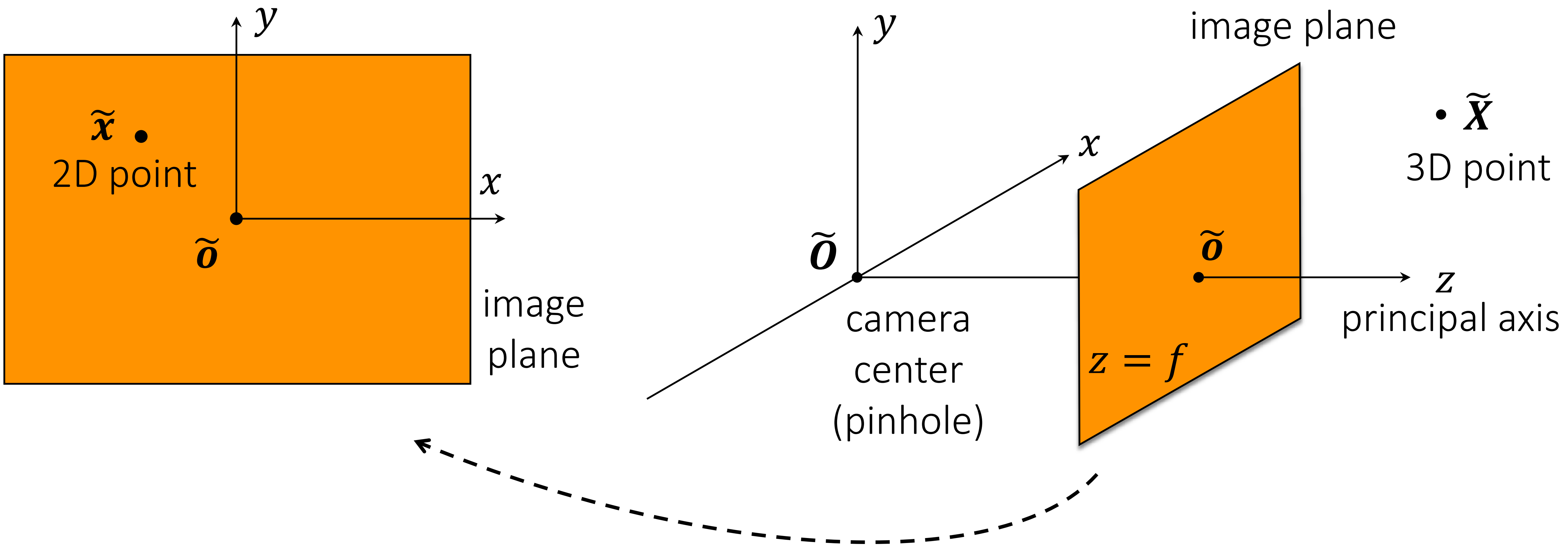
What does the pinhole camera projection look like?

Equivalently we can write:

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

combination of perspective projection and a 2D scaling transformation

Generalizations: coordinate systems

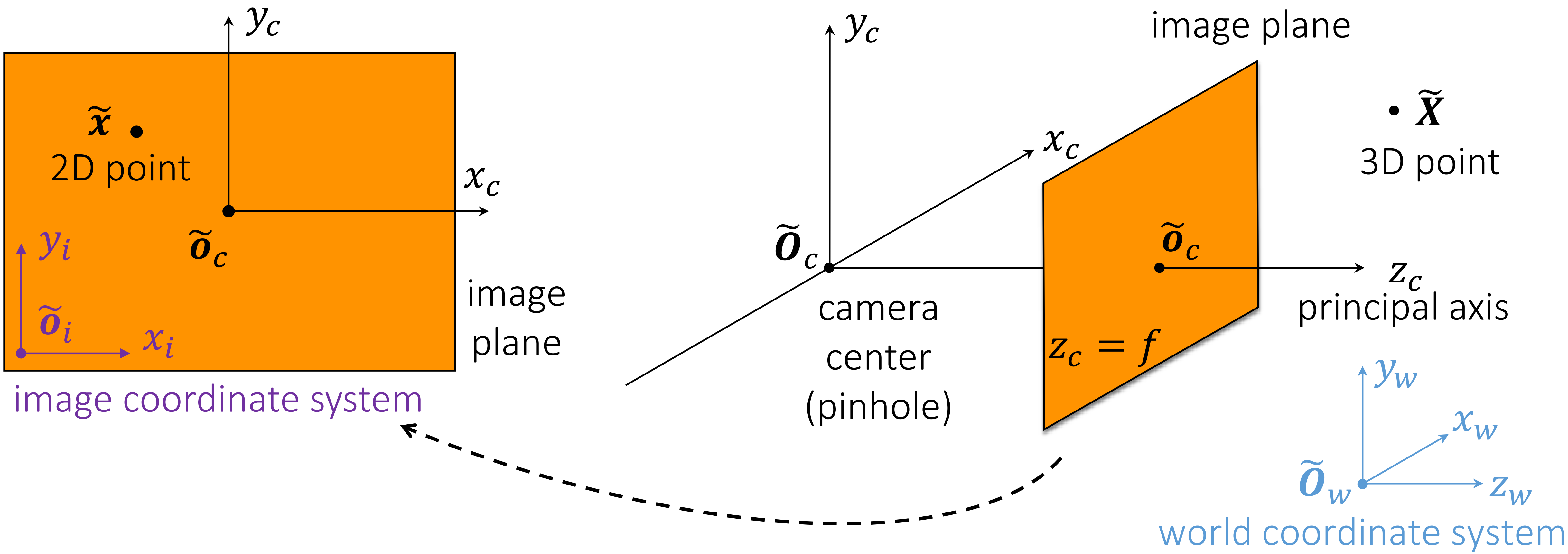


2D camera coordinate system

3D camera coordinate system

- A camera introduces two related coordinate systems, in 3D (world), and in 2D (image plane).

Generalizations: coordinate systems



2D camera coordinate system

3D camera coordinate system

- A camera introduces two related coordinate systems, in 3D (world), and in 2D (image plane).
- These coordinate systems may be different from the coordinate systems of our application.

Generalization: image coordinate system

In particular, the camera origin and image origin may be different.

- Can you think of a case when this happens?

image plane

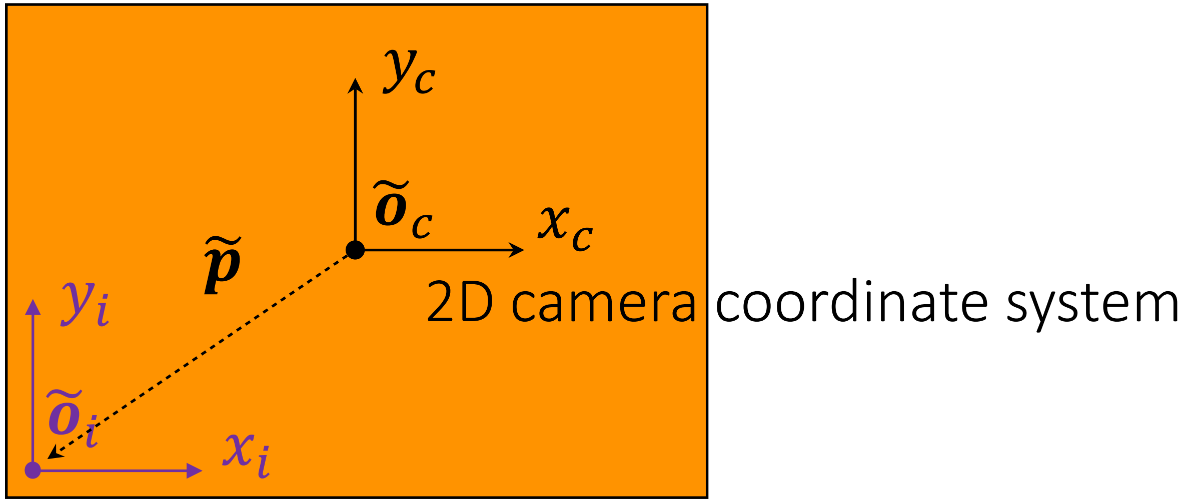


image coordinate system

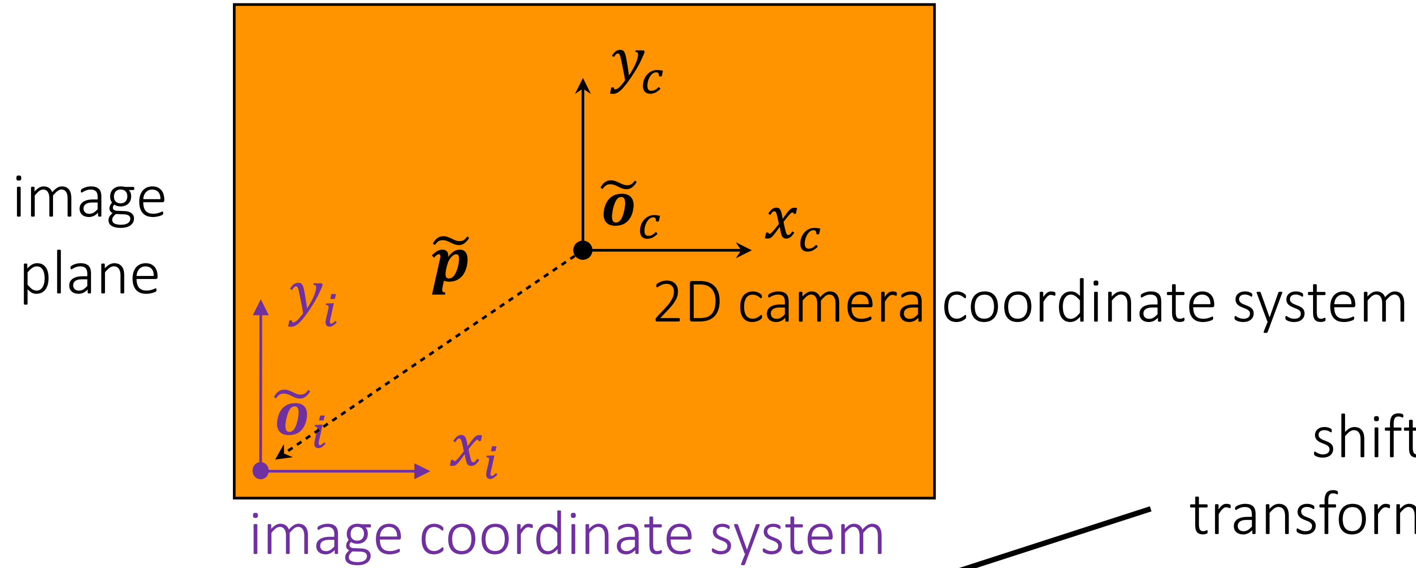
How does the camera matrix change?

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Generalization: image coordinate system

In particular, the camera origin and image origin may be different.

- Can you think of a case when this happens?



shift vector transforming camera origin to image origin

How does the camera matrix change?

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Camera matrix decomposition

We can decompose the camera matrix like this:

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right]$$

What does each part of the matrix represent?

Camera matrix decomposition

We can decompose the camera matrix like this:

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right]$$

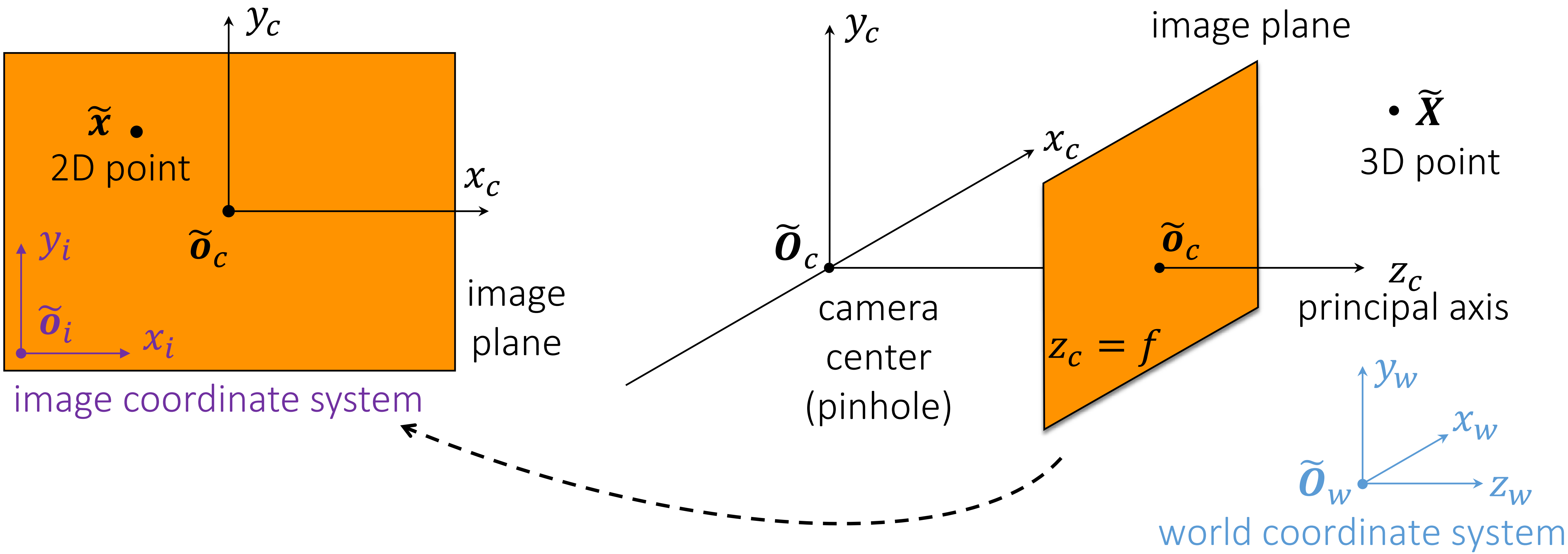
(homogeneous) transformation from 2D to 2D, accounting for non-unit focal length and origin shift

(homogeneous) perspective projection from 3D to 2D, assuming image plane at z = 1 and shared camera/image origin

Also written as:

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\boldsymbol{I} \mid \boldsymbol{0}]$$

Generalizations: coordinate systems

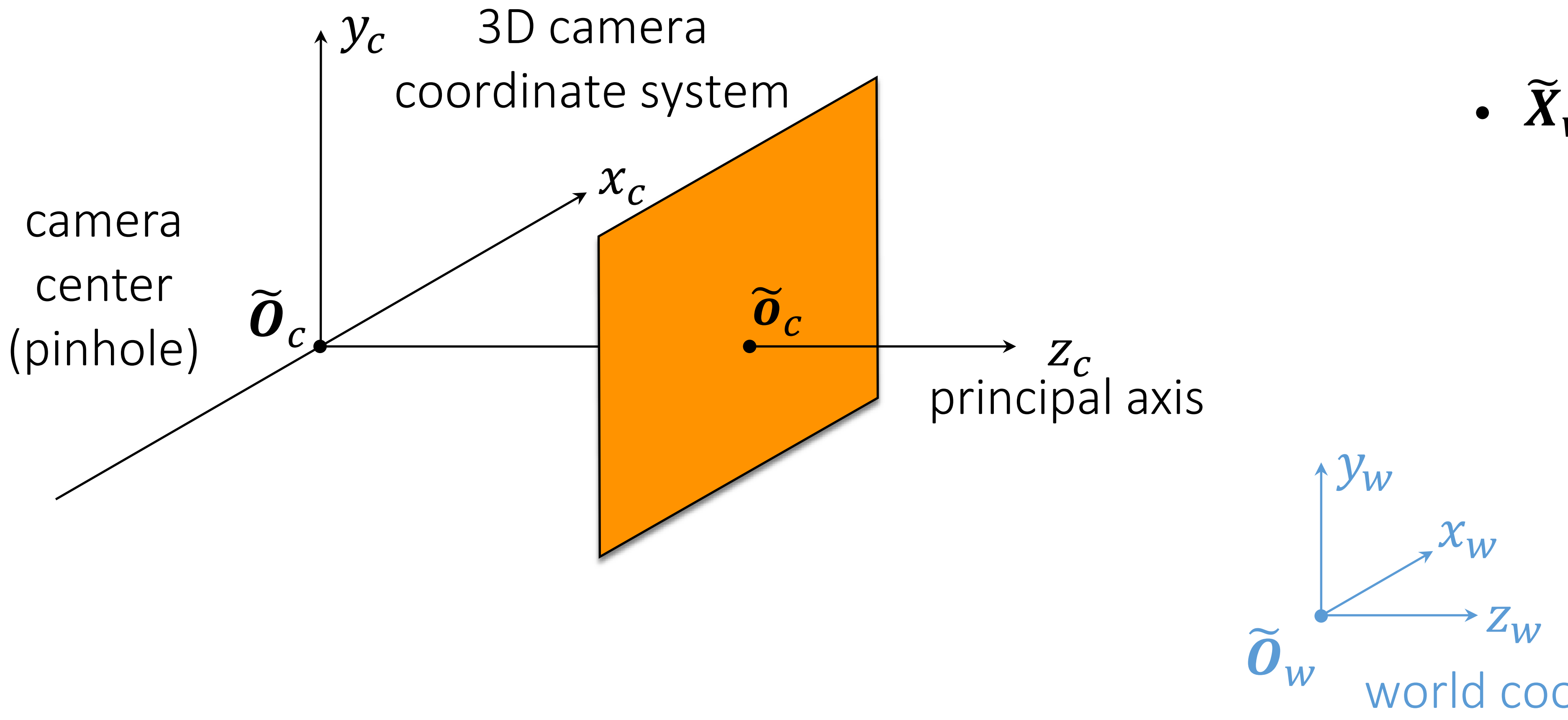


2D camera coordinate system

3D camera coordinate system

- A camera introduces two related coordinate systems, in 3D (world), and in 2D (image plane).
- These coordinate systems may be different from the coordinate systems of our application.

World-to-camera coordinate system transformation

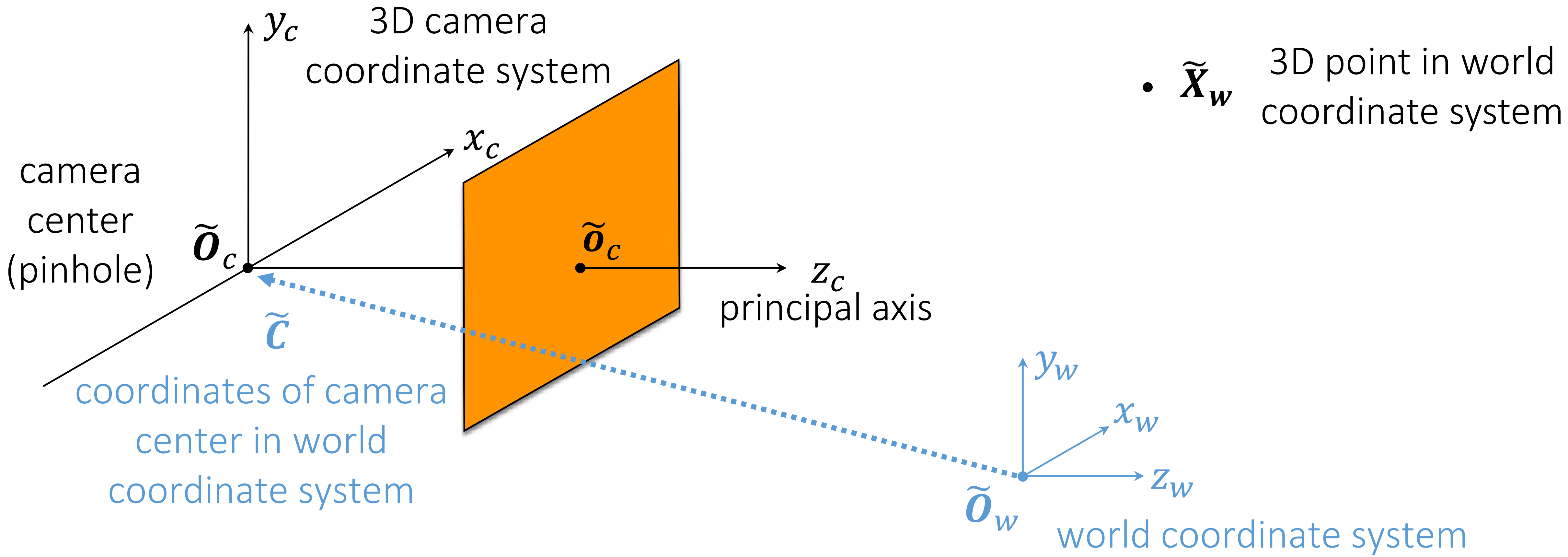


- $\tilde{\boldsymbol{X}}_w$ 3D point in world coordinate system

How do we express $\tilde{\boldsymbol{X}}$ in the 3D camera coordinate system?

$$\tilde{\boldsymbol{X}}_w$$

World-to-camera coordinate system transformation

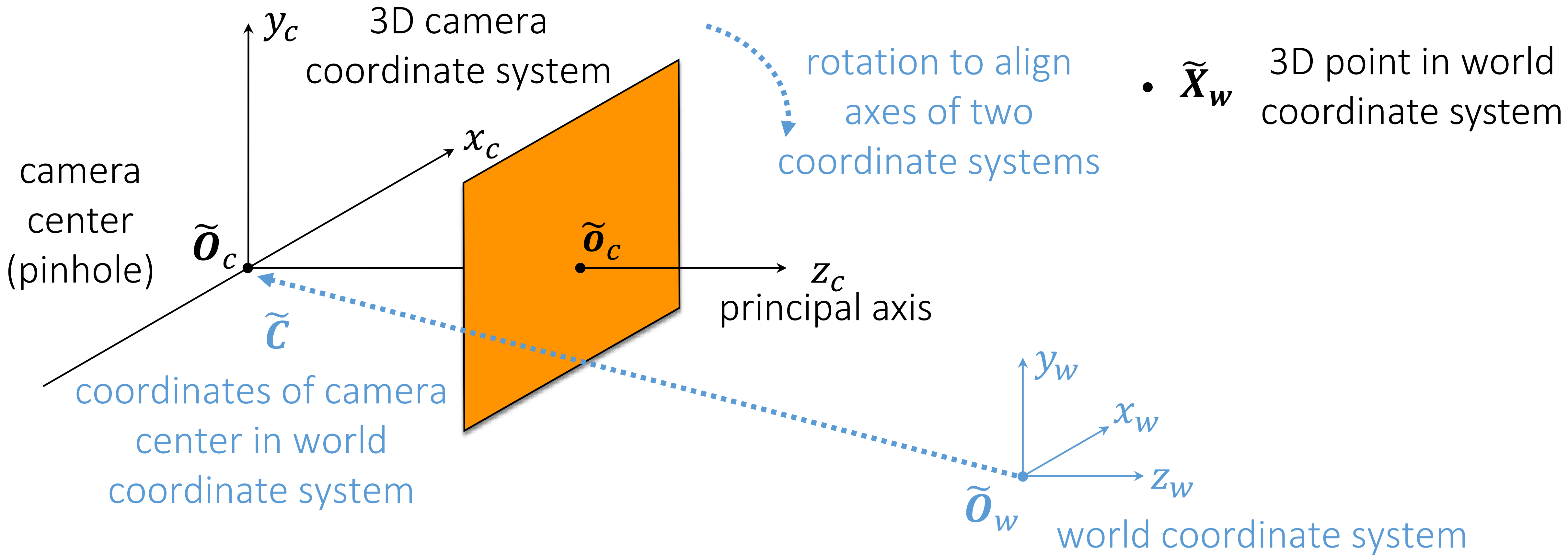


How do we express $\tilde{\boldsymbol{X}}$ in the 3D camera coordinate system?

$$\tilde{\boldsymbol{X}}_w - \tilde{\boldsymbol{C}}$$

translate

World-to-camera coordinate system transformation



How do we express $\tilde{\boldsymbol{X}}$ in the 3D camera coordinate system?

$$\boldsymbol{R} \cdot (\tilde{\boldsymbol{X}}_{\boldsymbol{w}} - \tilde{\boldsymbol{C}})$$

rotate translate

Modeling the 3D coordinate system transformation

In heterogeneous coordinates, we have:

$$\widetilde{\boldsymbol{X}}_{\boldsymbol{c}} = \boldsymbol{R} \cdot \left(\widetilde{\boldsymbol{X}}_{\boldsymbol{w}} - \widetilde{\boldsymbol{C}}\right)$$

How do we write this transformation in homogeneous coordinates?

Modeling the 3D coordinate system transformation

In heterogeneous coordinates, we have:

$$\widetilde{\boldsymbol{X}}_{\boldsymbol{c}} = \boldsymbol{R} \cdot \left(\widetilde{\boldsymbol{X}}_{\boldsymbol{w}} - \widetilde{\boldsymbol{C}}\right)$$

In homogeneous coordinates, we have:

$$\boldsymbol{X}_{\boldsymbol{c}} = \begin{bmatrix} \boldsymbol{R} & -\boldsymbol{R}\widetilde{\boldsymbol{C}} \\ \boldsymbol{0} & 1 \end{bmatrix} \boldsymbol{X}_{\boldsymbol{w}}$$

Incorporating the transform in the camera matrix

The previous camera matrix is for homogeneous 3D coordinates in camera coordinate system:

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X_c} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\boldsymbol{I} \mid \boldsymbol{0}]\boldsymbol{X_c}$$

We also just derived:

$$\boldsymbol{X_c} = \begin{bmatrix} \boldsymbol{R} & -\boldsymbol{R}\widetilde{\boldsymbol{C}} \\ \boldsymbol{0} & 1 \end{bmatrix} \boldsymbol{X_w}$$

Putting it all together

We can write everything into a single projection:

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}_{\boldsymbol{w}}$$

The camera matrix now looks like:

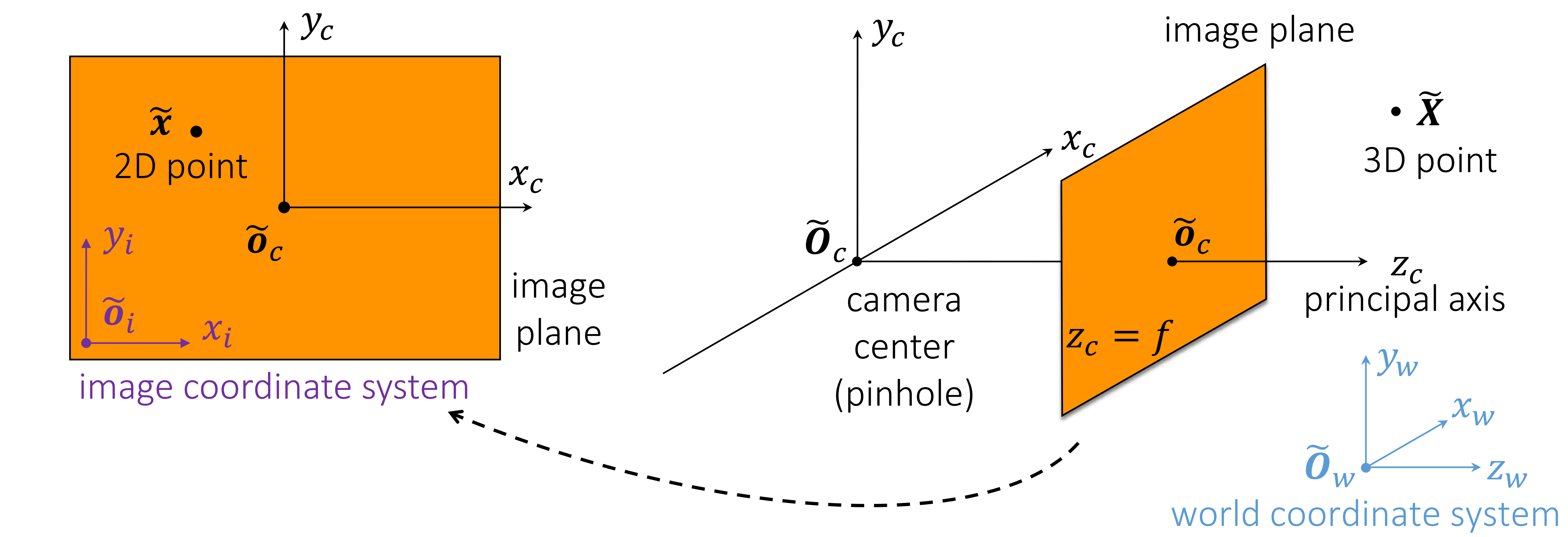
$$\boldsymbol{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\boldsymbol{I} \mid \boldsymbol{0}] \begin{bmatrix} \boldsymbol{R} & -\boldsymbol{R}\widetilde{\boldsymbol{C}} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

intrinsic parameters (3 x 3): correspond to camera internals (2D image-to-image transformation)

perspective projection (3 x 4): maps 3D to 2D points (camera-to-image transformation)

extrinsic parameters (4 x 4): correspond to camera externals (3D world-to-camera transformation)

Generalizations: coordinate systems



2D camera coordinate system

3D camera coordinate system

- A camera introduces two related coordinate systems, in 3D (world), and in 2D (image plane).
- These coordinate systems may be different from the coordinate systems of our application.

Putting it all together

We can write everything into a single projection:

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X}_{\boldsymbol{w}}$$

The camera matrix now looks like:

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\boldsymbol{R} \mid -\boldsymbol{R}\widetilde{\boldsymbol{C}}]$$

It is common to combine the perspective projection and extrinsics in one matrix.

intrinsic parameters (3 x 3): correspond to camera internals (2D image-to-image transformation)

extrinsic parameters (3 x 4): correspond to camera externals (3D world-to-camera transformation) *and* perspective projection

The pinhole camera matrix

More compactly, we can write the pinhole camera matrix as:

$$\boldsymbol{P} = \boldsymbol{K}[\boldsymbol{R} \mid \boldsymbol{t}]$$

where

$$\underbrace{\boldsymbol{K} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}}_{\text{intrinsic parameters}}$$

2D Euclidean transform

$$\boldsymbol{R} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \qquad \boldsymbol{t} = -\boldsymbol{R}\widetilde{\boldsymbol{C}} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

3D rotation

3D translation

intrinsic parameters

extrinsic parameters

More general pinhole camera matrices

The following is the standard pinhole camera matrix we saw.

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

How many degrees of freedom does this matrix have?

More general pinhole camera matrices

The following is the standard pinhole camera matrix we saw.

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

How many degrees of freedom does this matrix have?

- 9 degrees of freedom (3 for intrinsics, 3 for rotation, 3 for translation).

More general pinhole camera matrices

The following is the standard pinhole camera matrix we saw.

$$\boldsymbol{P} = \boxed{\begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

How many degrees of freedom does this matrix have?

- 9 degrees of freedom (3 for intrinsics, 3 for rotation, 3 for translation).

We can get more general pinhole cameras with more degrees of freedom by generalizing the intrinsics matrix, while leaving everything else the same..

More general pinhole camera matrices

CCD camera: pixels may not be square.

$$\boldsymbol{P} = \begin{bmatrix} a_x & 0 & p_x \\ 0 & a_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

How many degrees of freedom does this matrix have?

More general pinhole camera matrices

CCD camera: pixels may not be square.

$$\boldsymbol{P} = \begin{bmatrix} a_x & 0 & p_x \\ 0 & a_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

How many degrees of freedom does this matrix have?

- 10 degrees of freedom (4 for intrinsics, 3 for rotation, 3 for translation).

More general pinhole camera matrices

Finite projective camera: sensor may be skewed.

$$\boldsymbol{P} = \begin{bmatrix} a_x & s & p_x \\ 0 & a_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

How many degrees of freedom does this matrix have?

More general pinhole camera matrices

Finite projective camera: sensor may be skewed.

$$\boldsymbol{P} = \begin{bmatrix} a_x & s & p_x \\ 0 & a_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

How many degrees of freedom does this matrix have?

- 11 degrees of freedom (5 for intrinsics, 3 for rotation, 3 for translation).

Can we get a *perspective projection* camera with more degrees of freedom?

More general pinhole camera matrices

Finite projective camera: sensor may be skewed.

The finite projective camera is the most general camera implementing perspective projection.

$$\boldsymbol{P} = \begin{bmatrix} a_x & s & p_x \\ 0 & a_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

How many degrees of freedom does this matrix have?

- 11 degrees of freedom (5 for intrinsics, 3 for rotation, 3 for translation).

Can we get a *perspective projection* camera with more degrees of freedom?

- No, as the entire camera matrix $\boldsymbol{P}$ has 12 elements (3x4) and is defined up to scale.

More general pinhole camera matrices

Finite projective camera: sensor may be skewed.

The finite projective camera is the most general camera implementing perspective projection.

$$\boldsymbol{P} = \begin{bmatrix} a_x & s & p_x \\ 0 & a_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \boxed{\left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right]} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

How many degrees of freedom does this matrix have?

- 11 degrees of freedom (5 for intrinsics, 3 for rotation, 3 for translation).

Can we get a *perspective projection* camera with more degrees of freedom?

- No, as the entire camera matrix $\boldsymbol{P}$ has 12 elements (3x4) and is defined up to scale.

Perspective distortion

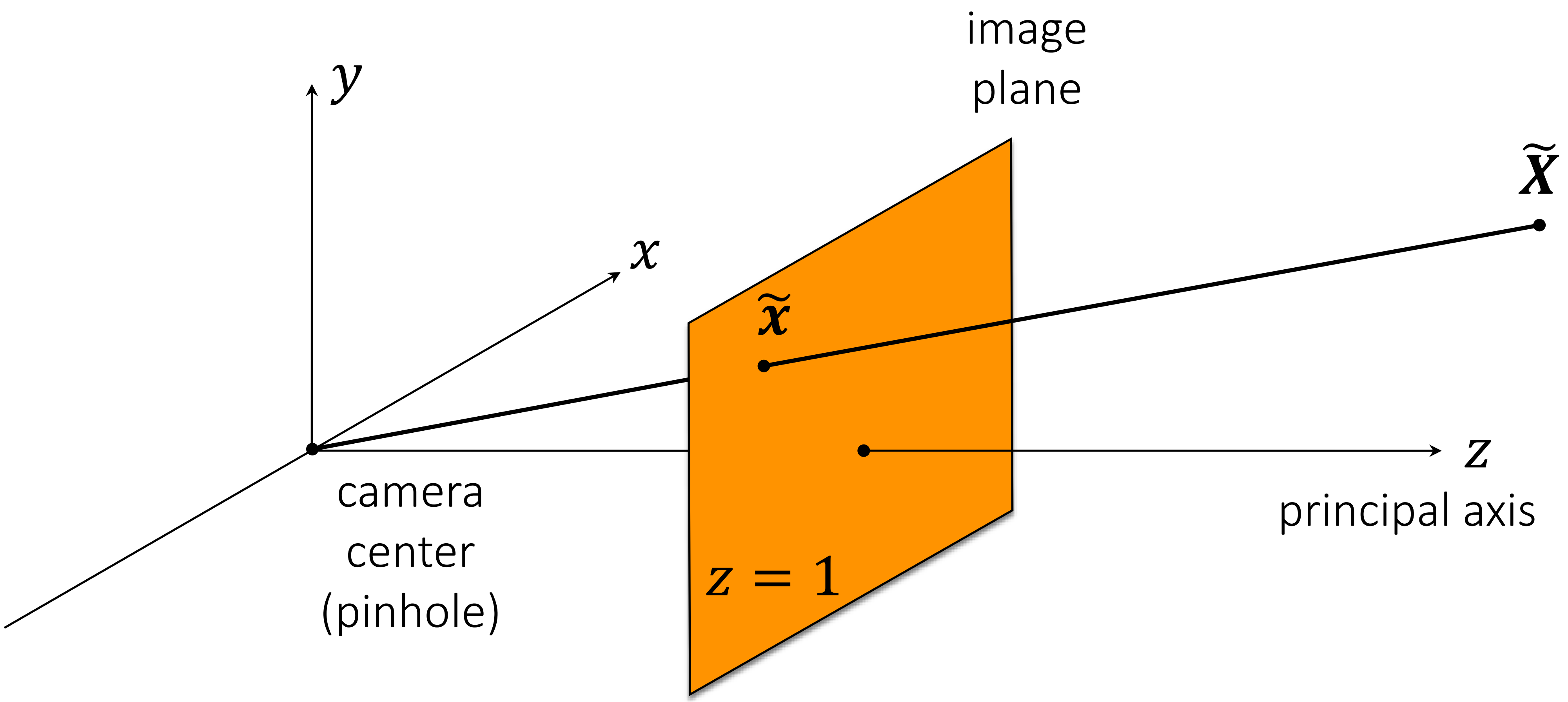
Finite projective camera

Let's ignore intrinsics and extrinsics for now.

$$\boldsymbol{P} = \begin{bmatrix} a_x & s & p_x \\ 0 & a_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

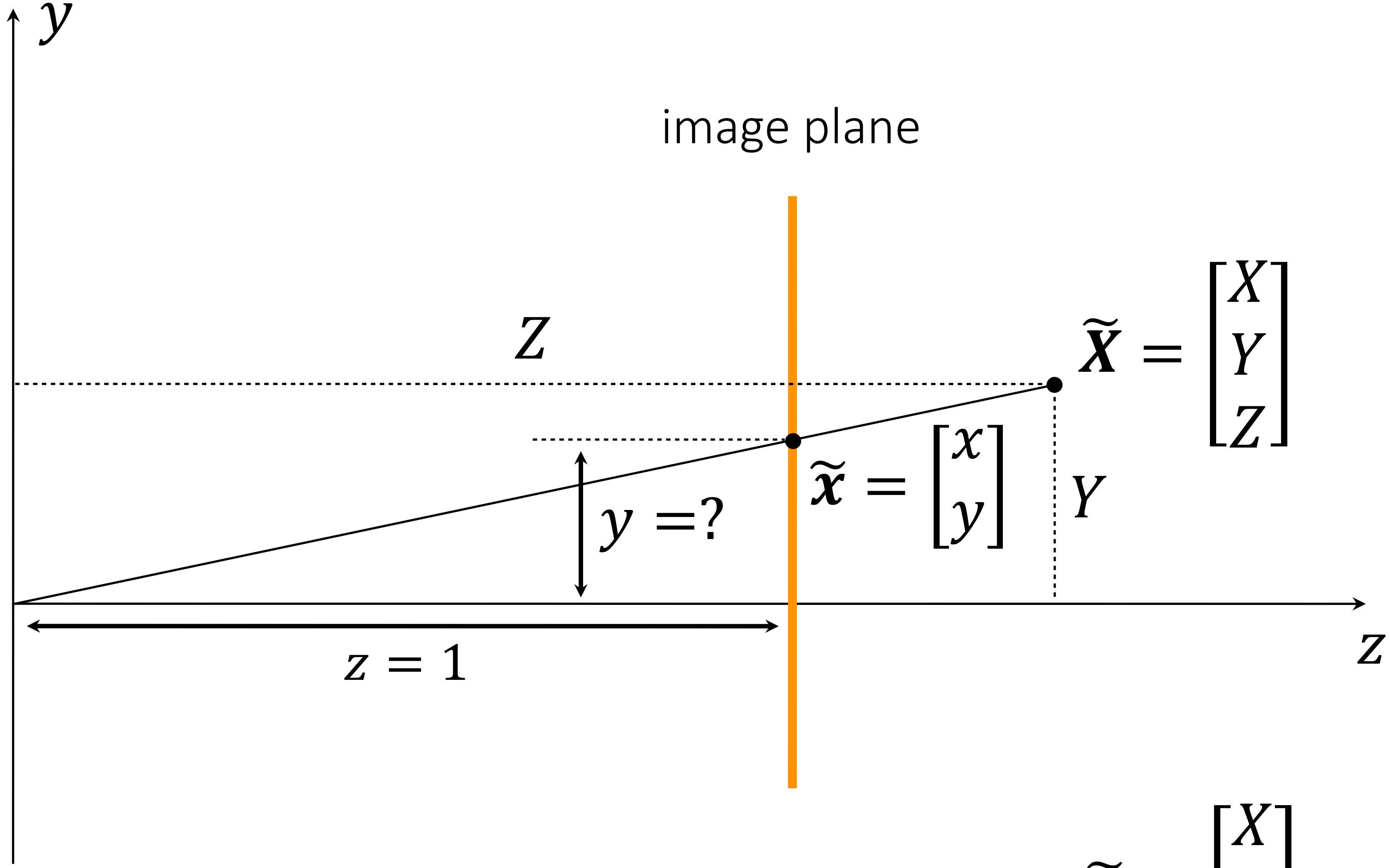
What is the effect of the perspective projection matrix?

The (rearranged) pinhole camera



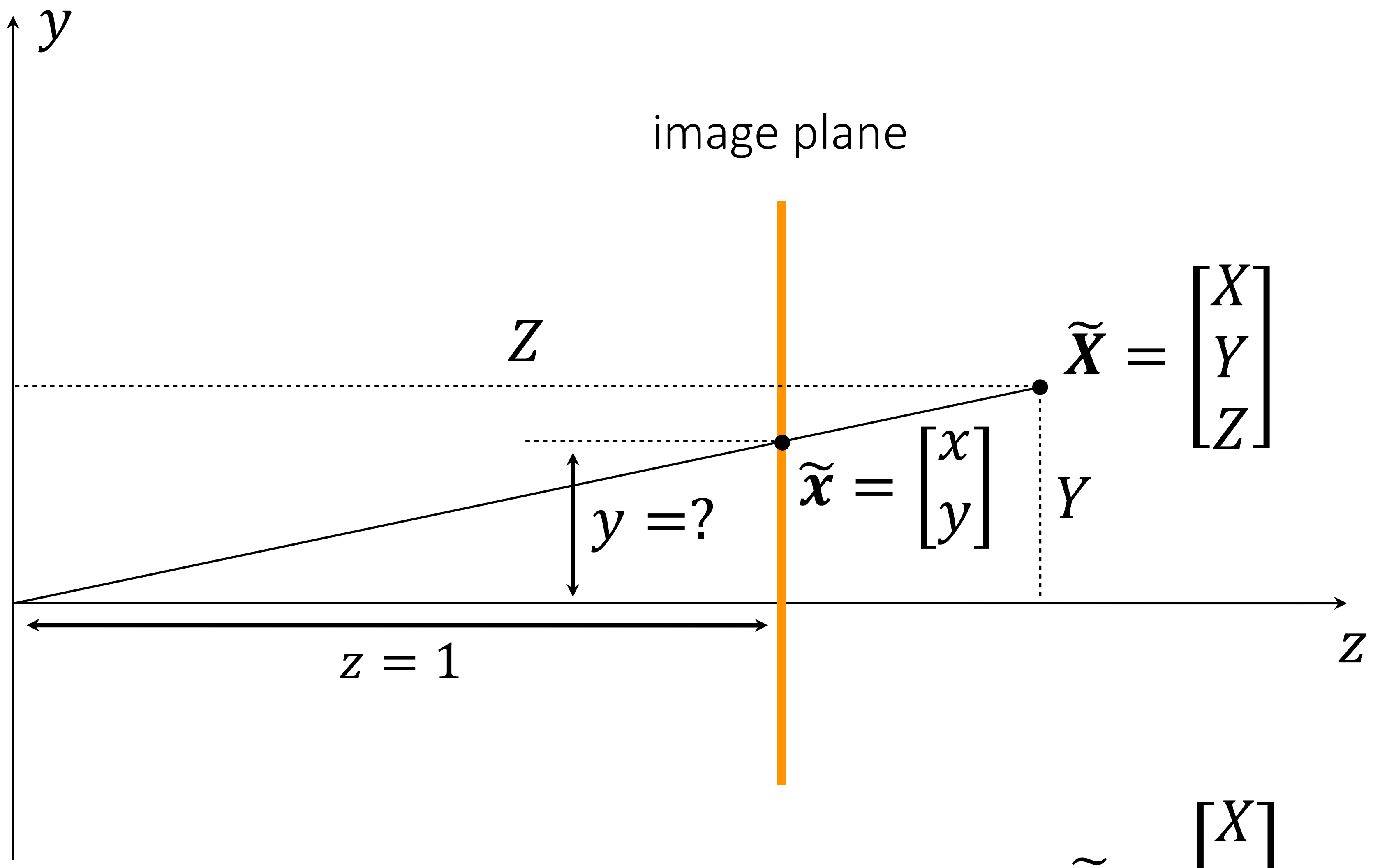
What is the equation for image coordinate $\tilde{\boldsymbol{x}}$ in terms of $\tilde{\boldsymbol{X}}$?

The 2D view of the (rearranged) pinhole camera



Using similar triangles: $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$

The 2D view of the (rearranged) pinhole camera



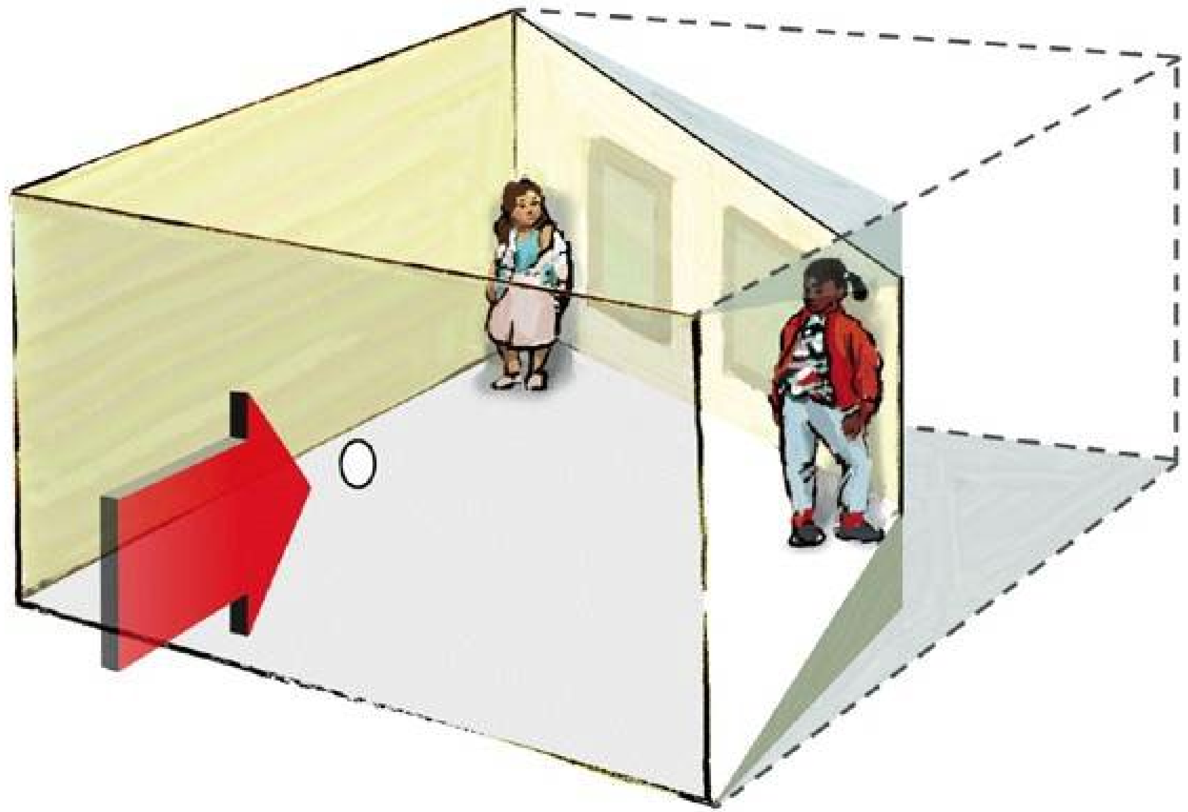
Perspective distortion: magnification changes with depth

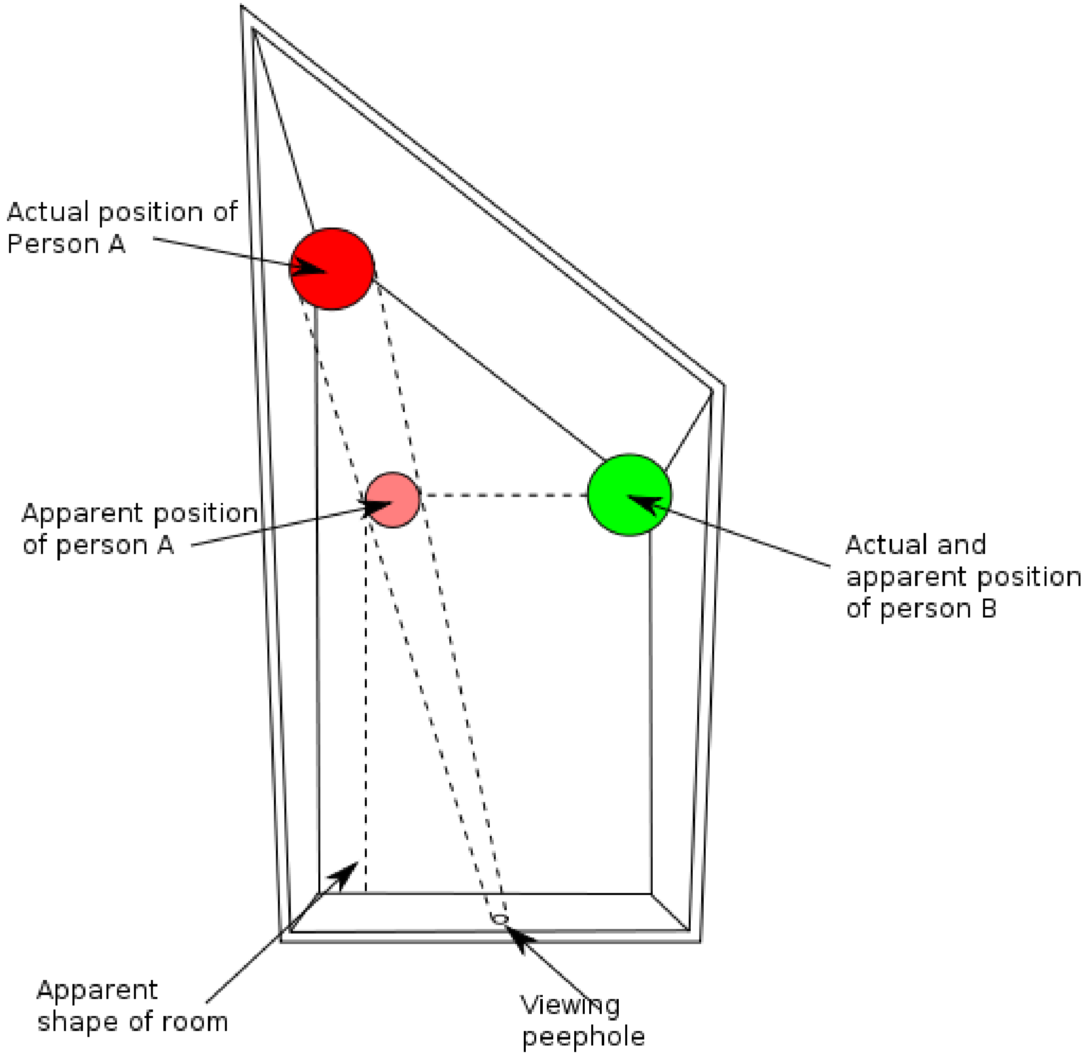
Using similar triangles: $\widetilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \widetilde{\boldsymbol{x}} = \begin{bmatrix} X/\boxed{Z} \\ Y/\boxed{Z} \end{bmatrix}$

Forced perspective

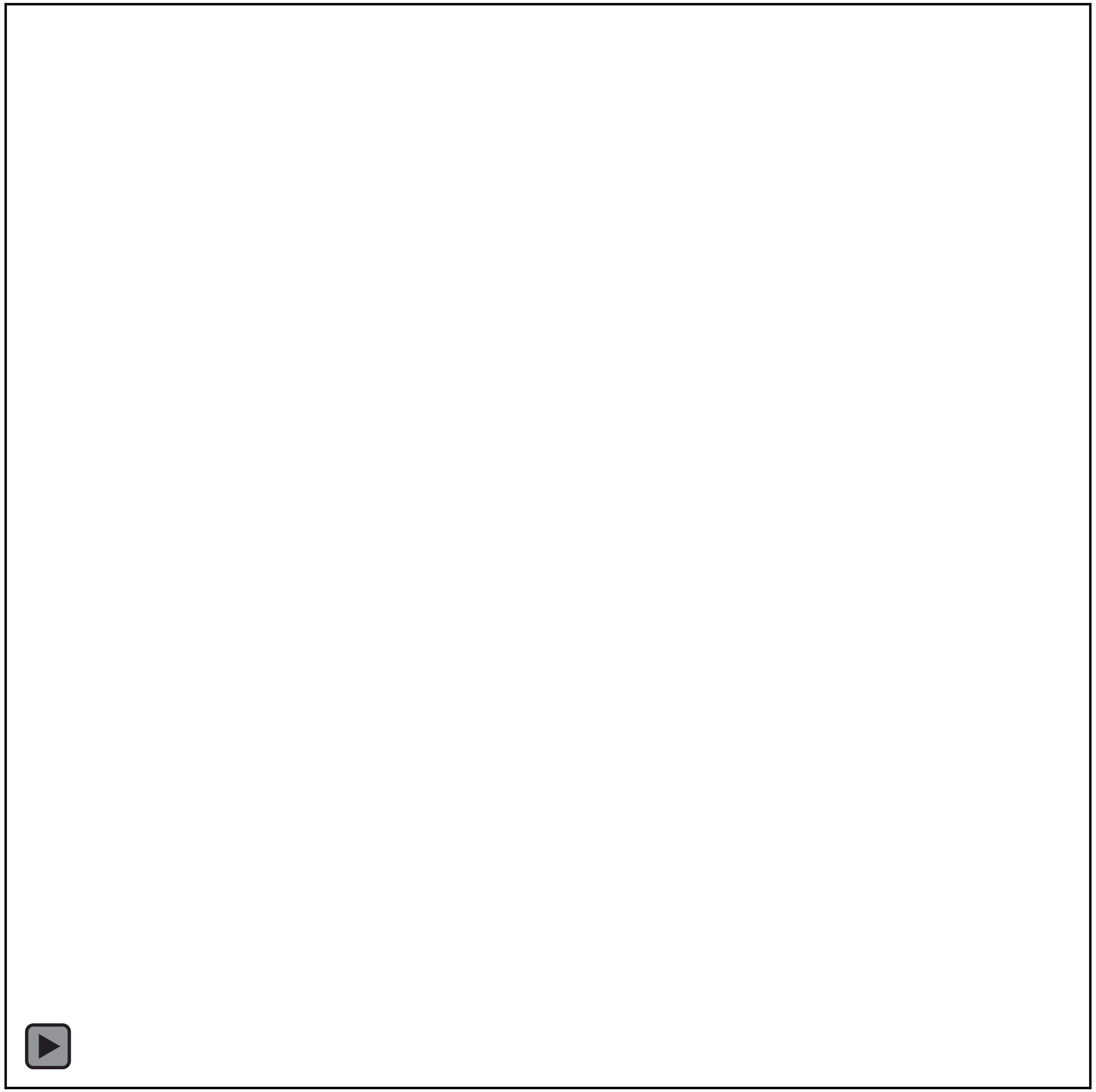
The Ames room illusion

The Ames room illusion





The arrow illusion

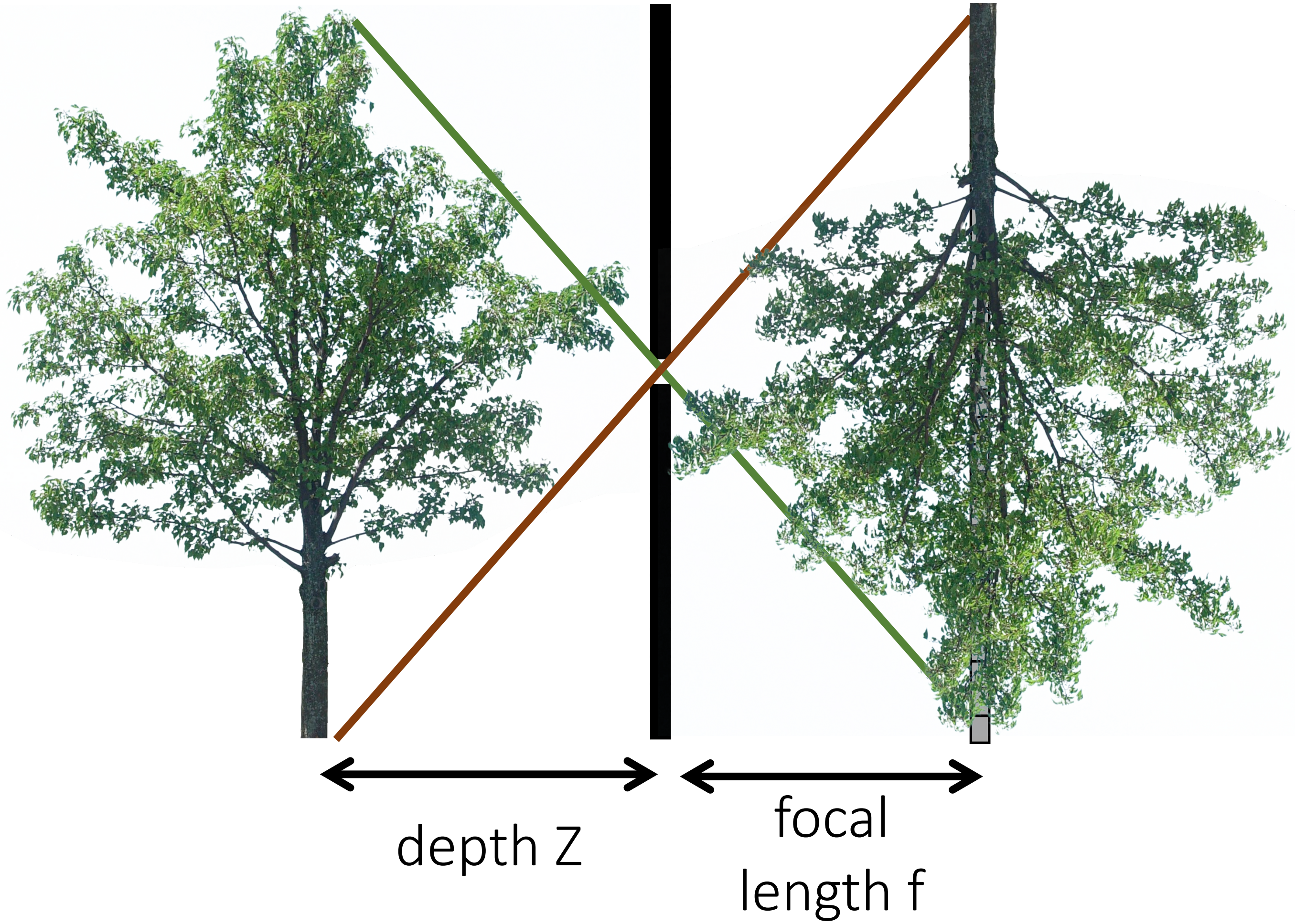


Is there a camera without perspective distortion?

Other camera models

What if…

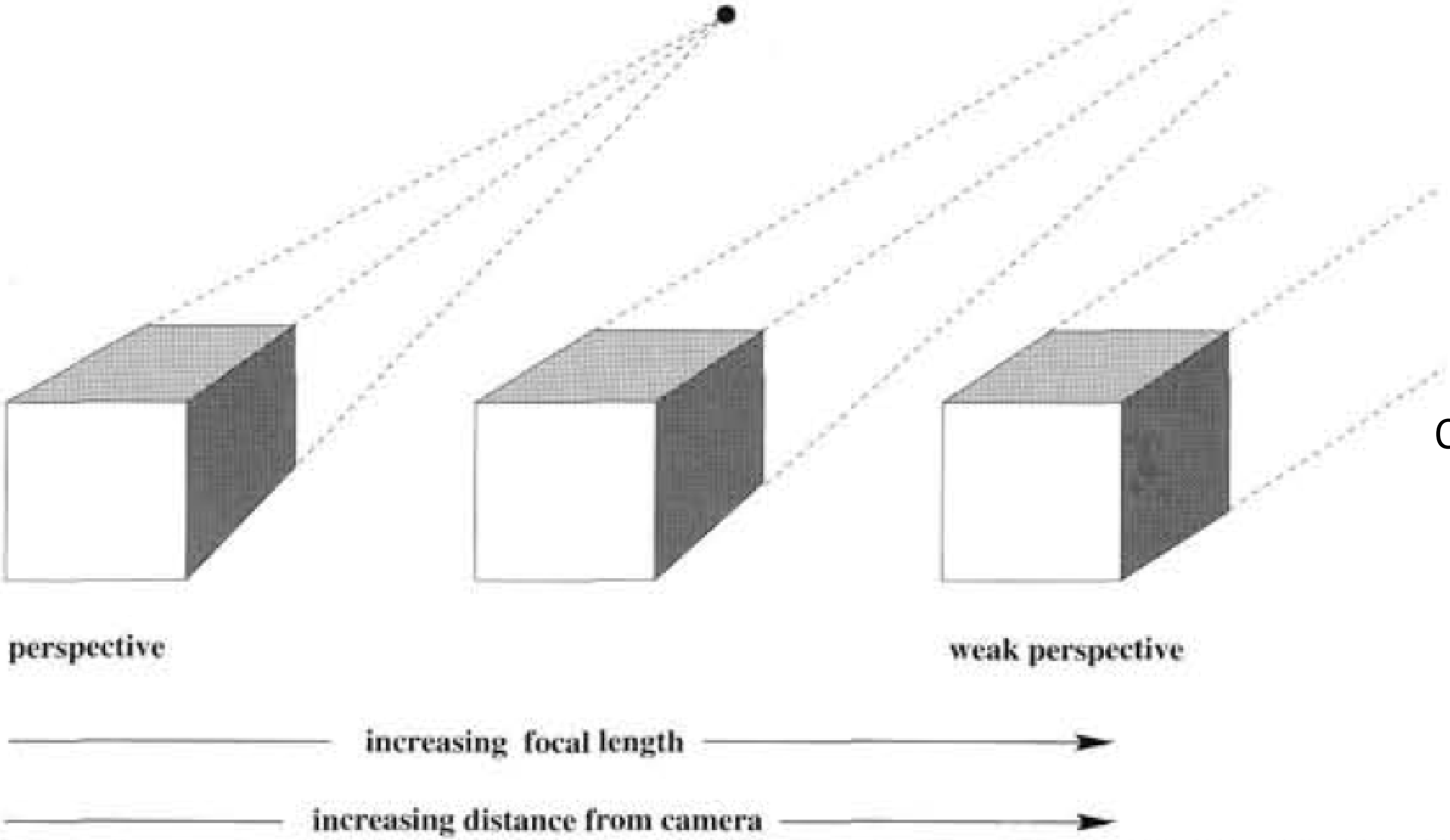
real-world object



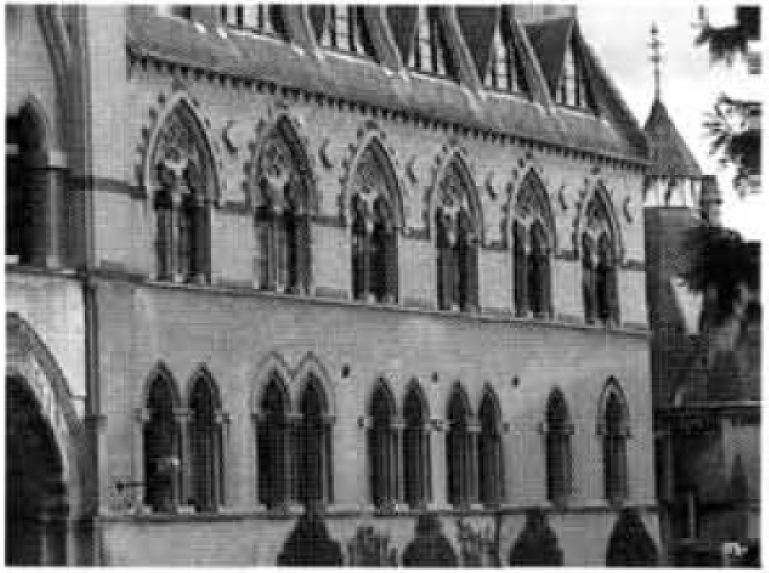
… we continue increasing Z and f while maintaining same magnification?

$$f \to \infty \text{ and } \frac{f}{Z} = \text{constant}$$

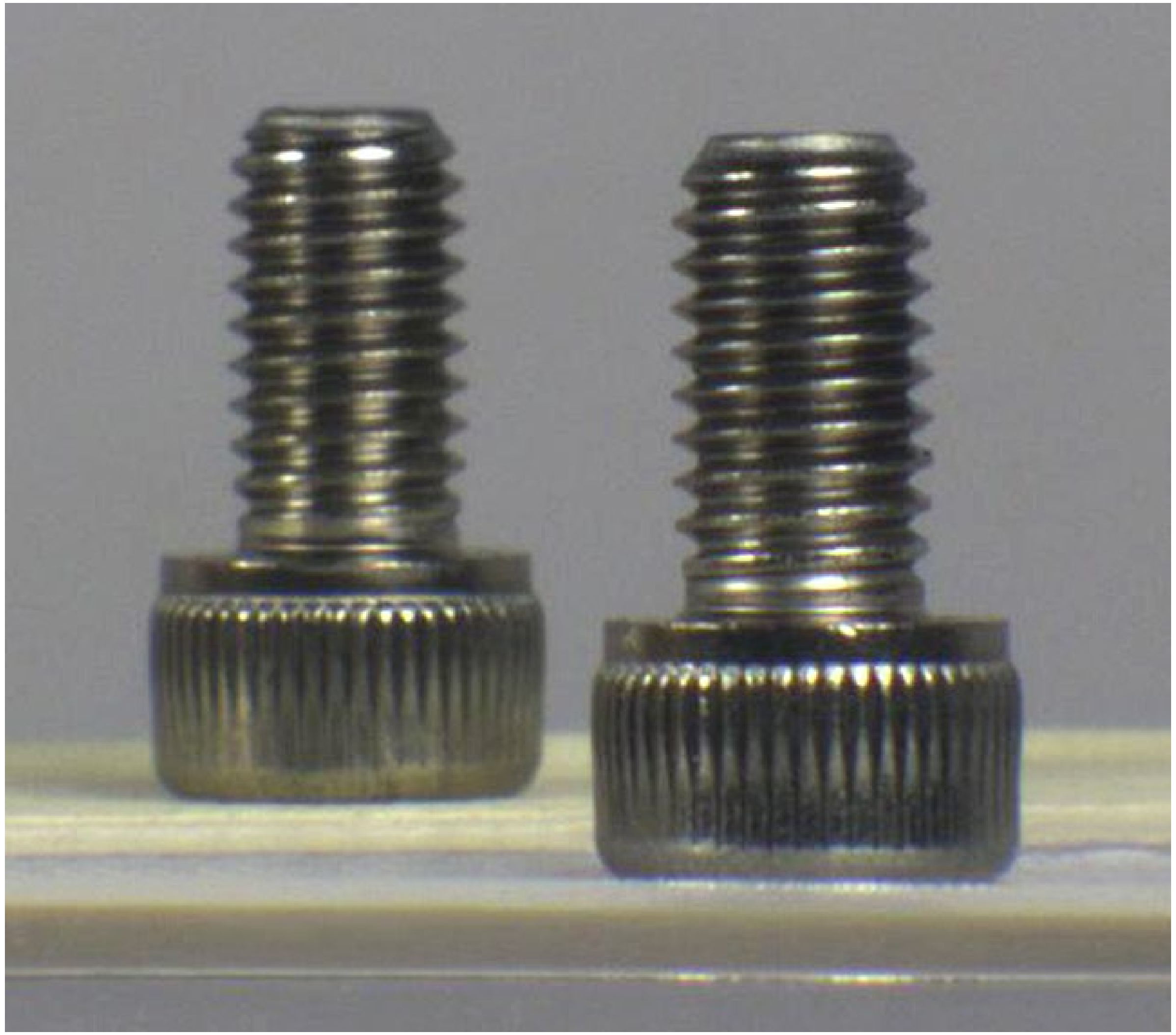
Perspective camera: camera is *close* to object and has *small* focal length



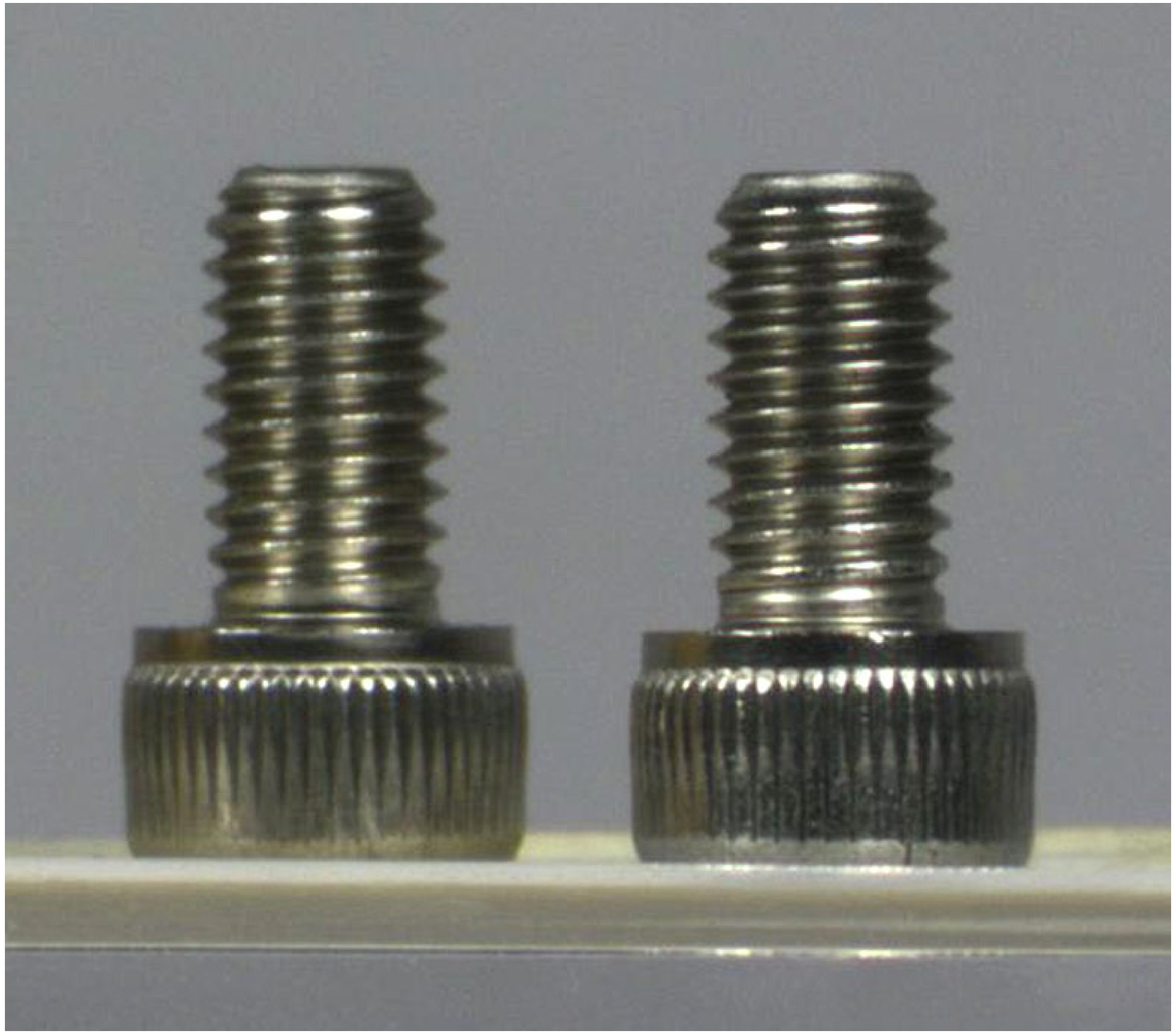
Weak-perspective camera: camera is *far* from object and has *large* focal length



Different cameras

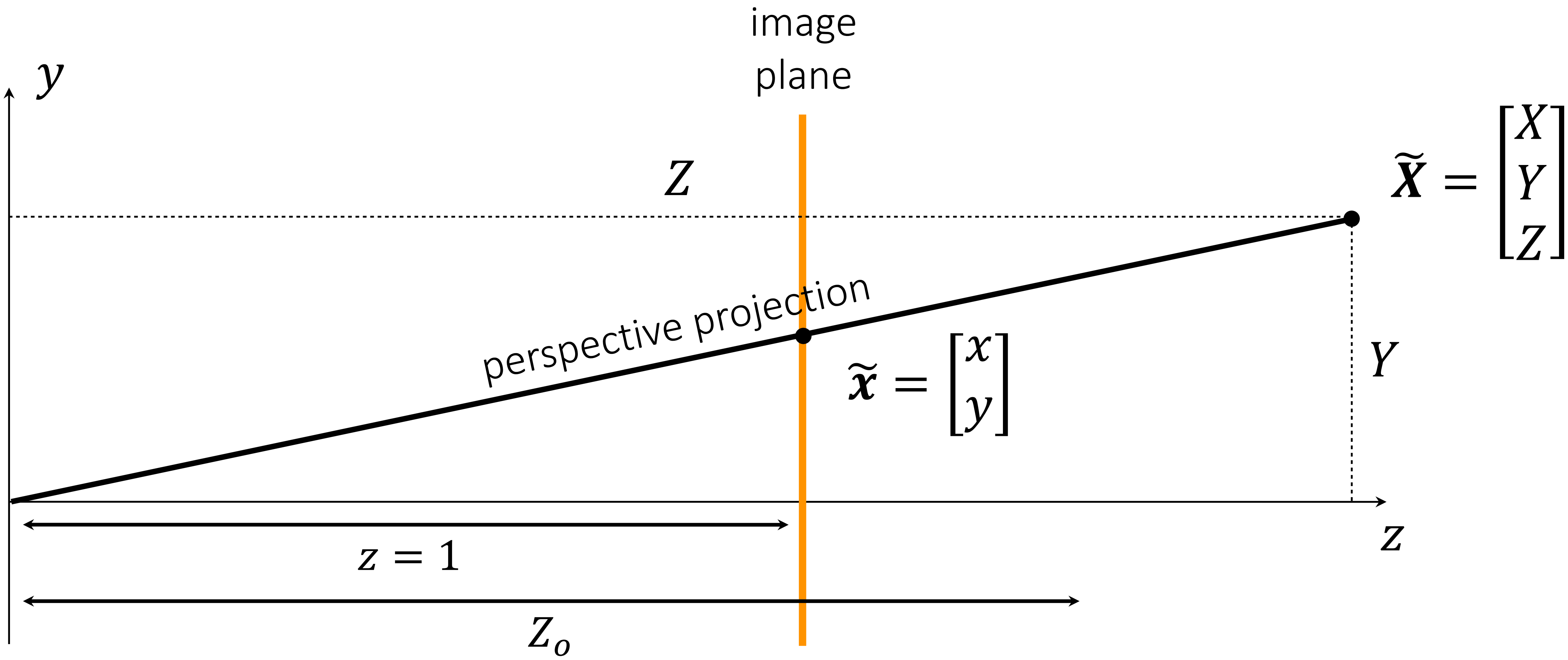


perspective camera



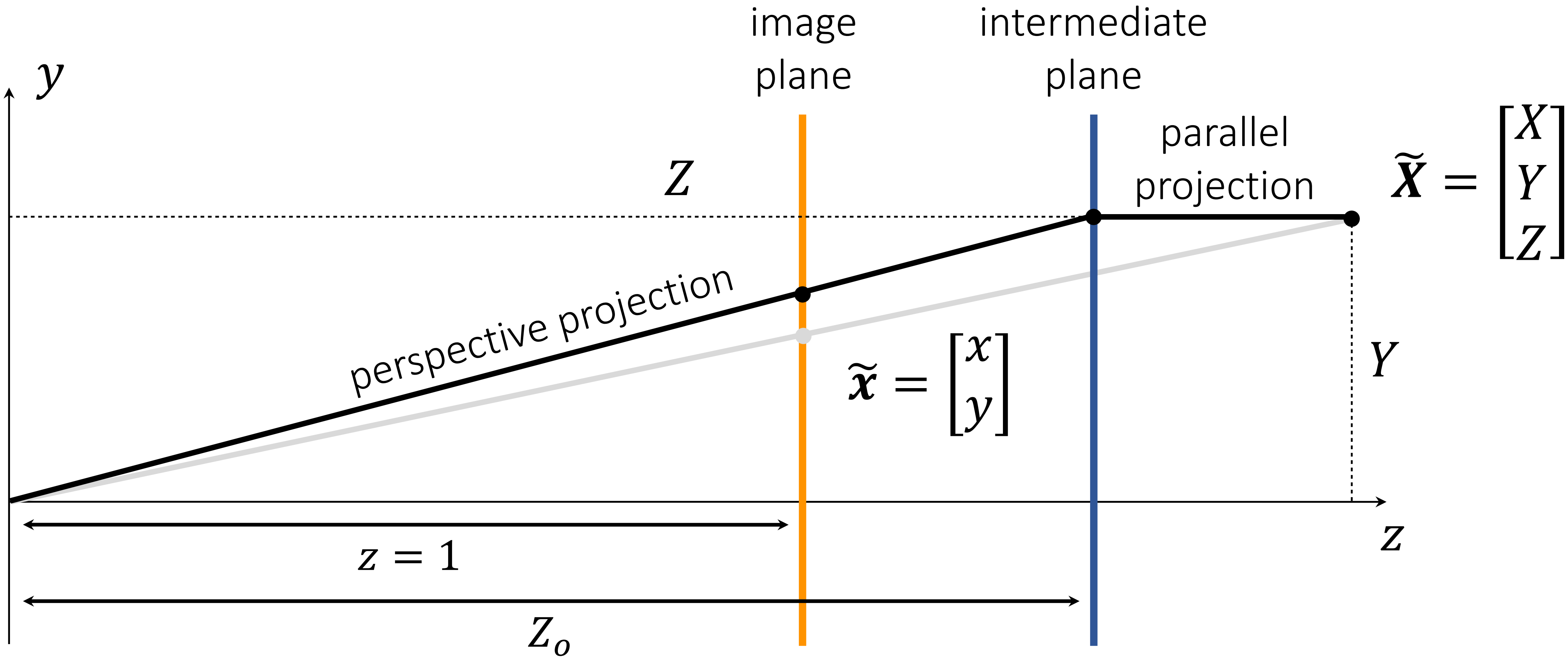
weak perspective camera

Perspective versus weak-perspective camera



perspective projection $\widetilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \widetilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$

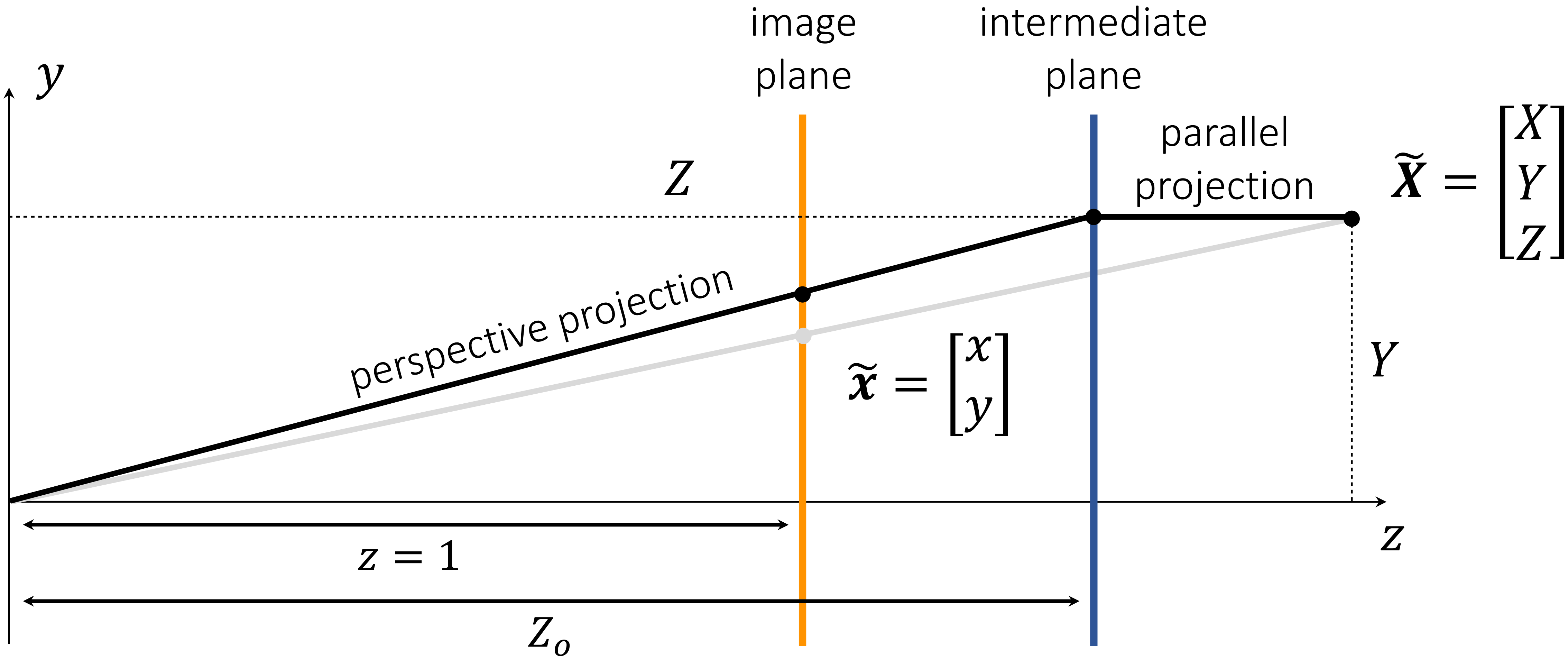
Perspective versus weak-perspective camera



perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$

weak-perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = ?$

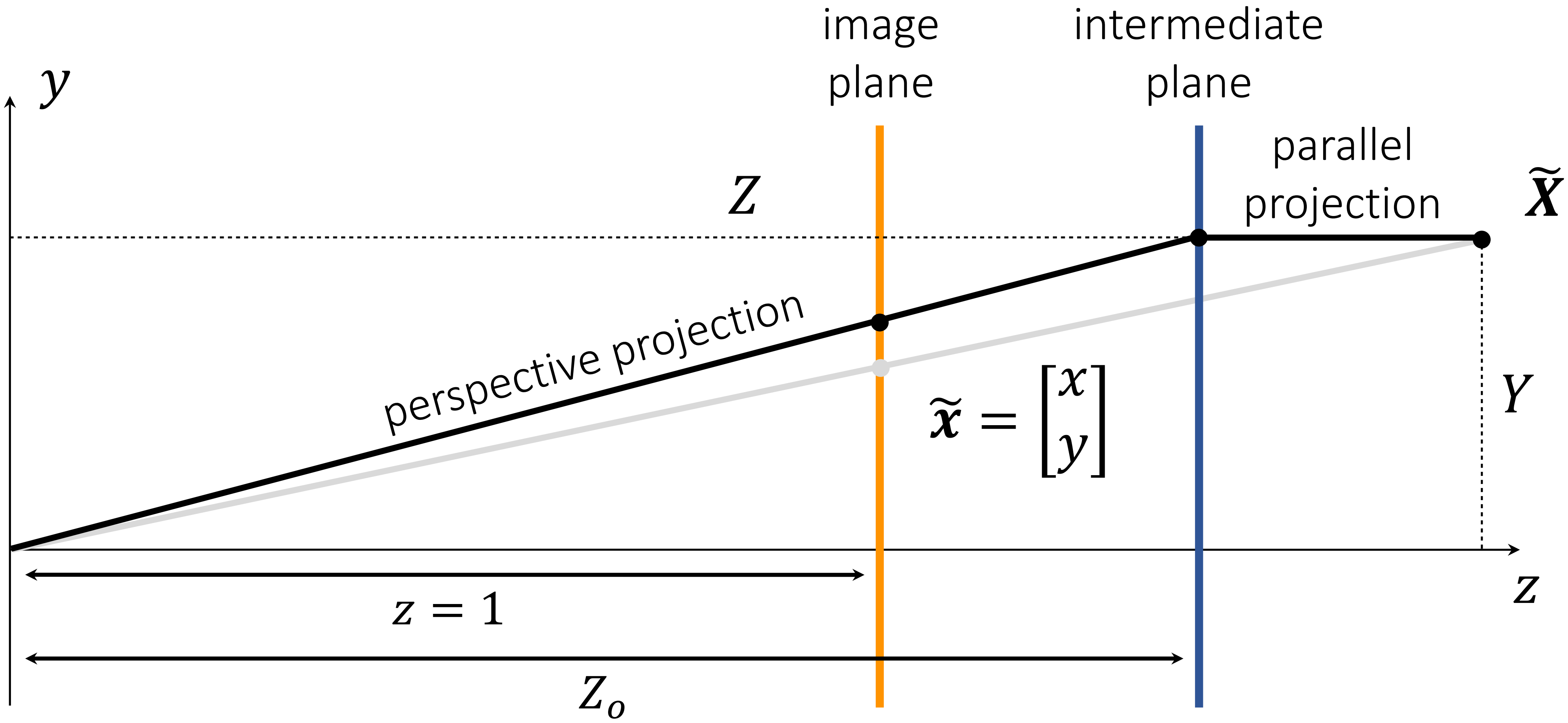
Perspective versus weak-perspective camera



perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$

weak-perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/Z_o \\ Y/Z_o \end{bmatrix}$

Perspective versus weak-perspective camera



magnification *independent* of depth, depends only on Z_o

perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$

weak-perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/\boxed{Z_o} \\ Y/\boxed{Z_o} \end{bmatrix}$

Comparing camera projection matrices

Let's ignore intrinsics and extrinscis for now.

- The *perspective projection matrix* can be written as:

$$\boldsymbol{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- What would the *weak-perspective projection matrix* look like?

Comparing camera projection matrices

Let's ignore intrinsics and extrinscis for now.

- The *perspective projection matrix* can be written as:

$$\boldsymbol{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- The *weak-perspective projection matrix* can be written as:

$$\boldsymbol{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & Z_o \end{bmatrix}$$

Comparing camera matrices

Let's now incorporate intrinsics and extrinsics.

- The *finite projective camera matrix* can be written as:

$$\boldsymbol{P} = \boldsymbol{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

- What would the matrix of the so-called *affine camera* look like?

Comparing camera matrices

Let's now incorporate intrinsics and extrinsics.

- The *finite projective camera matrix* can be written as:

$$\boldsymbol{P} = \boldsymbol{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

- The *affine camera matrix* can be written as:

$$\boldsymbol{P} = \boldsymbol{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & Z_o \end{bmatrix} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

Change only the projection matrix, and use the exact same matrices for intrinsics and extrinsics.

Special case: orthographic projection

Let's now incorporate intrinsics and extrinsics.

- The *finite projective camera matrix* can be written as:

$$\boldsymbol{P} = \boldsymbol{K}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

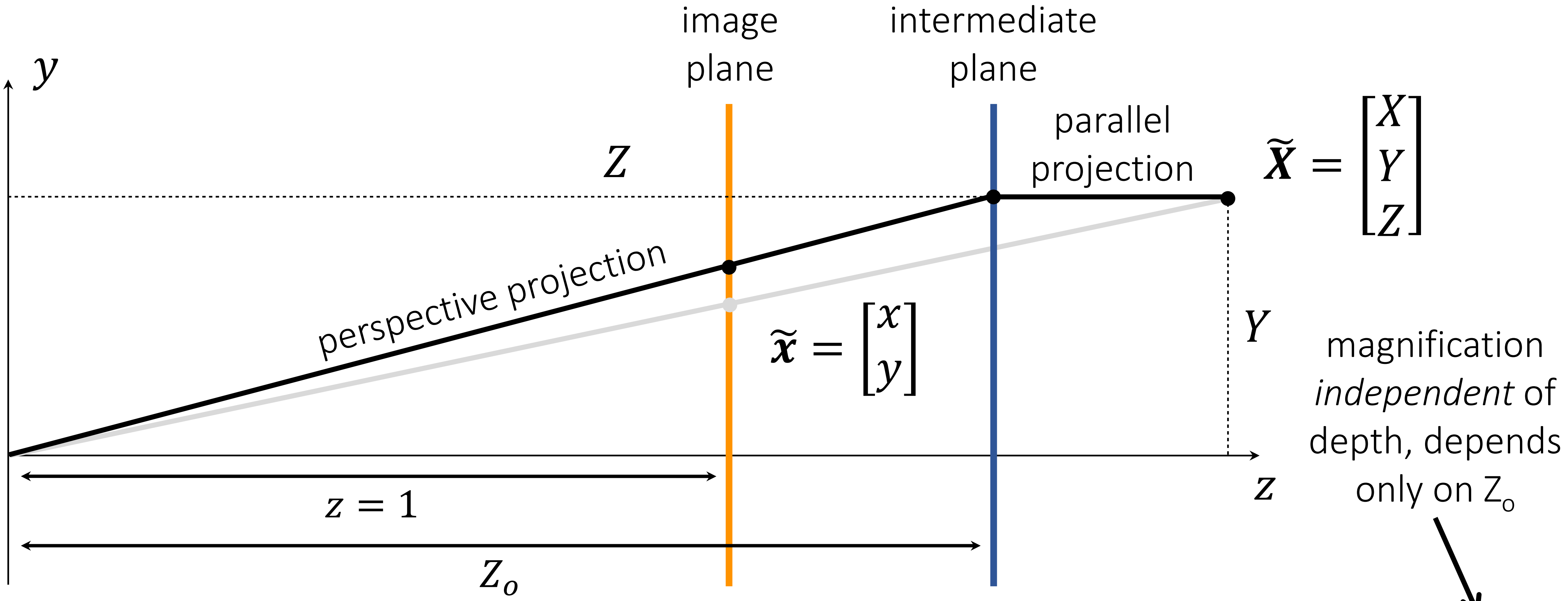
Change only the projection matrix, and use the exact same matrices for intrinsics and extrinsics.

- The *affine camera matrix* can be written as:

What's the effect of setting $Z_o = 1$?

$$\boldsymbol{P} = \boldsymbol{K}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \boxed{1} \end{bmatrix}\begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

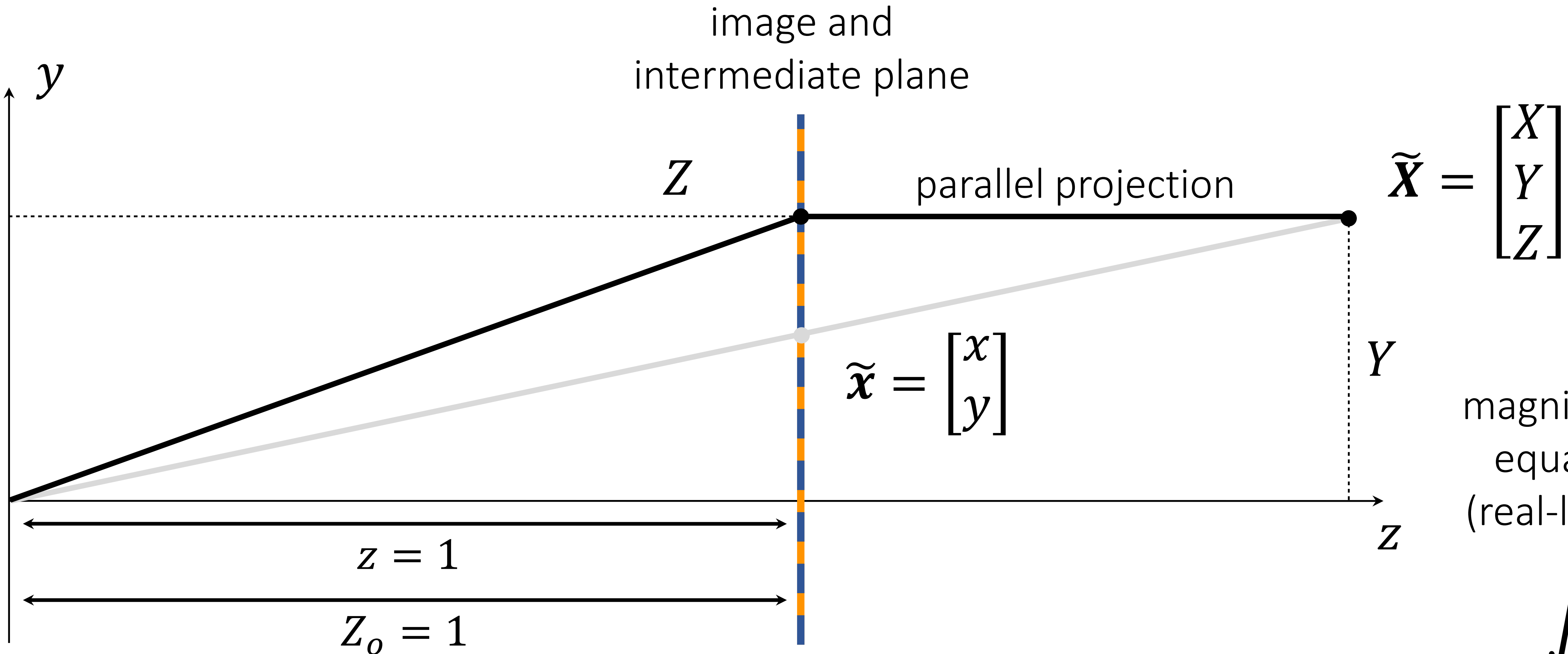
Perspective versus weak-perspective camera



perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$

weak-perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/Z_o \\ Y/Z_o \end{bmatrix}$

Perspective versus orthographic camera



magnification equal to 1 (real-life size)

perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$

weak-perspective projection $\tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\boldsymbol{x}} = \begin{bmatrix} X \\ Y \end{bmatrix}$

When can we assume a weak-perspective camera?

When can we assume a weak-perspective camera?

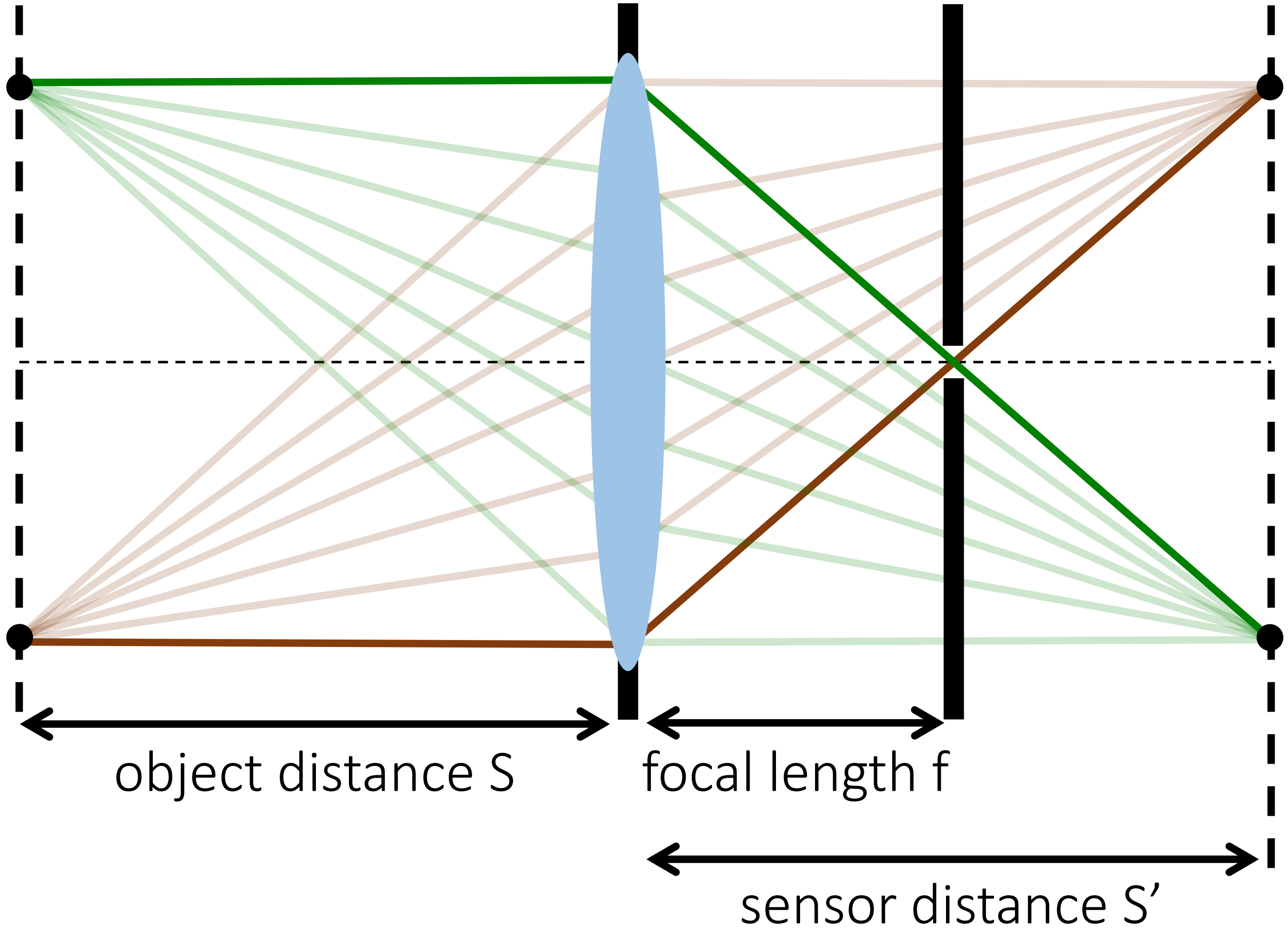
1. When the scene (or parts of it) is very far away.

Weak-perspective projection applies to the mountains.

When can we assume a weak-perspective camera?

2. When we use a telecentric lens.

Place a pinhole at focal length, so that only rays parallel to primary ray pass through.

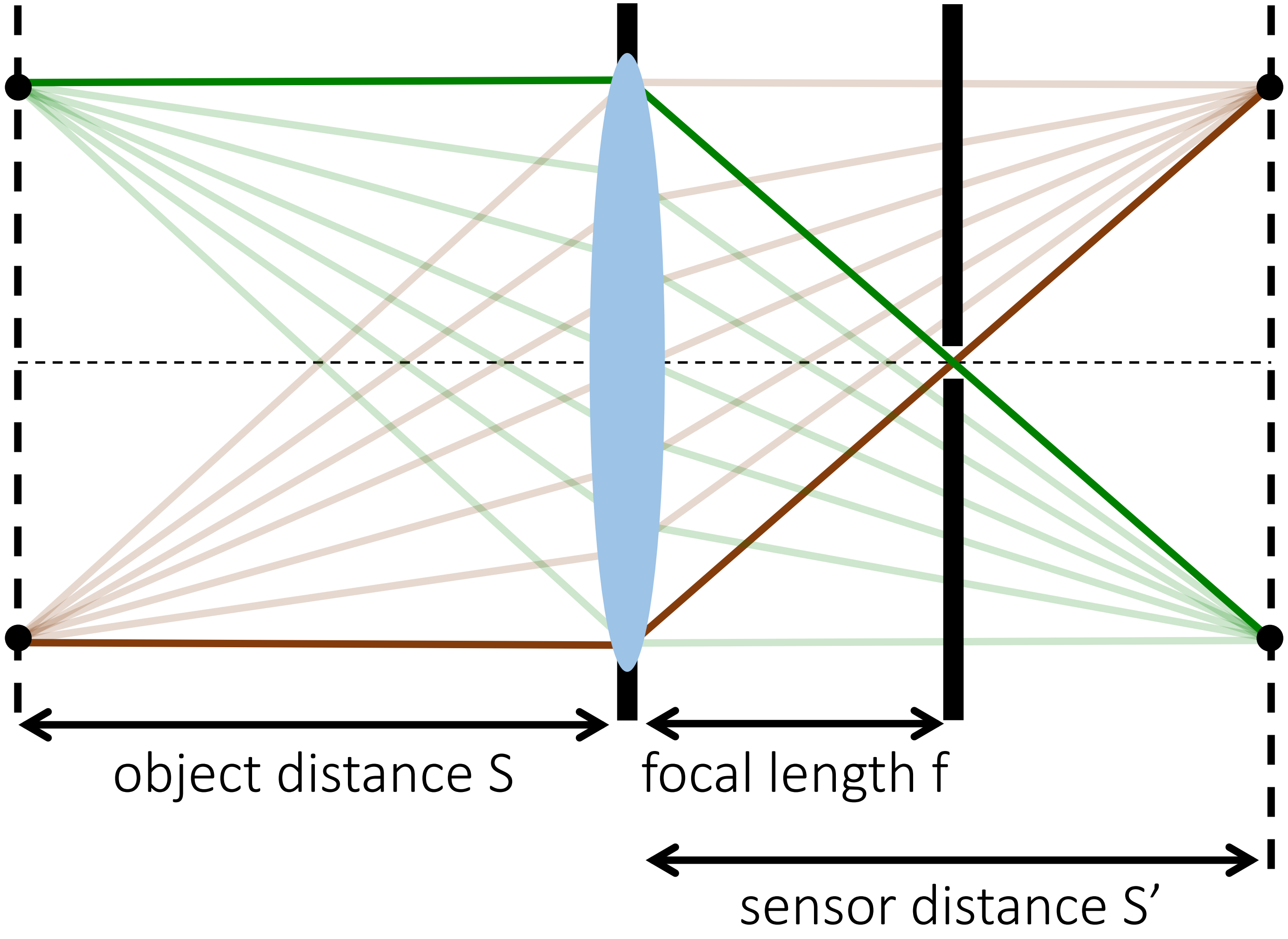


What does Z_o equal in this case?

When can we assume a weak-perspective camera?

2. When we use a telecentric lens.

Place a pinhole at focal length, so that only rays parallel to primary ray pass through.

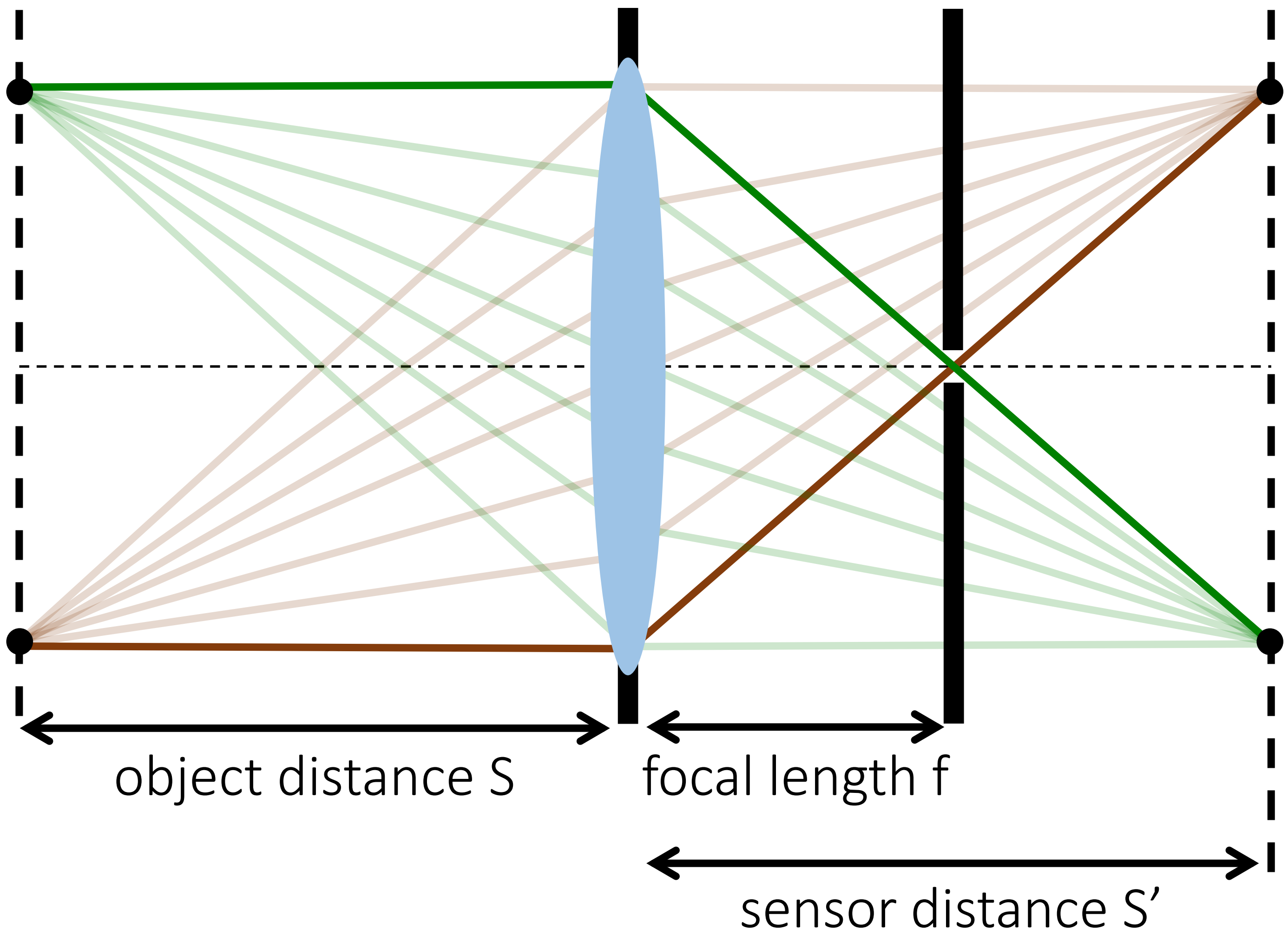


$$Z_o = \frac{f}{S' - f}$$

remember that *focal length* f refers to different things in pinhole and lens cameras

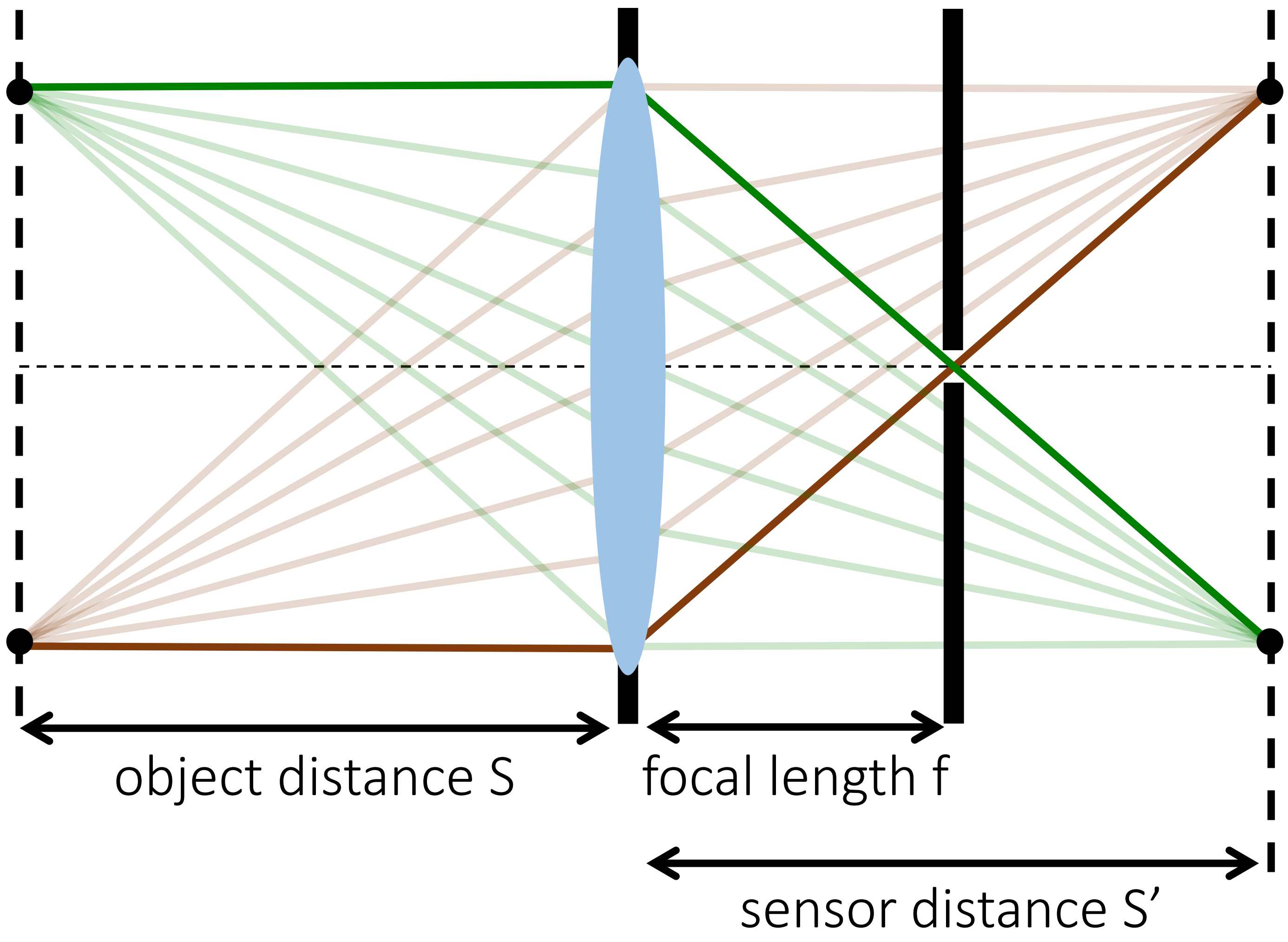
Orthographic projection using a telecentric lens

How do we make the telecentric lens act as an orthographic camera?



Orthographic projection using a telecentric lens

How do we make the telecentric lens act as an orthographic camera?

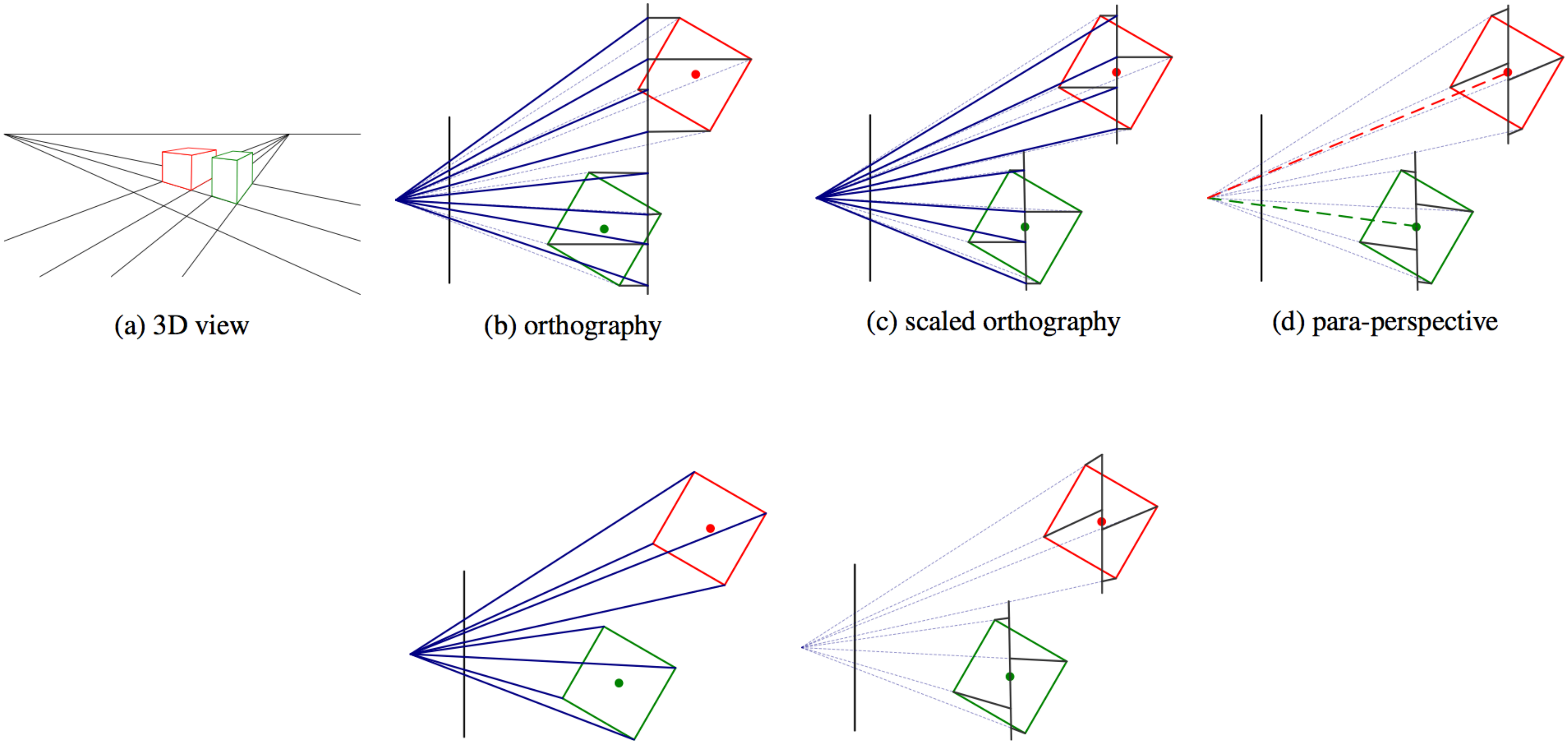


We set the sensor distance as:

$$S' = 2f$$

in order to achieve unit magnification.

Many other types of cameras



(a) 3D view (b) orthography (c) scaled orthography (d) para-perspective

(e) perspective (f) object-centered

Geometric camera calibration

Geometric camera calibration

Given a set of matched points

$$\{\mathbf{X}_i, \boldsymbol{x}_i\}$$

point in 3D space

point in the image

and camera model

$$\boldsymbol{x} = \boldsymbol{f}(\mathbf{X}; \boldsymbol{p}) = \mathbf{P}\mathbf{X}$$

projection model

parameters

Camera matrix

Find the (pose) estimate of

$$\mathbf{P}$$

We'll use a **perspective** camera model for pose estimation

Mapping between 3D point and image points

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

What are the unknowns?

Mapping between 3D point and image points

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \text{---} & \boldsymbol{p}_1^\top & \text{---} \\ \text{---} & \boldsymbol{p}_2^\top & \text{---} \\ \text{---} & \boldsymbol{p}_3^\top & \text{---} \end{bmatrix} \begin{bmatrix} | \\ \boldsymbol{X} \\ | \end{bmatrix}$$

Heterogeneous coordinates

$$x' = \frac{\boldsymbol{p}_1^\top \boldsymbol{X}}{\boldsymbol{p}_3^\top \boldsymbol{X}} \qquad y' = \frac{\boldsymbol{p}_2^\top \boldsymbol{X}}{\boldsymbol{p}_3^\top \boldsymbol{X}}$$

(non-linear relation between coordinates)

How can we make these relations linear?

How can we make these relations linear?

$$x' = \frac{\boldsymbol{p}_1^\top \boldsymbol{X}}{\boldsymbol{p}_3^\top \boldsymbol{X}} \qquad y' = \frac{\boldsymbol{p}_2^\top \boldsymbol{X}}{\boldsymbol{p}_3^\top \boldsymbol{X}}$$

Make them linear with algebraic manipulation…

$$\boldsymbol{p}_2^\top \boldsymbol{X} - \boldsymbol{p}_3^\top \boldsymbol{X} y' = 0$$

$$\boldsymbol{p}_1^\top \boldsymbol{X} - \boldsymbol{p}_3^\top \boldsymbol{X} x' = 0$$

Now we can setup a system of linear equations with multiple point correspondences

$$\boldsymbol{p}_2^\top \boldsymbol{X} - \boldsymbol{p}_3^\top \boldsymbol{X} y' = 0$$

$$\boldsymbol{p}_1^\top \boldsymbol{X} - \boldsymbol{p}_3^\top \boldsymbol{X} x' = 0$$

How do we proceed?

$$\boldsymbol{p}_2^\top \boldsymbol{X} - \boldsymbol{p}_3^\top \boldsymbol{X} y' = 0$$

$$\boldsymbol{p}_1^\top \boldsymbol{X} - \boldsymbol{p}_3^\top \boldsymbol{X} x' = 0$$

In matrix form …
$$\begin{bmatrix} \boldsymbol{X}^\top & \boldsymbol{0} & -x'\boldsymbol{X}^\top \\ \boldsymbol{0} & \boldsymbol{X}^\top & -y'\boldsymbol{X}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{bmatrix} = \boldsymbol{0}$$

How do we proceed?

$$\boldsymbol{p}_2^\top \boldsymbol{X} - \boldsymbol{p}_3^\top \boldsymbol{X} y' = 0$$

$$\boldsymbol{p}_1^\top \boldsymbol{X} - \boldsymbol{p}_3^\top \boldsymbol{X} x' = 0$$

In matrix form …

$$\begin{bmatrix} \boldsymbol{X}^\top & \mathbf{0} & -x'\boldsymbol{X}^\top \\ \mathbf{0} & \boldsymbol{X}^\top & -y'\boldsymbol{X}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{bmatrix} = \mathbf{0}$$

For N points …

$$\begin{bmatrix} \boldsymbol{X}_1^\top & \mathbf{0} & -x'\boldsymbol{X}_1^\top \\ \mathbf{0} & \boldsymbol{X}_1^\top & -y'\boldsymbol{X}_1^\top \\ \vdots & \vdots & \vdots \\ \boldsymbol{X}_N^\top & \mathbf{0} & -x'\boldsymbol{X}_N^\top \\ \mathbf{0} & \boldsymbol{X}_N^\top & -y'\boldsymbol{X}_N^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{bmatrix} = \mathbf{0}$$

How do we solve this system?

Solve for camera matrix by

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\arg\min} \|\mathbf{A}\boldsymbol{x}\|^2 \text{ subject to } \|\boldsymbol{x}\|^2 = 1$$

$$\mathbf{A} = \begin{bmatrix} \boldsymbol{X}_1^\top & \mathbf{0} & -x'\boldsymbol{X}_1^\top \\ \mathbf{0} & \boldsymbol{X}_1^\top & -y'\boldsymbol{X}_1^\top \\ \vdots & \vdots & \vdots \\ \boldsymbol{X}_N^\top & \mathbf{0} & -x'\boldsymbol{X}_N^\top \\ \mathbf{0} & \boldsymbol{X}_N^\top & -y'\boldsymbol{X}_N^\top \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{bmatrix}$$

SVD!

Solve for camera matrix by

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\mathbf{A}\boldsymbol{x}\|^2 \text{ subject to } \|\boldsymbol{x}\|^2 = 1$$

$$\mathbf{A} = \begin{bmatrix} \boldsymbol{X}_1^\top & \mathbf{0} & -x'\boldsymbol{X}_1^\top \\ \mathbf{0} & \boldsymbol{X}_1^\top & -y'\boldsymbol{X}_1^\top \\ \vdots & \vdots & \vdots \\ \boldsymbol{X}_N^\top & \mathbf{0} & -x'\boldsymbol{X}_N^\top \\ \mathbf{0} & \boldsymbol{X}_N^\top & -y'\boldsymbol{X}_N^\top \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{bmatrix}$$

Solution **x** is the column of **V** corresponding to smallest singular value of

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

Solve for camera matrix by

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\mathbf{A}\boldsymbol{x}\|^2 \text{ subject to } \|\boldsymbol{x}\|^2 = 1$$

$$\mathbf{A} = \begin{bmatrix} \boldsymbol{X}_1^\top & \mathbf{0} & -x'\boldsymbol{X}_1^\top \\ \mathbf{0} & \boldsymbol{X}_1^\top & -y'\boldsymbol{X}_1^\top \\ \vdots & \vdots & \vdots \\ \boldsymbol{X}_N^\top & \mathbf{0} & -x'\boldsymbol{X}_N^\top \\ \mathbf{0} & \boldsymbol{X}_N^\top & -y'\boldsymbol{X}_N^\top \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{bmatrix}$$

Equivalently, solution ***x*** is the Eigenvector corresponding to smallest Eigenvalue of

$$\mathbf{A}^\top\mathbf{A}$$

Now we have:

$$\mathbf{P} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix}$$

Are we done?

Almost there …

$$\mathbf{P} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix}$$

How do you get the intrinsic and extrinsic parameters from the projection matrix?

Decomposition of the Camera Matrix

$$\mathbf{P} = \left[\begin{array}{ccc|c} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{array}\right]$$

Decomposition of the Camera Matrix

$$\mathbf{P} = \left[\begin{array}{ccc|c} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{array}\right]$$

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$$

Decomposition of the Camera Matrix

$$\mathbf{P} = \left[\begin{array}{ccc|c} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{array}\right]$$

$$\begin{aligned} \mathbf{P} &= \mathbf{K}[\mathbf{R}|\mathbf{t}] \\ &= \mathbf{K}[\mathbf{R}| - \mathbf{Rc}] \\ &= [\mathbf{M}| - \mathbf{Mc}] \end{aligned}$$

Decomposition of the Camera Matrix

$-\mathbf{M}c$

$$\mathbf{P} = \left[\begin{array}{ccc|c} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{array}\right]$$

$$\begin{aligned} \mathbf{P} &= \mathbf{K}[\mathbf{R}|\mathbf{t}] \\ &= \mathbf{K}[\mathbf{R}| - \mathbf{R}\mathbf{c}] \\ &= [\mathbf{M}| - \mathbf{M}\mathbf{c}] \end{aligned}$$

$M^{-1}p$

Find the camera center **C**

What is the projection of the camera center?

Find intrinsic **K** and rotation **R**

Decomposition of the Camera Matrix

$$\mathbf{P} = \left[\begin{array}{ccc|c} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{array}\right]$$

$$\begin{aligned} \mathbf{P} &= \mathbf{K}[\mathbf{R}|\mathbf{t}] \\ &= \mathbf{K}[\mathbf{R}| - \mathbf{Rc}] \\ &= [\mathbf{M}| - \mathbf{Mc}] \end{aligned}$$

Find the camera center **C**

$$\mathbf{Pc} = \mathbf{0}$$

How do we compute the camera center from this?

Find intrinsic **K** and rotation **R**

Decomposition of the Camera Matrix

$$\mathbf{P} = \left[\begin{array}{ccc|c} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{array}\right]$$

$$\begin{aligned} \mathbf{P} &= \mathbf{K}[\mathbf{R}|\mathbf{t}] \\ &= \mathbf{K}[\mathbf{R}| - \mathbf{Rc}] \\ &= [\mathbf{M}| - \mathbf{Mc}] \end{aligned}$$

Find the camera center **C**

$$\mathbf{Pc} = \mathbf{0}$$

SVD of P!

***c** is the Eigenvector corresponding to smallest Eigenvalue*

Find intrinsic **K** and rotation **R**

Decomposition of the Camera Matrix

$$\mathbf{P} = \left[\begin{array}{ccc|c} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{array}\right]$$

$$\begin{aligned} \mathbf{P} &= \mathbf{K}[\mathbf{R}|\mathbf{t}] \\ &= \mathbf{K}[\mathbf{R}| - \mathbf{Rc}] \\ &= [\mathbf{M}| - \mathbf{Mc}] \end{aligned}$$

Find the camera center **C**

$$\mathbf{Pc} = \mathbf{0}$$

SVD of P!

***c** is the Eigenvector corresponding to smallest Eigenvalue*

Find intrinsic **K** and rotation **R**

$$\mathbf{M} = \mathbf{KR}$$

Any useful properties of K and R we can use?

Decomposition of the Camera Matrix

$$\mathbf{P} = \left[\begin{array}{ccc|c} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{array}\right]$$

$$\begin{aligned} \mathbf{P} &= \mathbf{K}[\mathbf{R}|\mathbf{t}] \\ &= \mathbf{K}[\mathbf{R}| - \mathbf{Rc}] \\ &= [\mathbf{M}| - \mathbf{Mc}] \end{aligned}$$

Find the camera center **C**

$$\mathbf{Pc} = \mathbf{0}$$

SVD of P!

***c** is the Eigenvector corresponding to smallest Eigenvalue*

Find intrinsic **K** and rotation **R**

$$\mathbf{M} = \underset{\text{right upper triangle}}{\mathbf{K}}\underset{\text{orthogonal}}{\mathbf{R}}$$

How do we find K and R?

Decomposition of the Camera Matrix

$$\mathbf{P} = \left[\begin{array}{ccc|c} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{array}\right]$$

$$\begin{aligned} \mathbf{P} &= \mathbf{K}[\mathbf{R}|\mathbf{t}] \\ &= \mathbf{K}[\mathbf{R}| - \mathbf{Rc}] \\ &= [\mathbf{M}| - \mathbf{Mc}] \end{aligned}$$

Find the camera center **C**

$$\mathbf{Pc} = \mathbf{0}$$

SVD of P!

***c** is the Eigenvector corresponding to smallest Eigenvalue*

Find intrinsic **K** and rotation **R**

$$\mathbf{M} = \mathbf{KR}$$

QR decomposition

Geometric camera calibration

Given a set of matched points

$$\{\mathbf{X}_i, \boldsymbol{x}_i\}$$

point in 3D space | point in the image

Where do we get these matched points from?

and camera model

$$\boldsymbol{x} = \boldsymbol{f}(\mathbf{X}; \boldsymbol{p}) = \mathbf{P}\mathbf{X}$$

projection model | parameters | Camera matrix

Find the (pose) estimate of

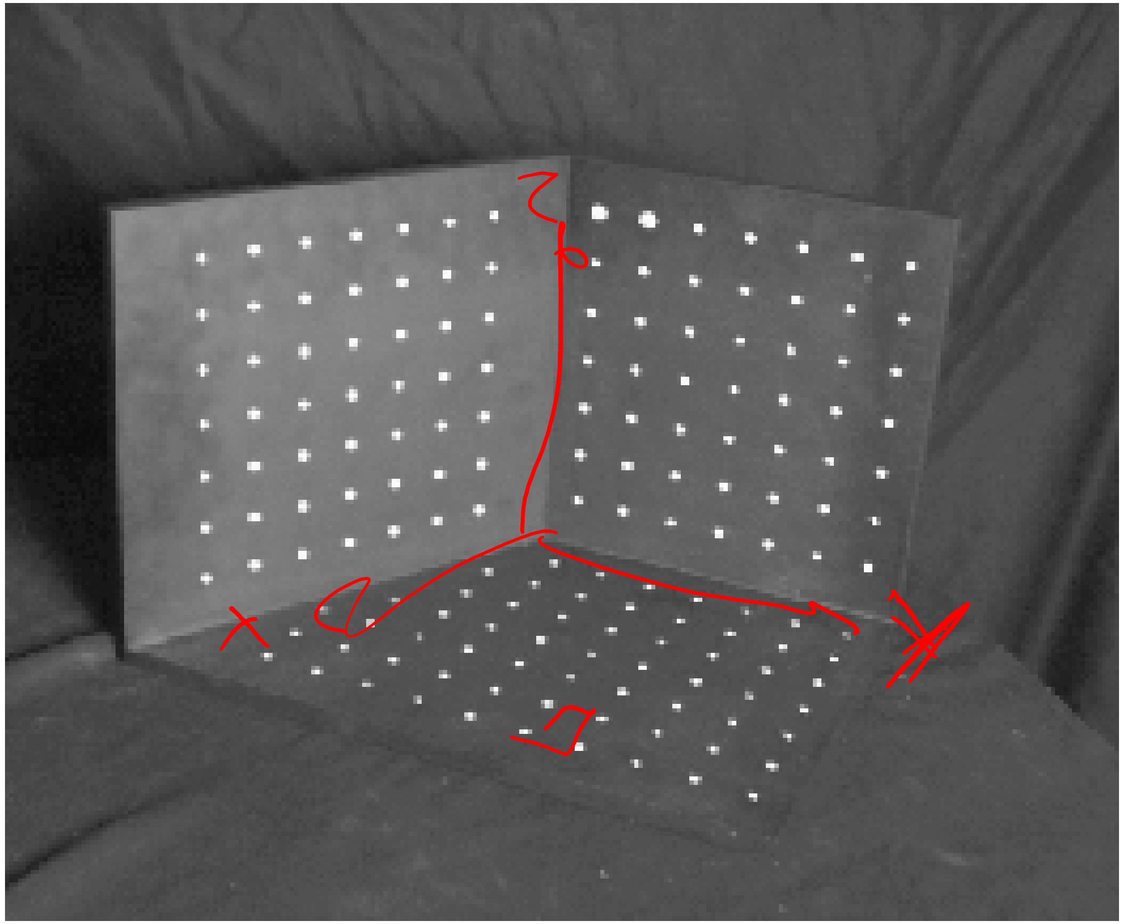
$$\mathbf{P}$$

We'll use a **perspective** camera model for pose estimation

Calibration using a reference object

Place a known object in the scene:

- identify correspondences between image and scene
- compute mapping from scene to image



Issues:

- must know geometry very accurately
- must know 3D->2D correspondence

Geometric camera calibration

Advantages:

- Very simple to formulate.
- Analytical solution.

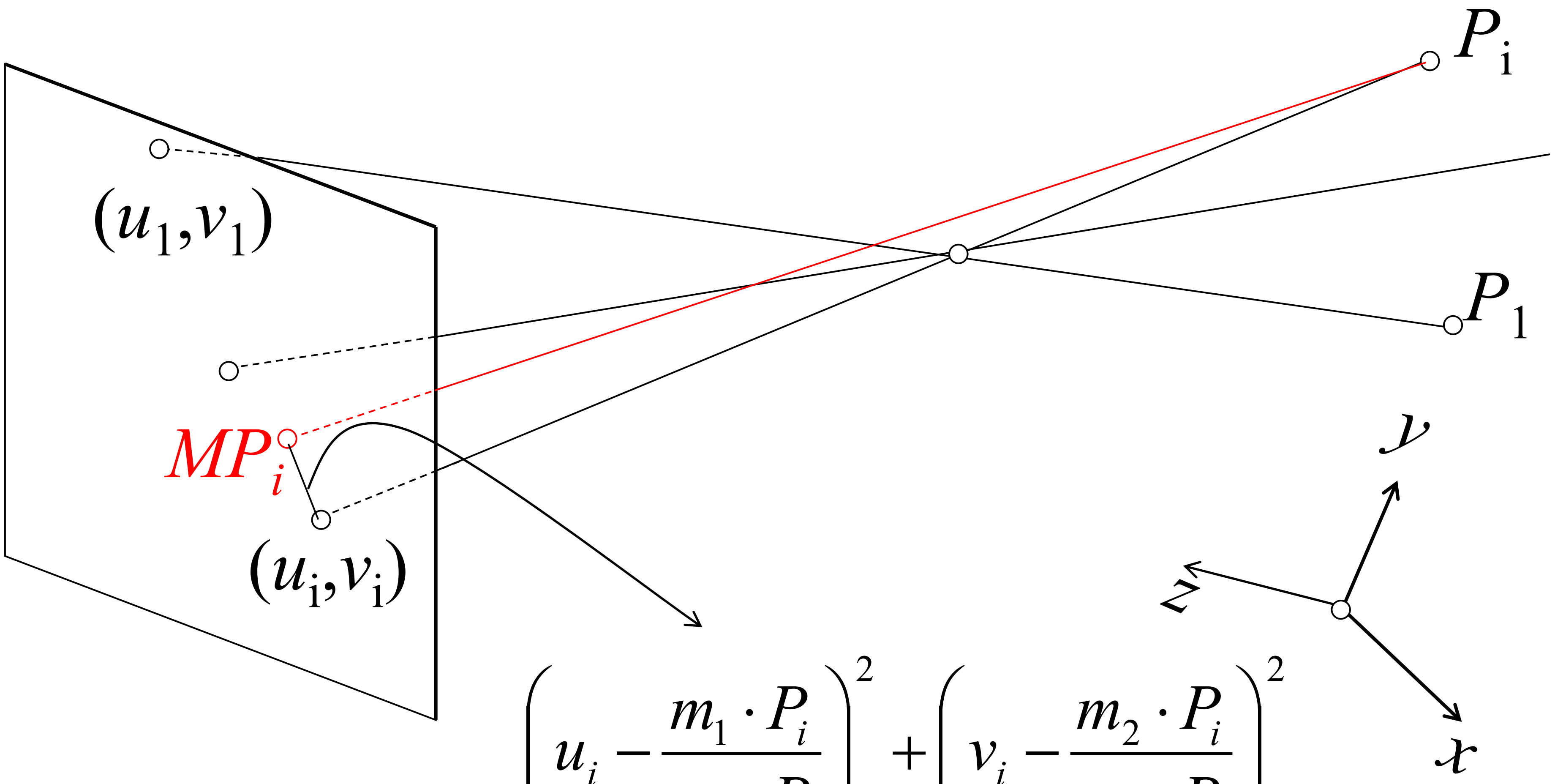
Disadvantages:

- Doesn't model radial distortion.
- Hard to impose constraints (e.g., known *f*).
- Doesn't minimize the correct error function.

For these reasons, *nonlinear methods* are preferred

- Define error function E between projected 3D points and image positions
  - E is nonlinear function of intrinsics, extrinsics, radial distortion
- Minimize E using nonlinear optimization techniques

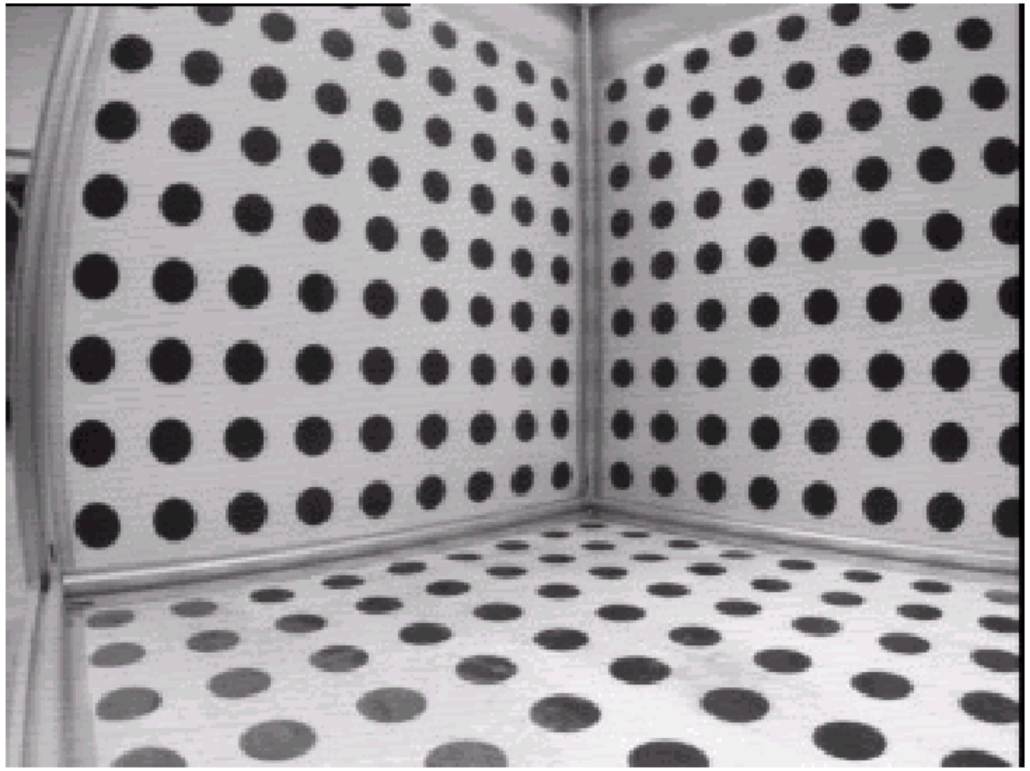
Minimizing reprojection error



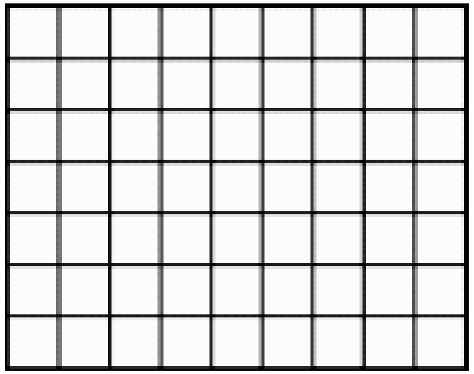
$$\left(u_i - \frac{m_1 \cdot P_i}{m_3 \cdot P_i}\right)^2 + \left(v_i - \frac{m_2 \cdot P_i}{m_3 \cdot P_i}\right)^2$$

Is this equivalent to what we were doing previously?

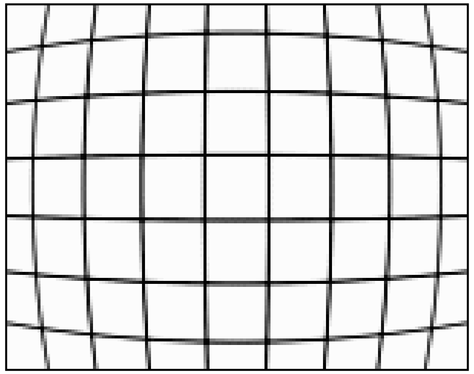
Radial distortion



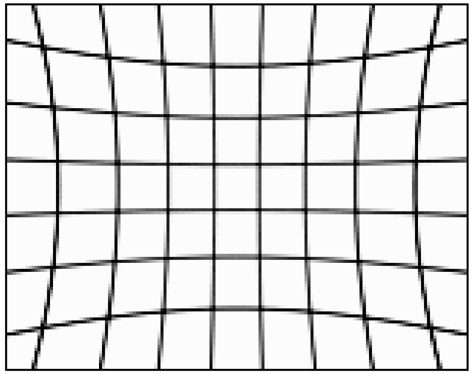
What causes this distortion?



no distortion

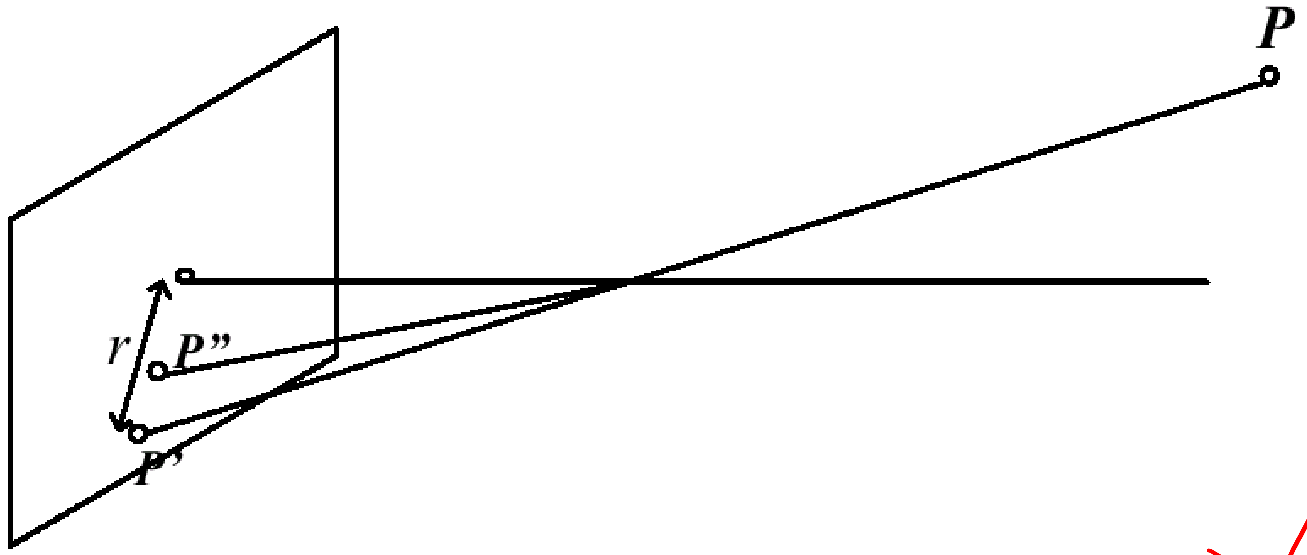


barrel distortion

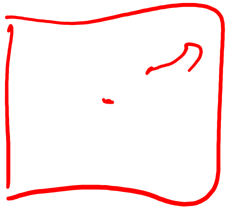


pincushion distortion

Radial distortion model



$\sqrt{x^2+y^2}$



Ideal:

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

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

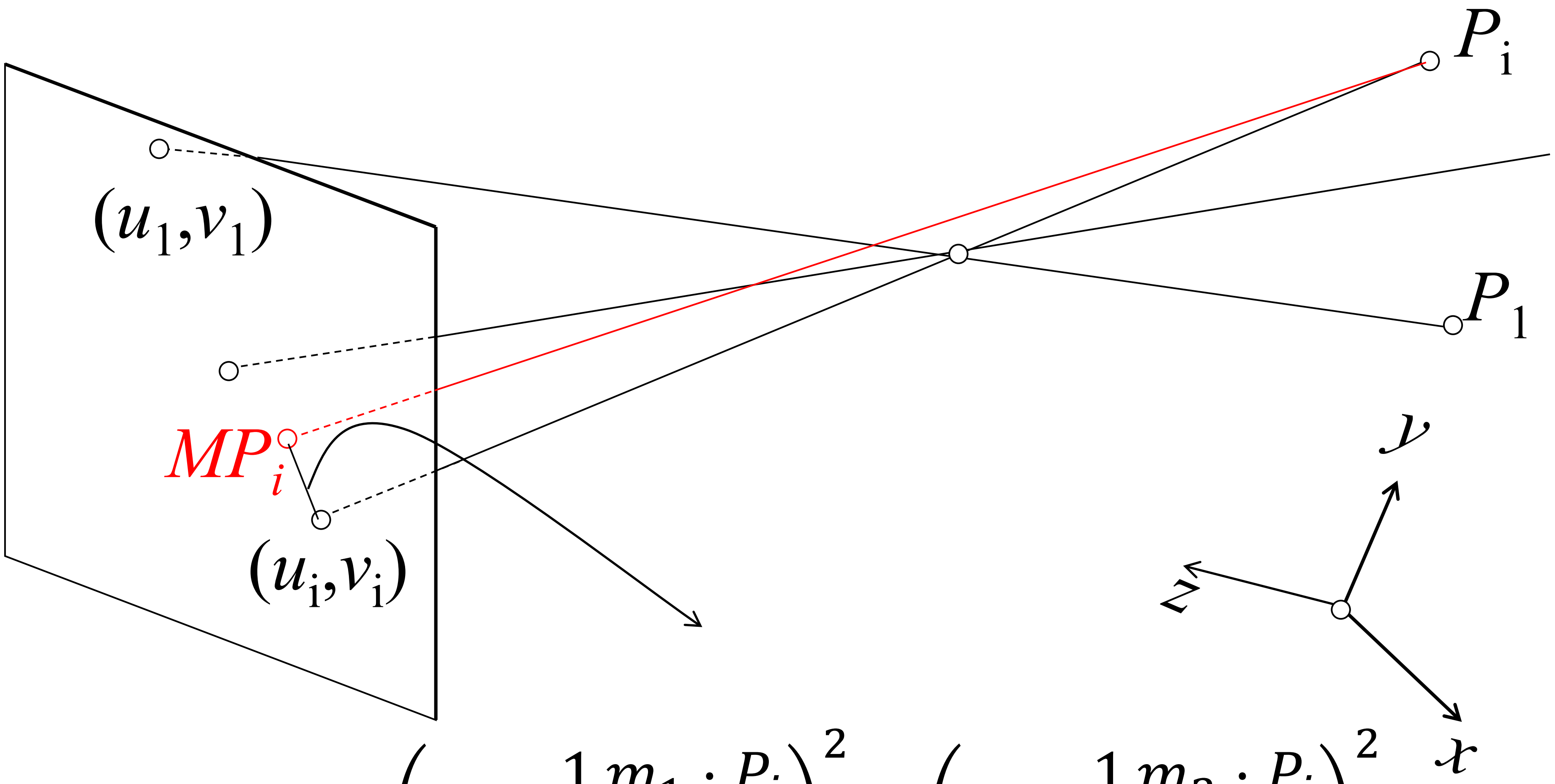
Distorted:

$$x'' = \frac{1}{\lambda}x'$$

$$y'' = \frac{1}{\lambda}y'$$

$$\lambda = 1 + k_1 r^2 + k_2 r^4 + \cdots$$

Minimizing reprojection error with radial distortion



Add distortions to reprojection error:

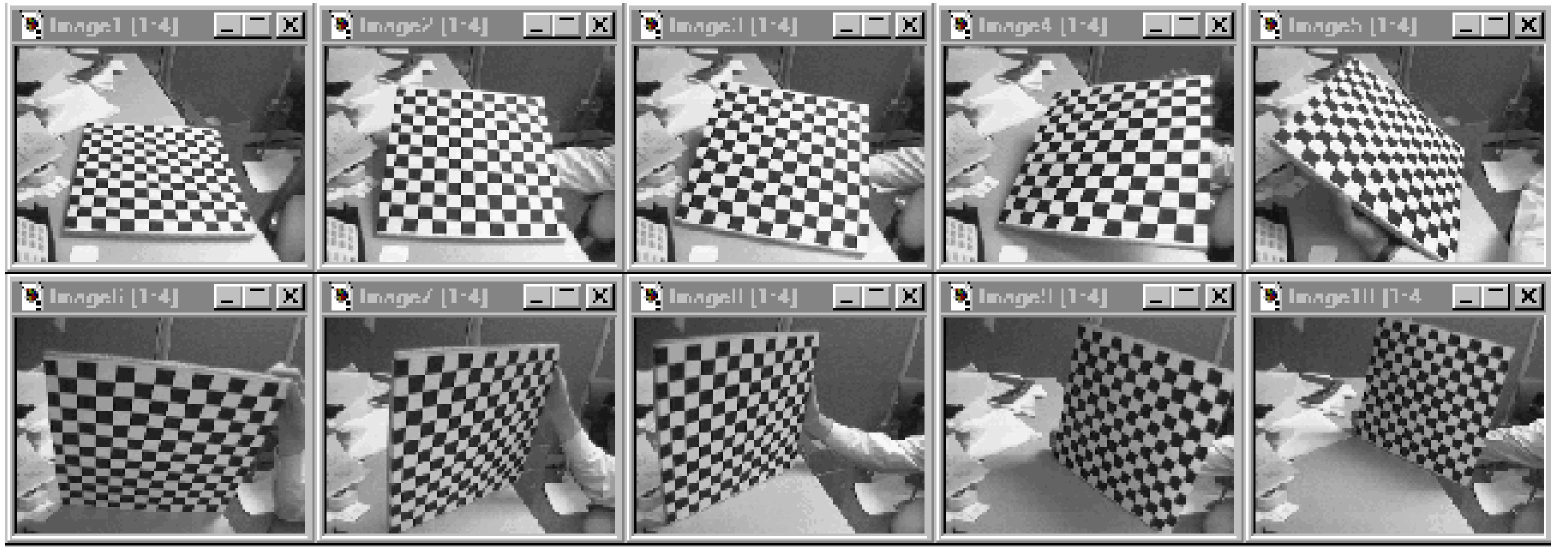
$$\left(u_i - \frac{1}{\lambda}\frac{m_1 \cdot P_i}{m_3 \cdot P_i}\right)^2 + \left(v_i - \frac{1}{\lambda}\frac{m_2 \cdot P_i}{m_3 \cdot P_i}\right)^2$$

Correcting radial distortion

before

after

Alternative: Multi-plane calibration

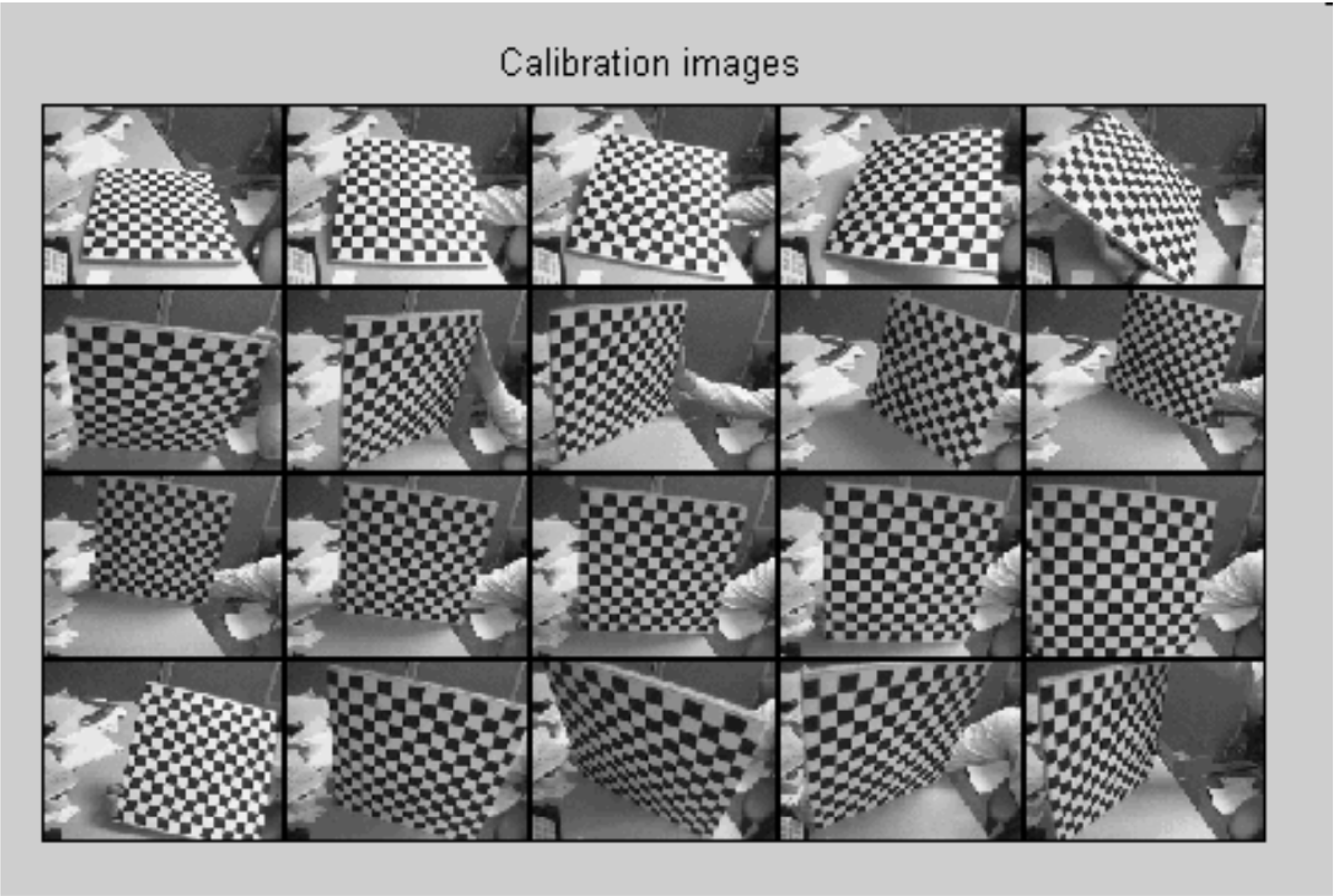


Advantages:

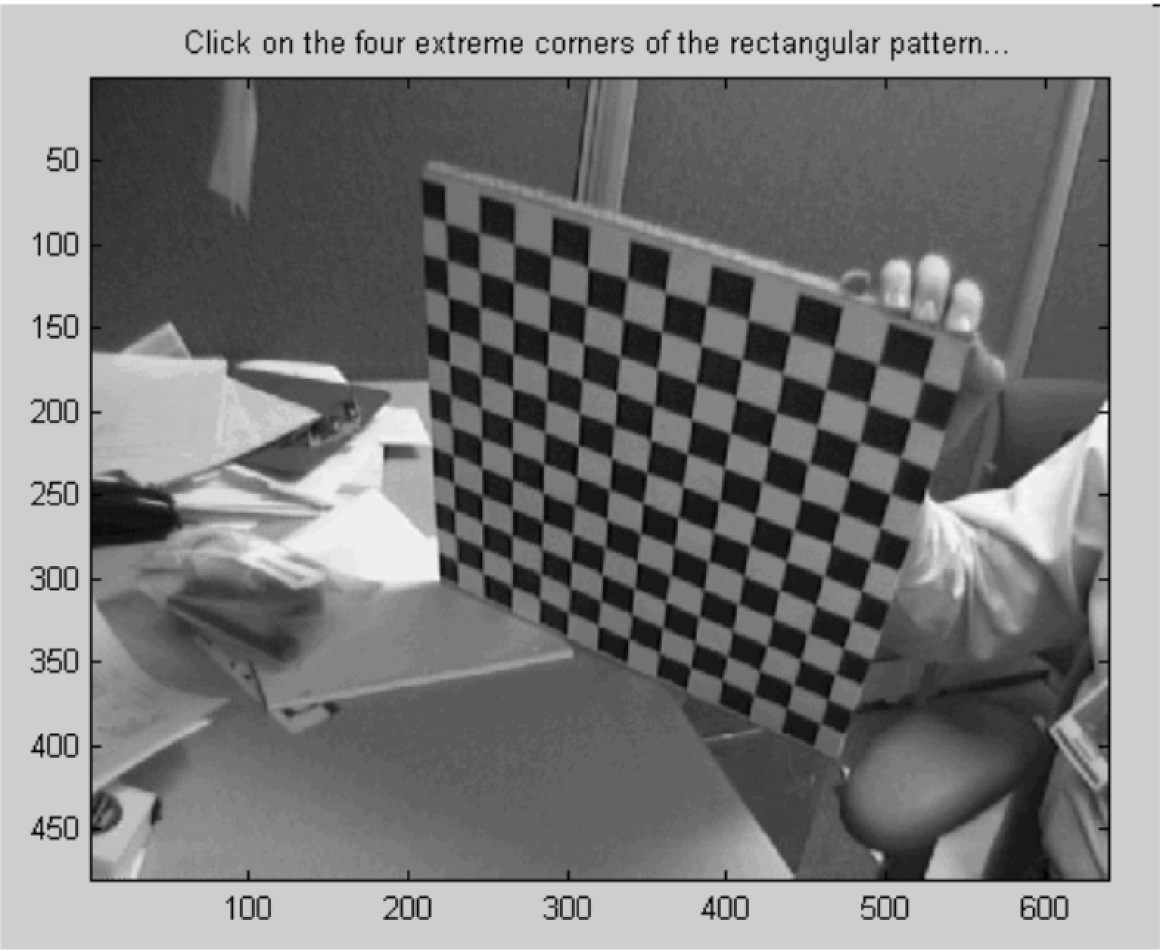
- Only requires a plane
- Don’t have to know positions/orientations
- Great code available online!
  - Matlab version: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html
  - Also available on OpenCV.

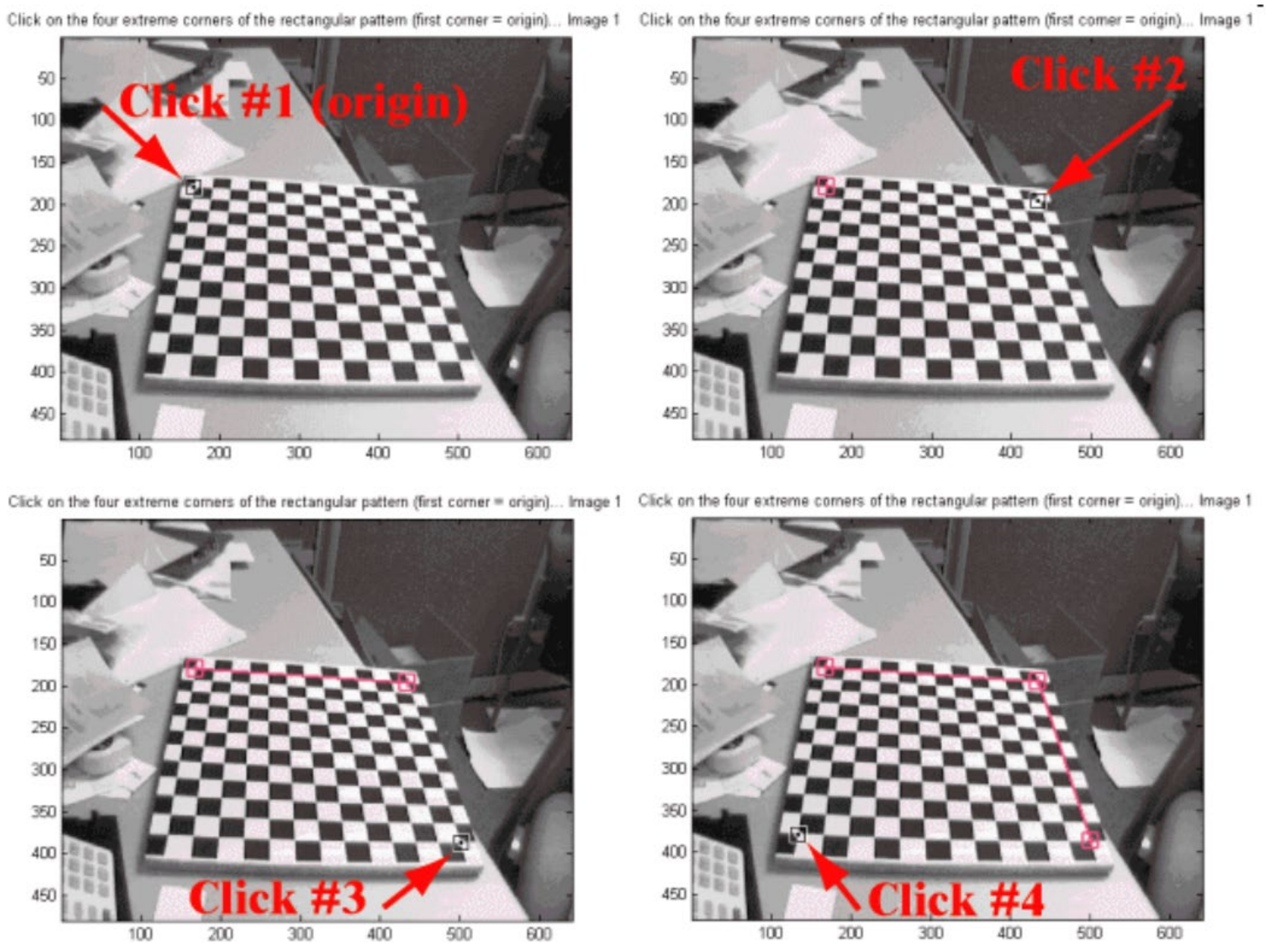
Disadvantage: Need to solve non-linear optimization problem.

Step-by-step demonstration

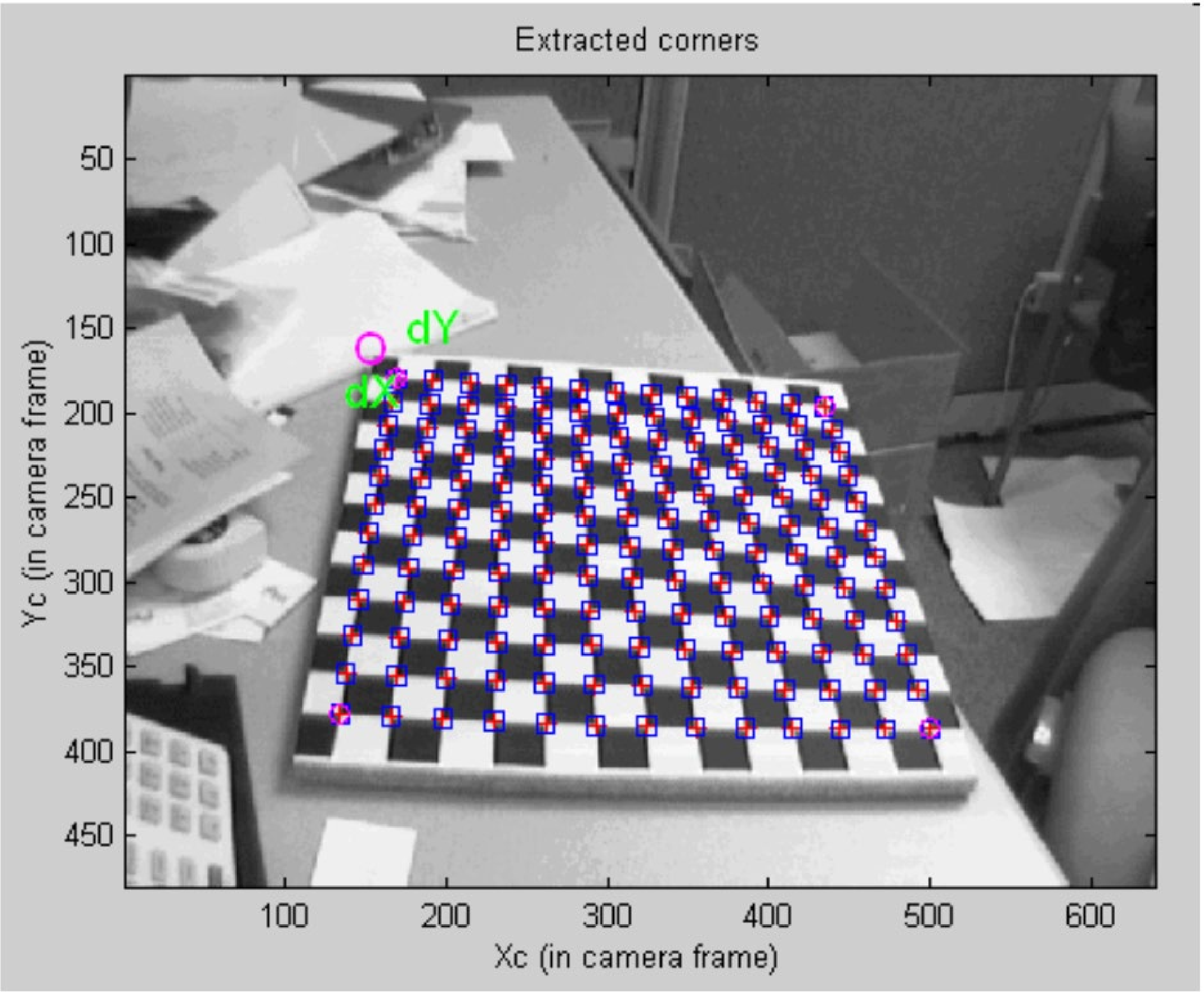


Step-by-step demonstration

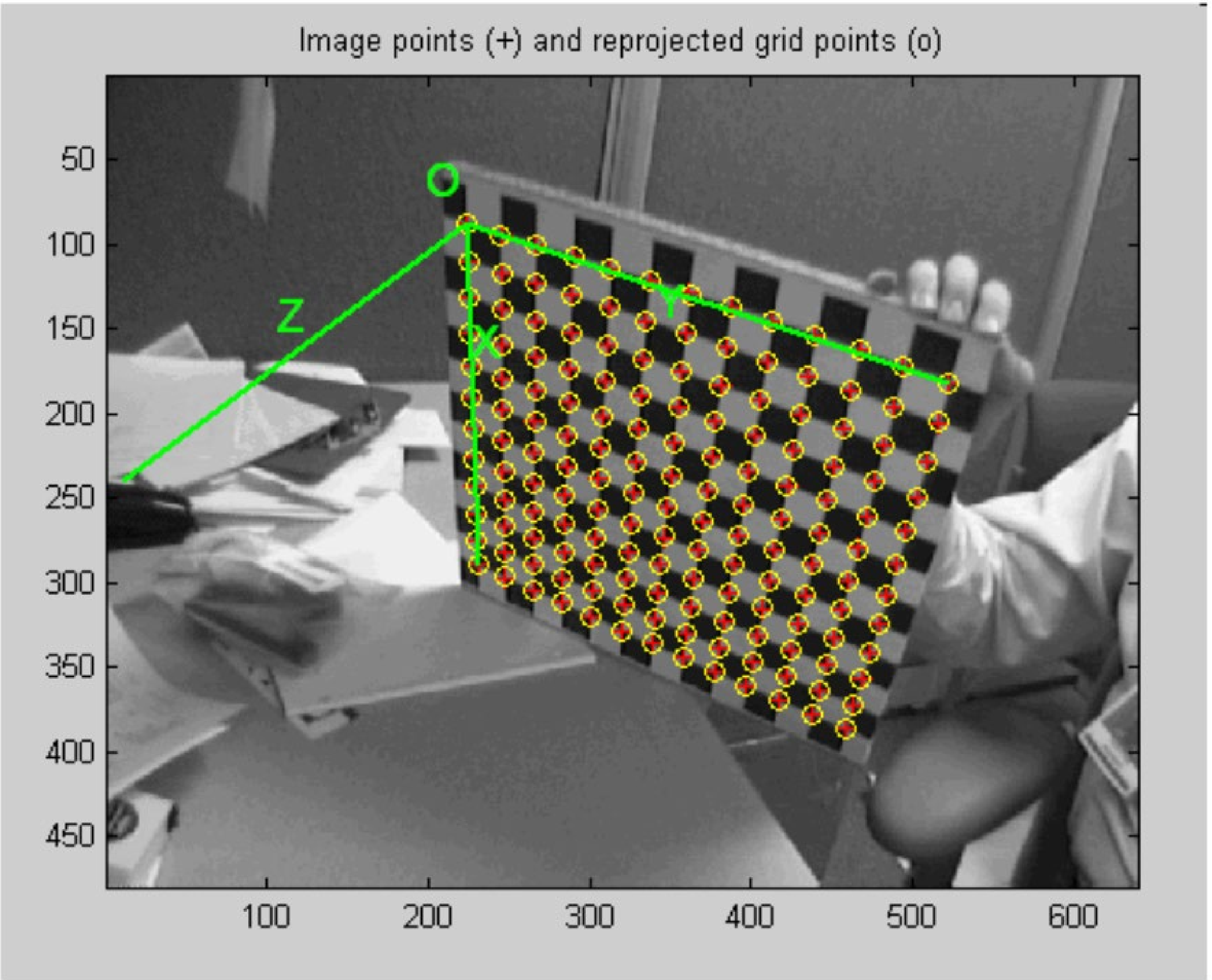




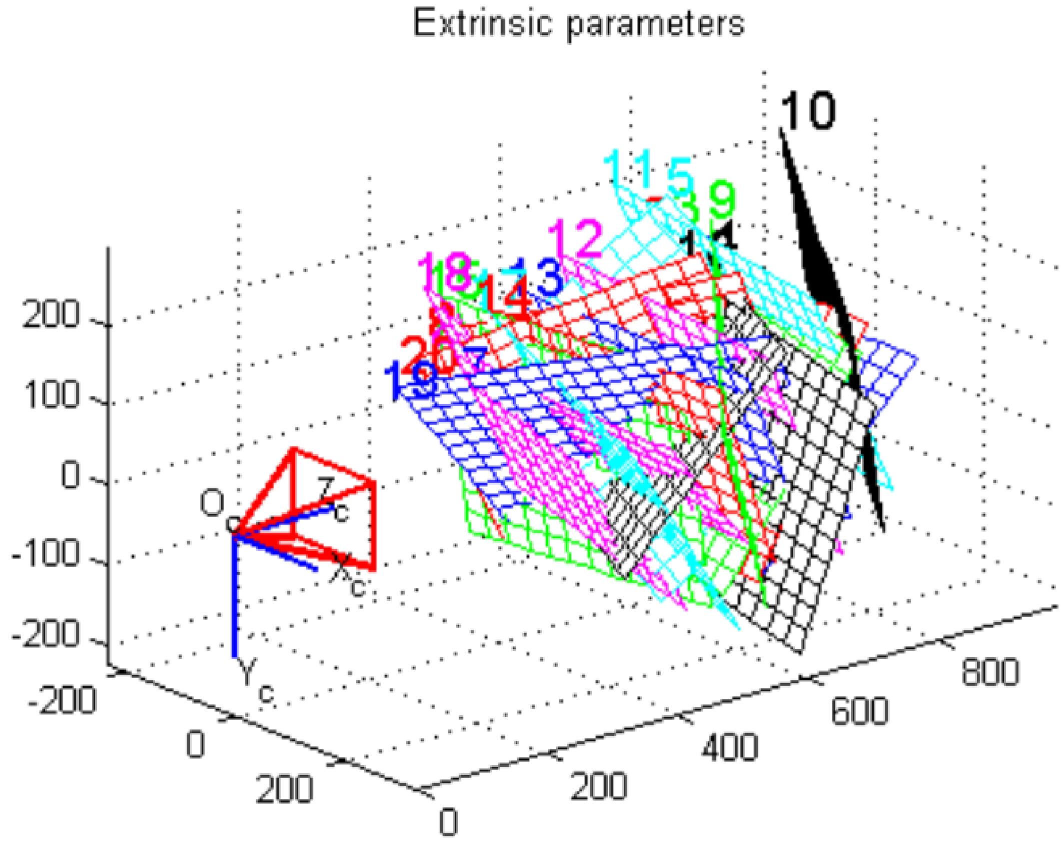
Step-by-step demonstration

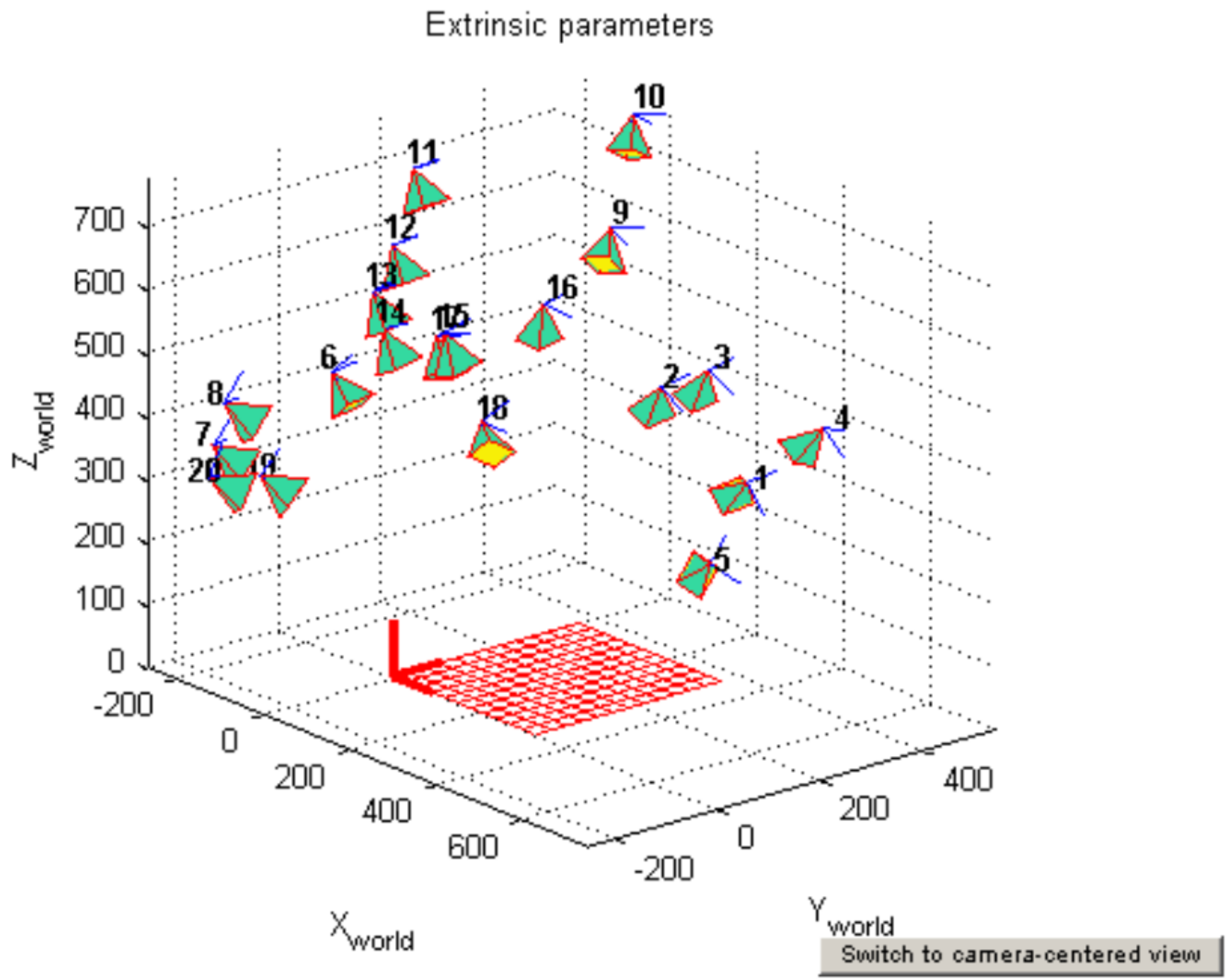


Step-by-step demonstration



Step-by-step demonstration





What does it mean to “calibrate a camera”?

What does it mean to “calibrate a camera”?

Many different ways to calibrate a camera:

- Radiometric calibration. ← lecture 5-ish
- Color calibration. ← lecture 7-ish
- Geometric calibration. ← lecture 19 (this lecture)
- Noise calibration. ← lecture 6-ish
- Lens (or aberration) calibration. ← lecture 12-ish, (maybe) later lecture

References

Basic reading:

- Szeliski textbook, Section 2.1.5, 6.2.
- Bouguet, "Camera calibration toolbox for Matlab," available at http://www.vision.caltech.edu/bouguetj/calib_doc/

  The main resource for camera calibration in Matlab, where the screenshots in this lecture were taken from. It also has a detailed of the camera calibration algorithm and an extensive reference section.

Additional reading:

- Hartley and Zisserman, "Multiple View Geometry in Computer Vision," Cambridge University Press 2004.

  Chapter 6 of this book has a very thorough treatment of camera models.
- Gortler, "Foundations of 3D Computer Graphics," MIT Press 2012.

  Chapter 10 of this book has a nice discussion of pinhole cameras from a graphics point of view.
- Zhang, "A flexible new technique for camera calibration," PAMI 2000.

  The paper that introduced camera calibration from multiple views of a planar target.