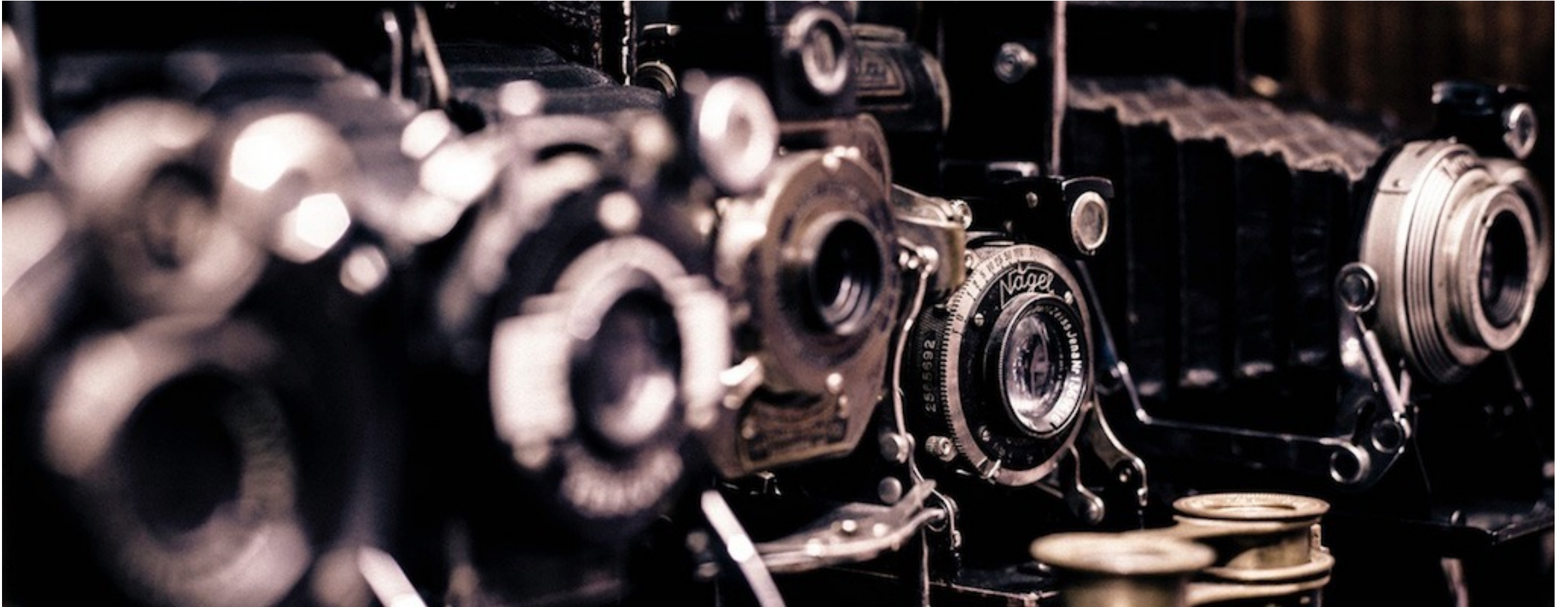


# Geometric camera models (cont.)



# Overview of today's lecture

- Review of 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).

# Recap

What is the size and meaning of each term in the camera matrix?

$$\mathbf{P} = \mathbf{KR}[\mathbf{I} \mid -\mathbf{C}]$$

The diagram shows the equation  $\mathbf{P} = \mathbf{KR}[\mathbf{I} \mid -\mathbf{C}]$  with four arrows pointing upwards from question marks to the terms  $\mathbf{P}$ ,  $\mathbf{KR}$ ,  $[\mathbf{I} \mid -\mathbf{C}]$ , and  $\mathbf{C}$ . The arrows originate from question marks located below the terms: one under  $\mathbf{P}$ , one under  $\mathbf{KR}$ , one under the vertical bar in  $[\mathbf{I} \mid -\mathbf{C}]$ , and one under  $\mathbf{C}$ .

# Recap

What is the size and meaning of each term in the camera matrix?

$$\mathbf{P} = \mathbf{KR}[\mathbf{I} \mid \mathbf{C}]$$

The diagram shows the equation  $\mathbf{P} = \mathbf{KR}[\mathbf{I} \mid \mathbf{C}]$  with four arrows pointing from labels below to the terms  $\mathbf{P}$ ,  $\mathbf{K}$ ,  $\mathbf{I}$ , and  $\mathbf{C}$ . The label for  $\mathbf{P}$  is "3x3" and "intrinsic". The labels for  $\mathbf{K}$ ,  $\mathbf{I}$ , and  $\mathbf{C}$  are each a question mark.

3x3  
intrinsic

?

?

?

# Recap

What is the size and meaning of each term in the camera matrix?

$$\mathbf{P} = \mathbf{KR}[\mathbf{I} | \mathbf{C}]$$

3x3  
intrinsics

3x3  
3D rotation

?

?

# Recap

What is the size and meaning of each term in the camera matrix?


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

The diagram shows the equation  $\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I} | \mathbf{C}]$  with four arrows pointing from labels below to the terms  $\mathbf{K}$ ,  $\mathbf{R}$ ,  $[\mathbf{I} |$ , and  $\mathbf{C}]$ . The labels are:  $3 \times 3$  intrinsics,  $3 \times 3$  3D rotation,  $3 \times 3$  identity, and a question mark.

$3 \times 3$  intrinsics     $3 \times 3$  3D rotation     $3 \times 3$  identity    ?

# Recap

What is the size and meaning of each term in the camera matrix?

$$\mathbf{P} = \mathbf{KR}[\mathbf{I} | -\mathbf{C}]$$


3x3 intrinsics    3x3 3D rotation    3x3 identity    3x1 3D translation



Perspective distortion

# Finite projective camera

$$\mathbf{P} = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[ \mathbf{R} \mid -\mathbf{RC} \right]$$



What does this matrix look like if the camera and world have the same coordinate system?

# Finite projective camera

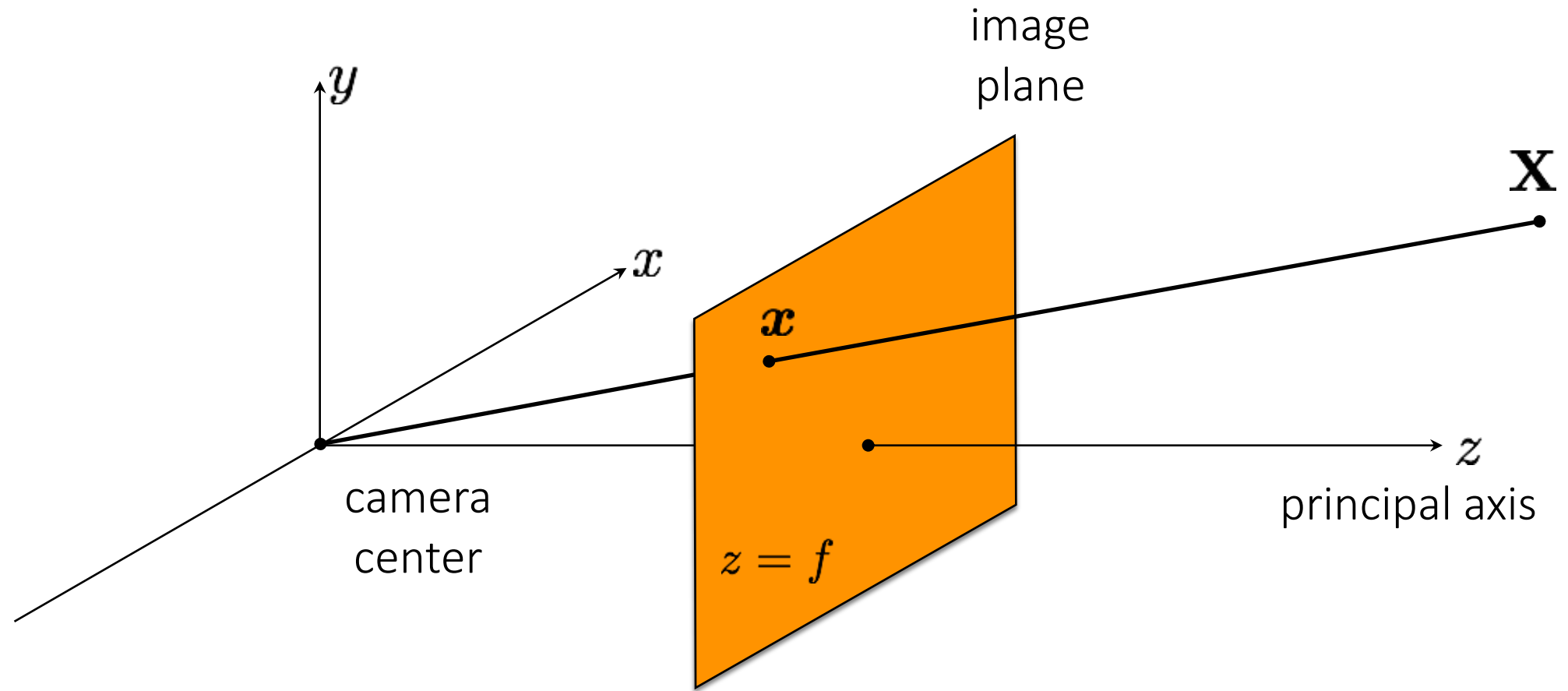
The pinhole camera and all of the more general cameras we have seen so far have “*perspective distortion*”.

$$\mathbf{P} = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



*Perspective* projection from  
(homogeneous) 3D to 2D coordinates

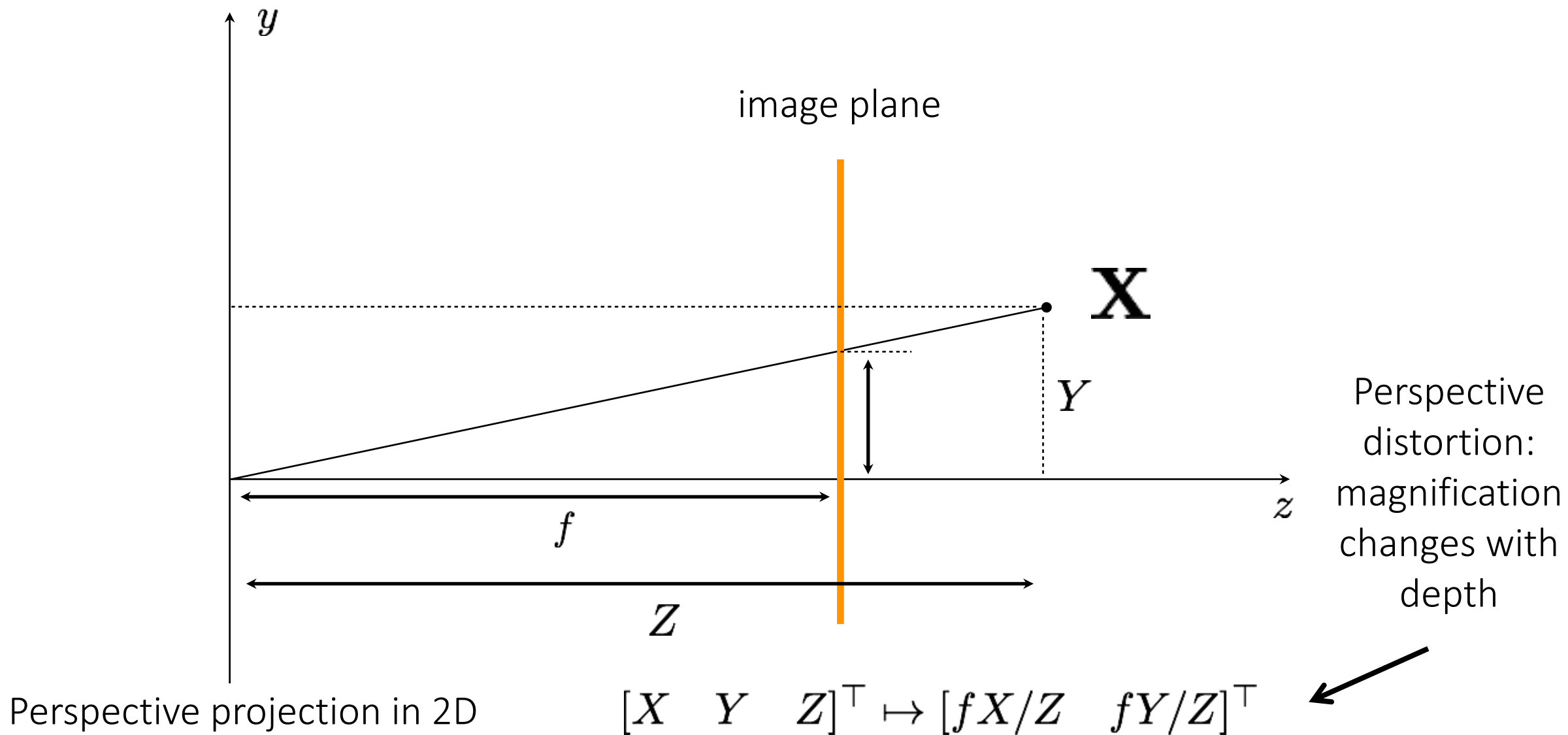
# The (rearranged) pinhole camera



Perspective projection in 3D

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

# The 2D view of the (rearranged) pinhole camera



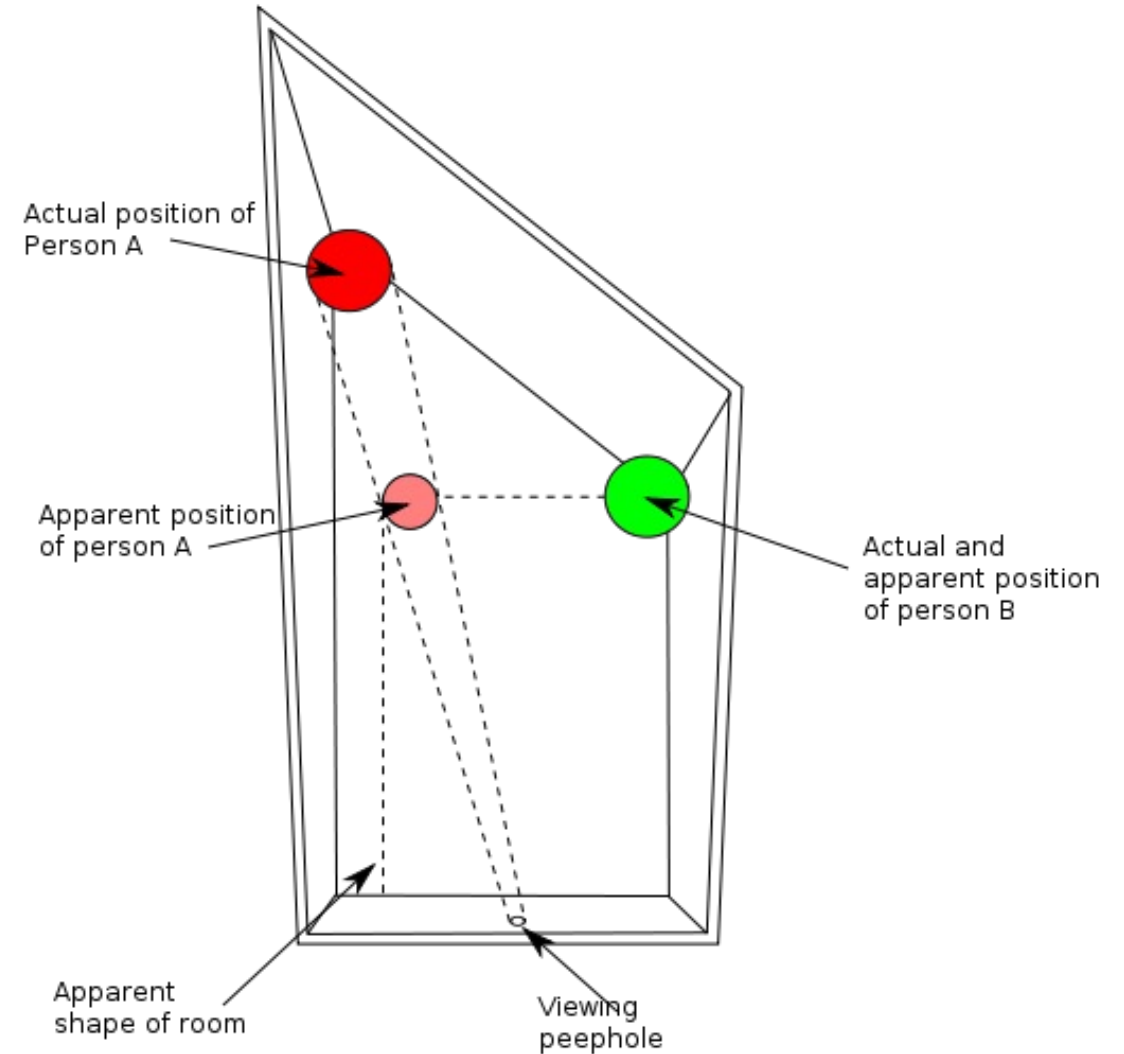
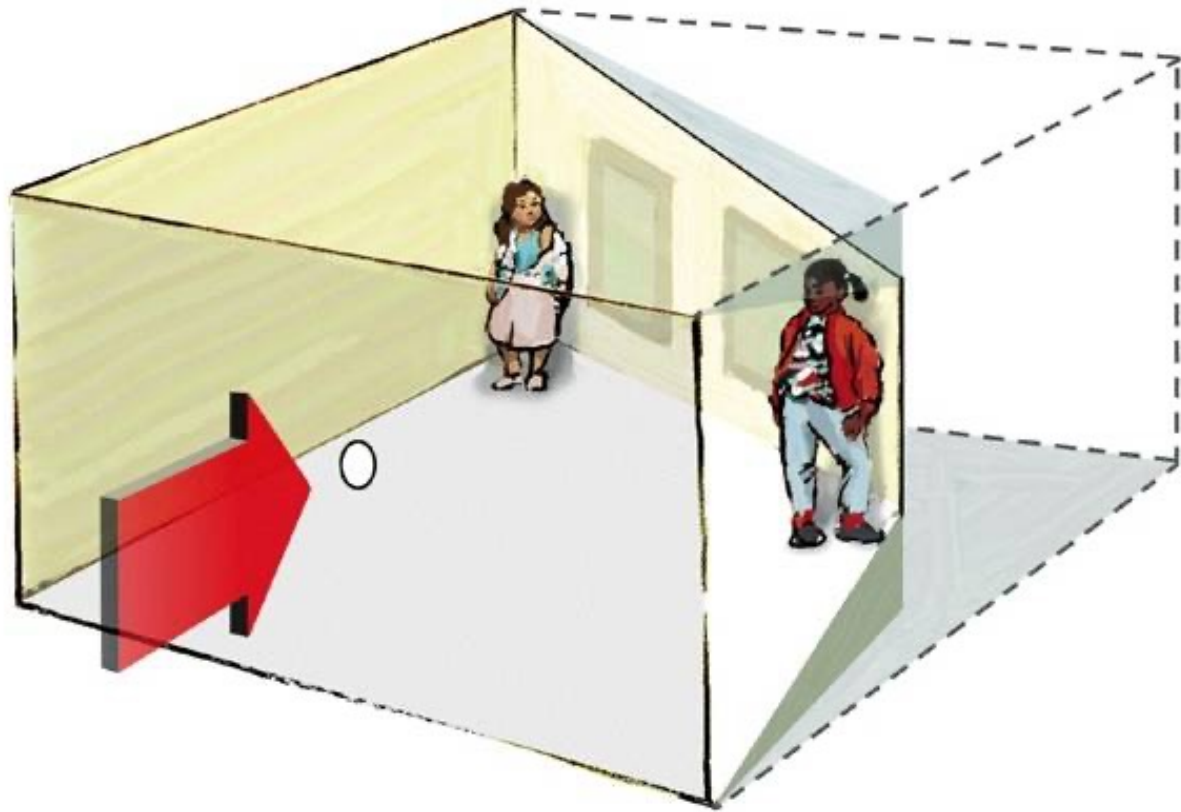
# Forced perspective



# The Ames room illusion



# The Ames room illusion





# The arrow illusion



# Forced Perspective Display



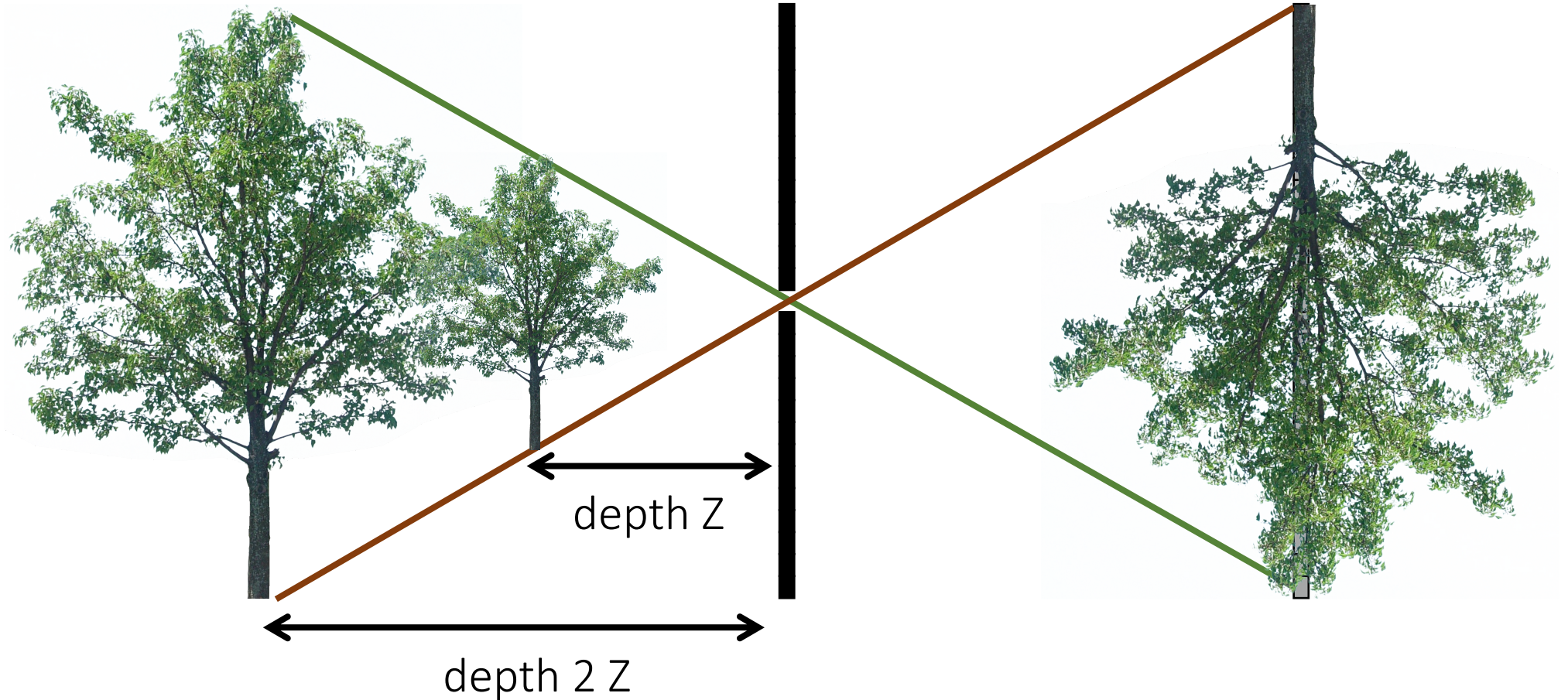
# Forced Perspective Display



# Magnification depends on depth

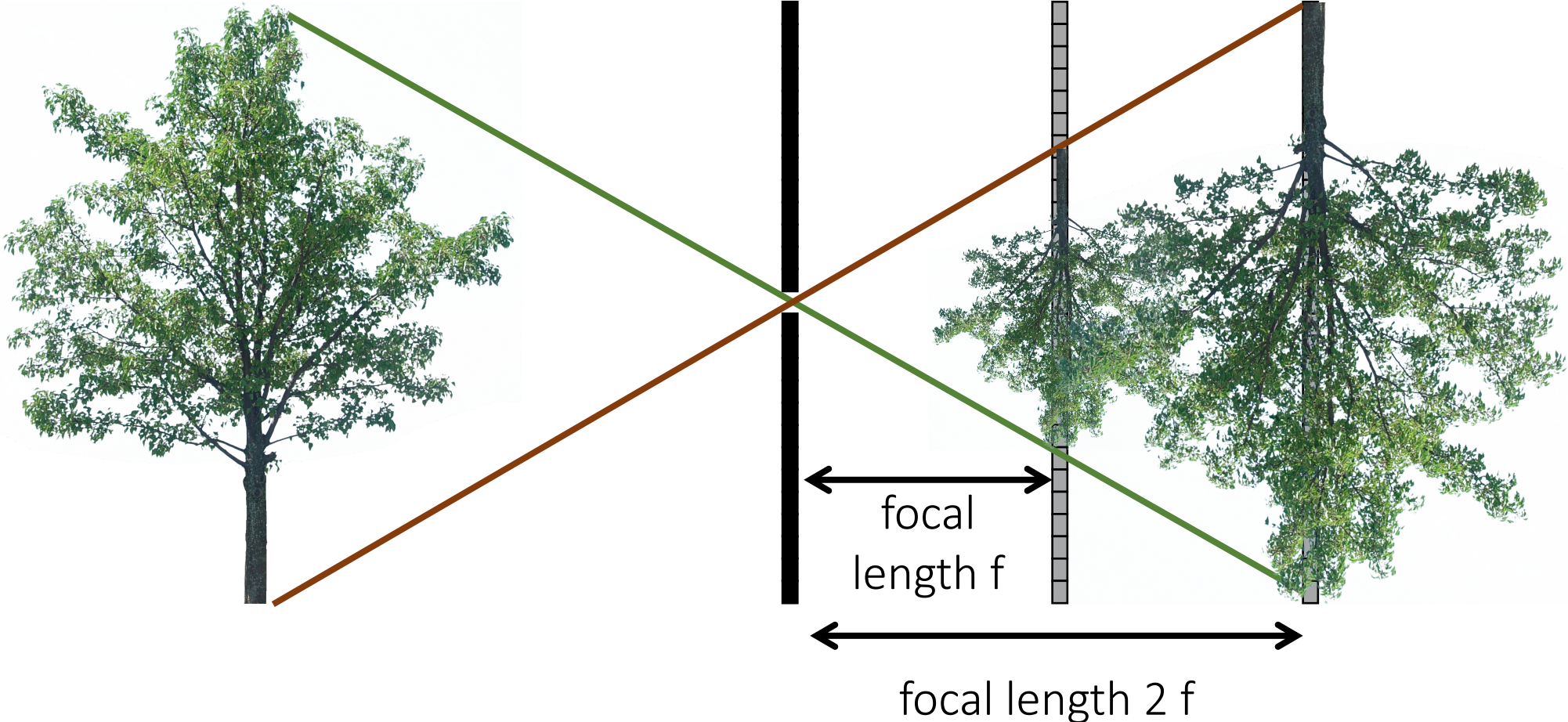
What happens as we change the focal length?

real-world  
object



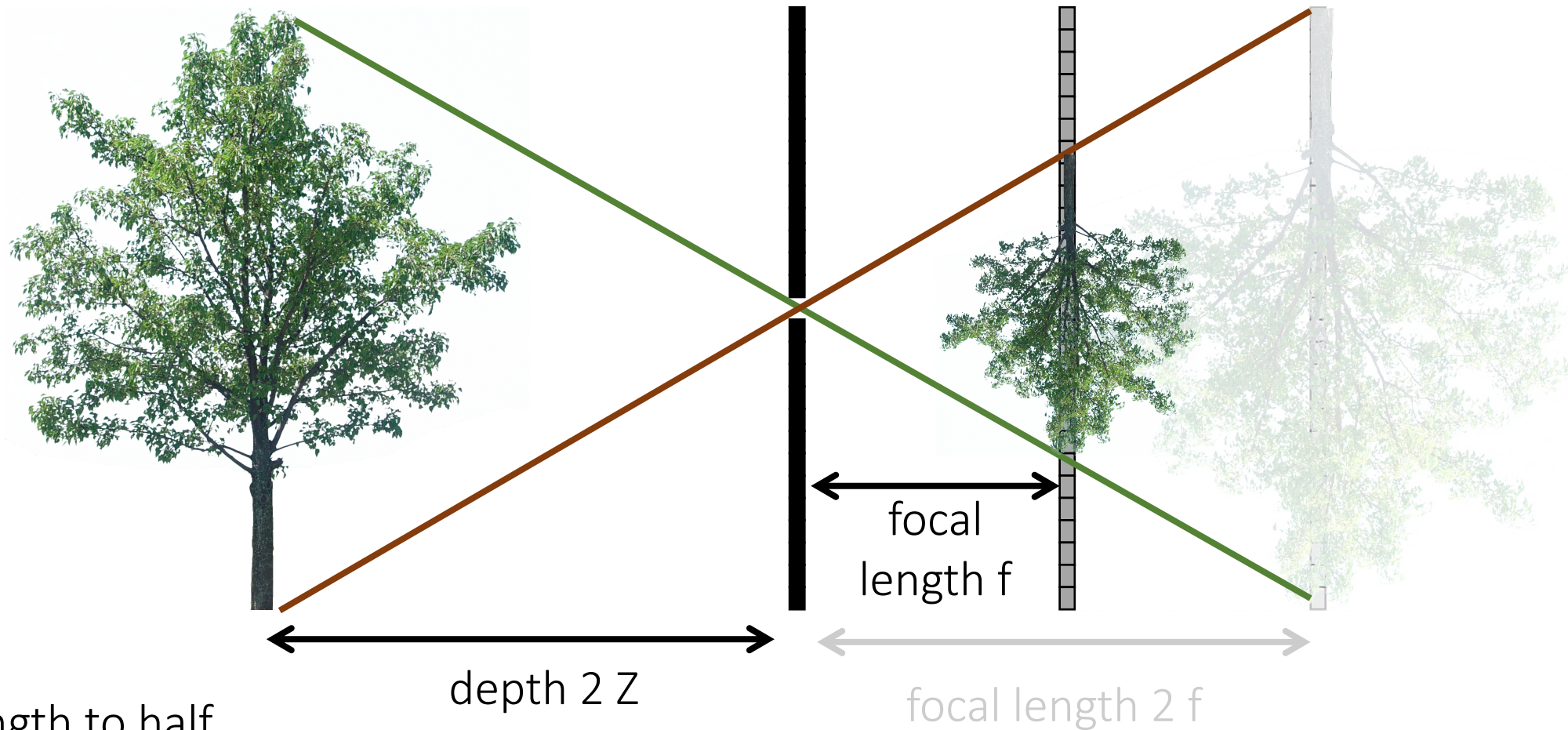
# Magnification depends on focal length

real-world  
object



# What if...

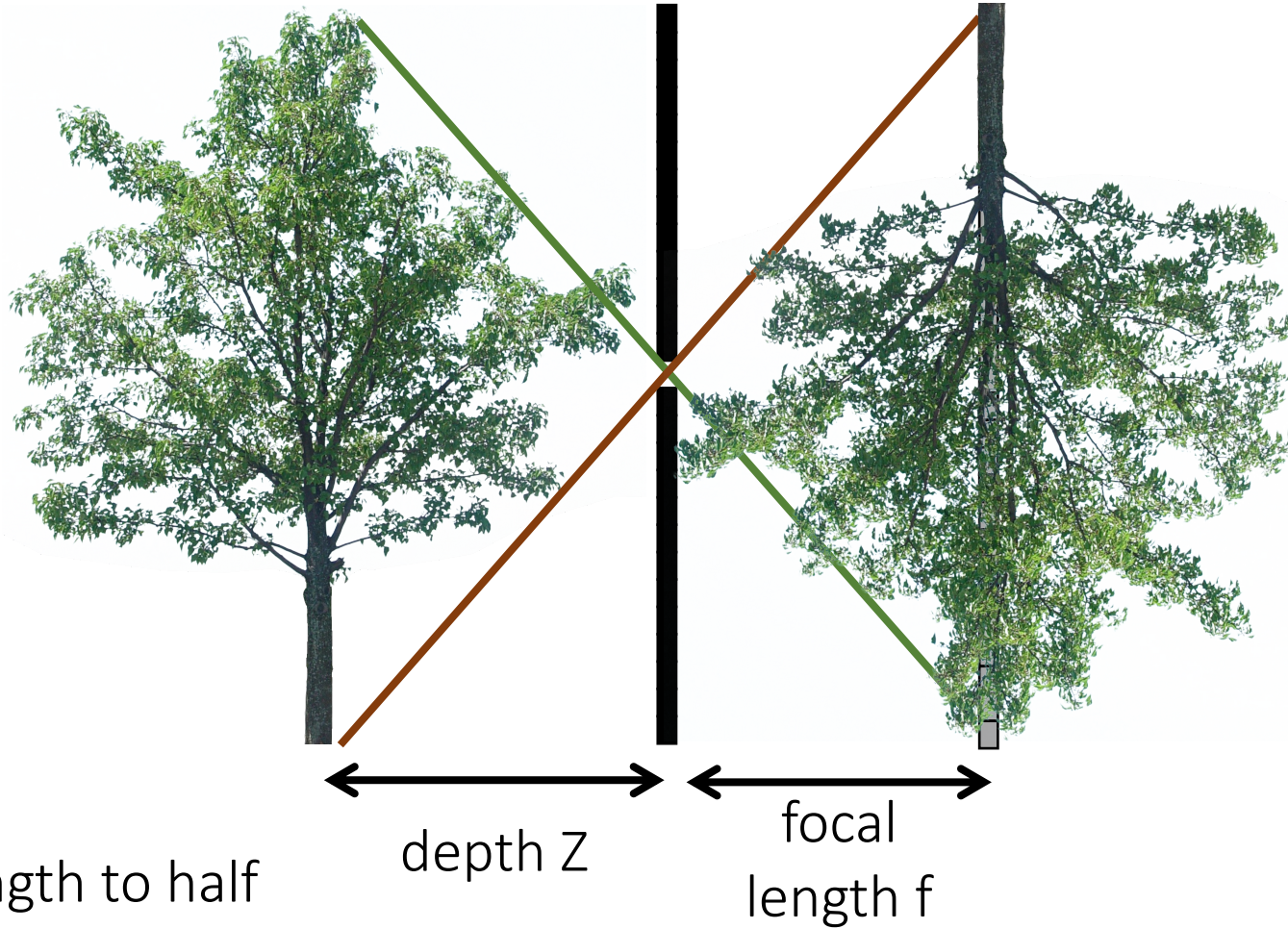
real-world  
object



1. Set focal length to half

# What if...

real-world  
object



Is this the same image as  
the one I had at focal  
length  $2f$  and distance  $2Z$ ?

1. Set focal length to half
2. Set depth to half

# Perspective distortion



long focal length



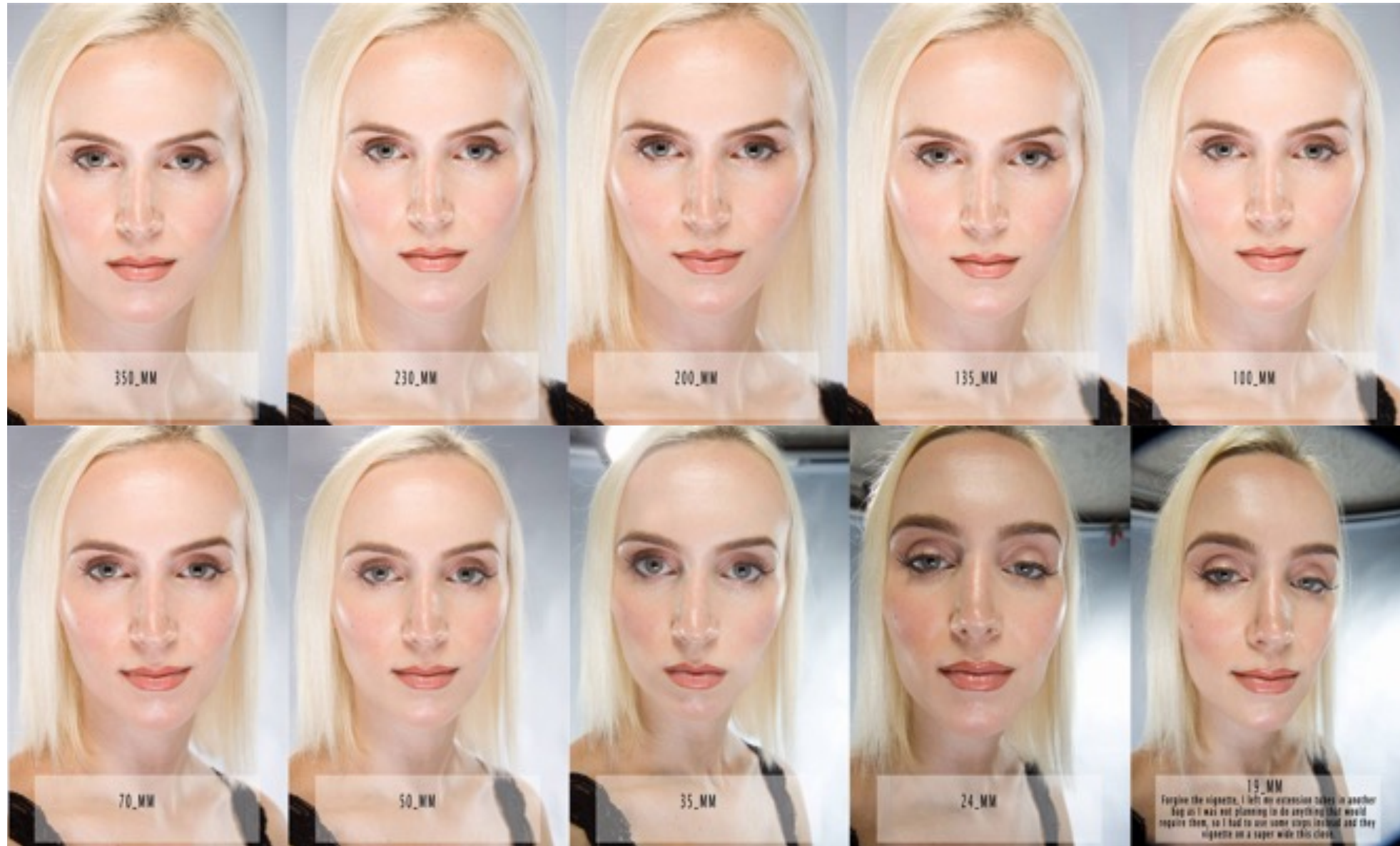
mid focal length



short focal length



# Perspective distortion



# Vertigo effect

Named after Alfred Hitchcock's movie

- also known as “dolly zoom”



# Vertigo effect



How would you  
create this effect?

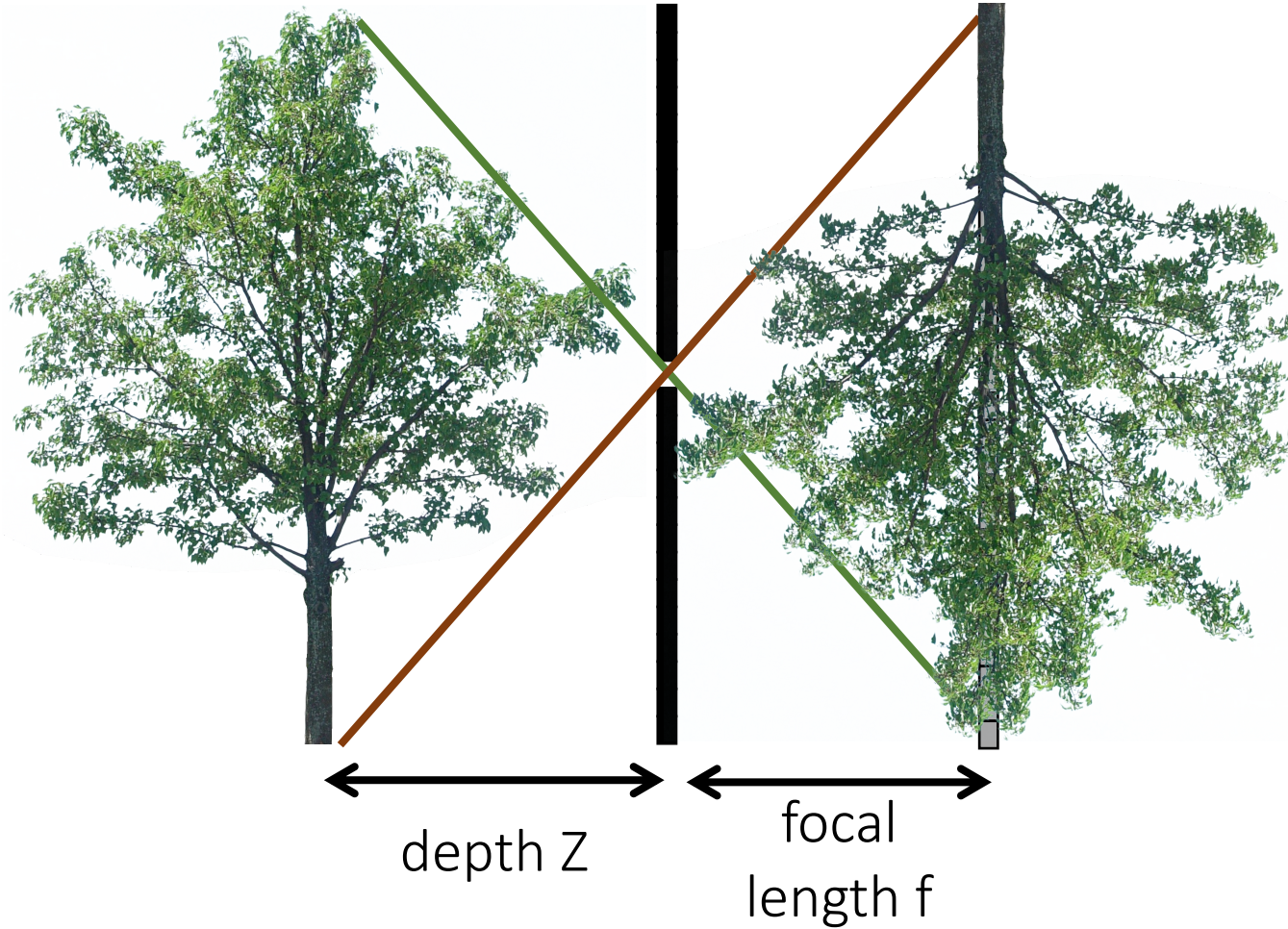
# Vertigo effect



Other camera models

# What if...

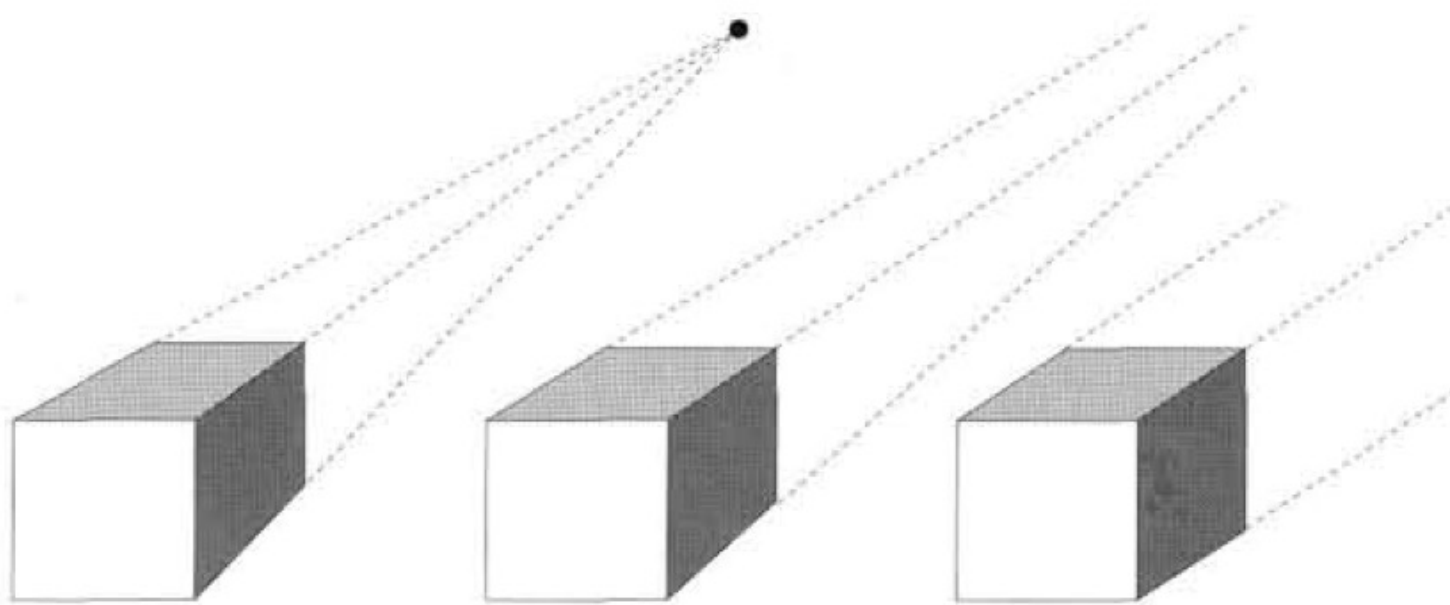
real-world  
object



... we continue increasing  $Z$   
and  $f$  while maintaining  
same magnification?

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

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



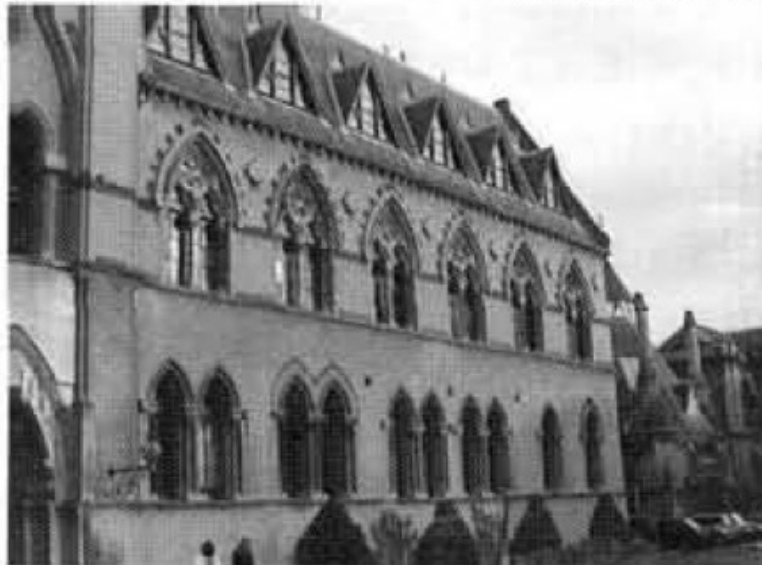
**perspective**

**weak perspective**

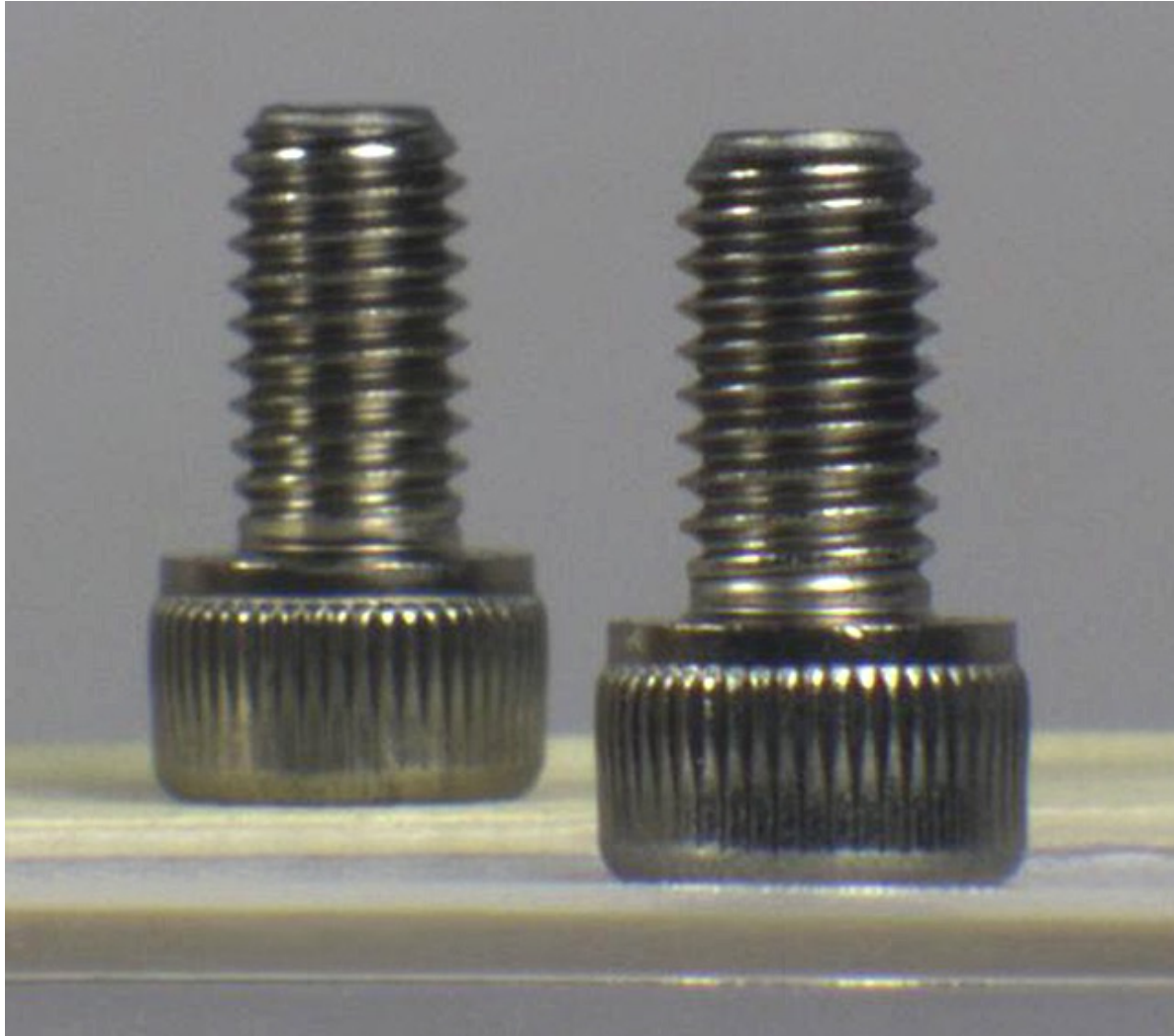
camera is *far* from  
object and has  
*large* focal length

————— increasing focal length —————>

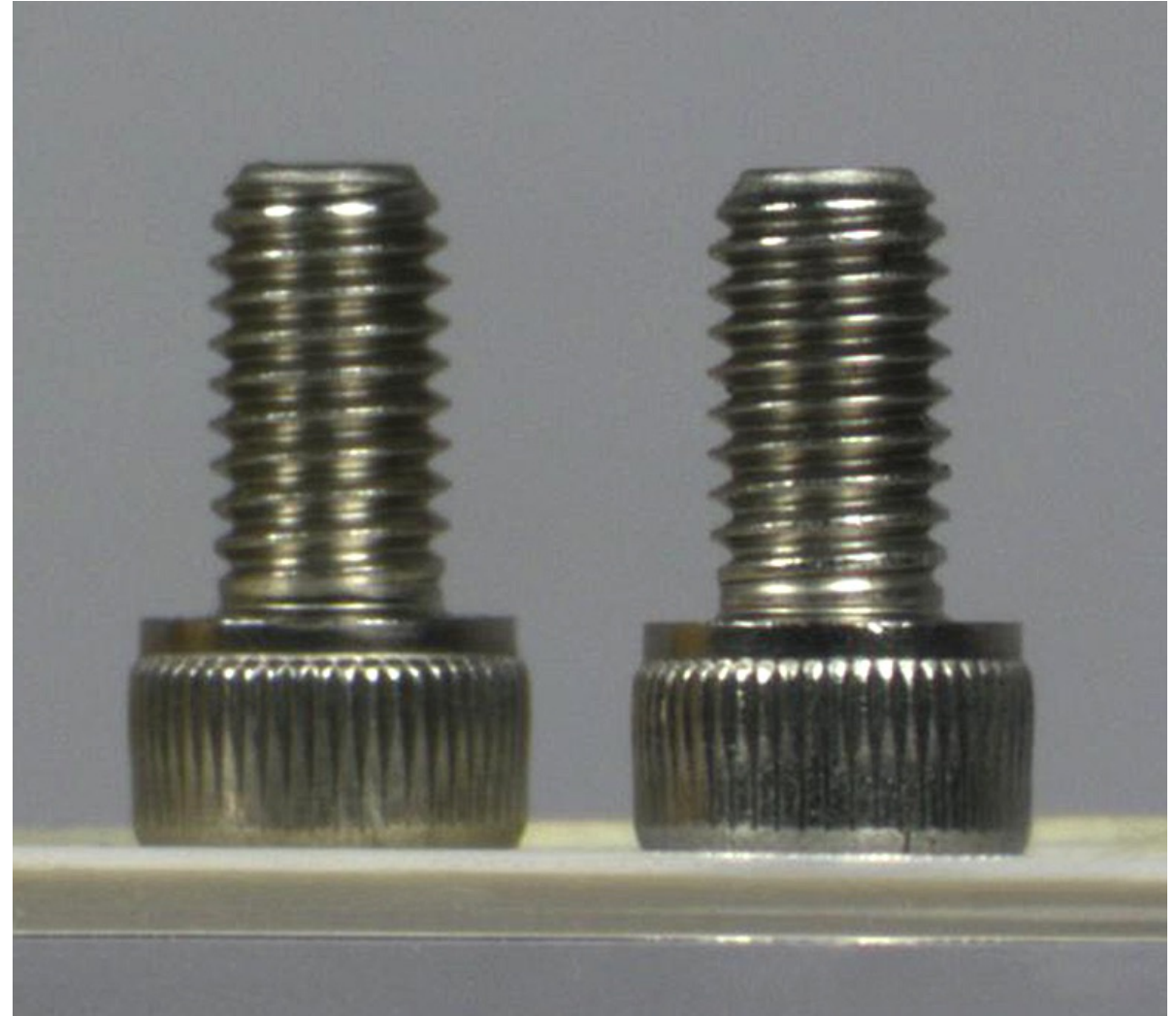
————— increasing distance from camera —————>



# Different cameras



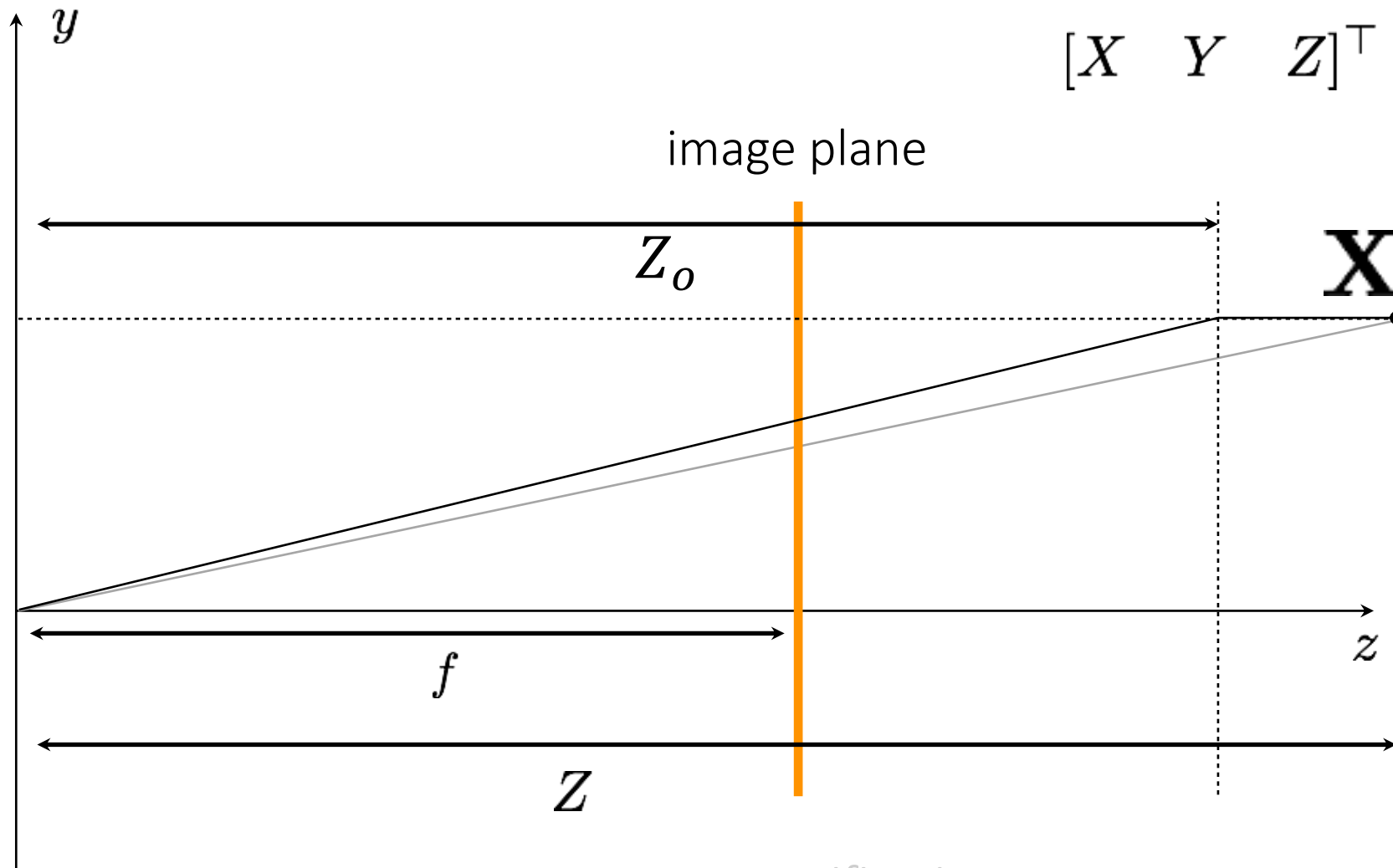
perspective camera



weak perspective camera



# Weak perspective vs perspective camera



$$[X \ Y \ Z]^T \mapsto [fX/Z_0 \ fY/Z_0]^T$$

- magnification does not change with depth
- *constant* magnification depending on  $f$  and  $Z_0$

magnification  
changes with depth

$$[X \ Y \ Z]^T \mapsto [fX/Z \ fY/Z]^T$$

# Comparing camera matrices

Let's assume that the world and camera coordinate systems are the same.

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

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

- What would the matrix of the weak perspective camera look like?

# Comparing camera matrices

Let's assume that the world and camera coordinate systems are the same.

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

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

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

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

# Comparing camera matrices

Let's assume that the world and camera coordinate systems are the same.

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

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

where we now have the more general intrinsic matrix

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

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

$$\mathbf{K} = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

In both cameras, we can incorporate extrinsic parameters same as we did before.

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.

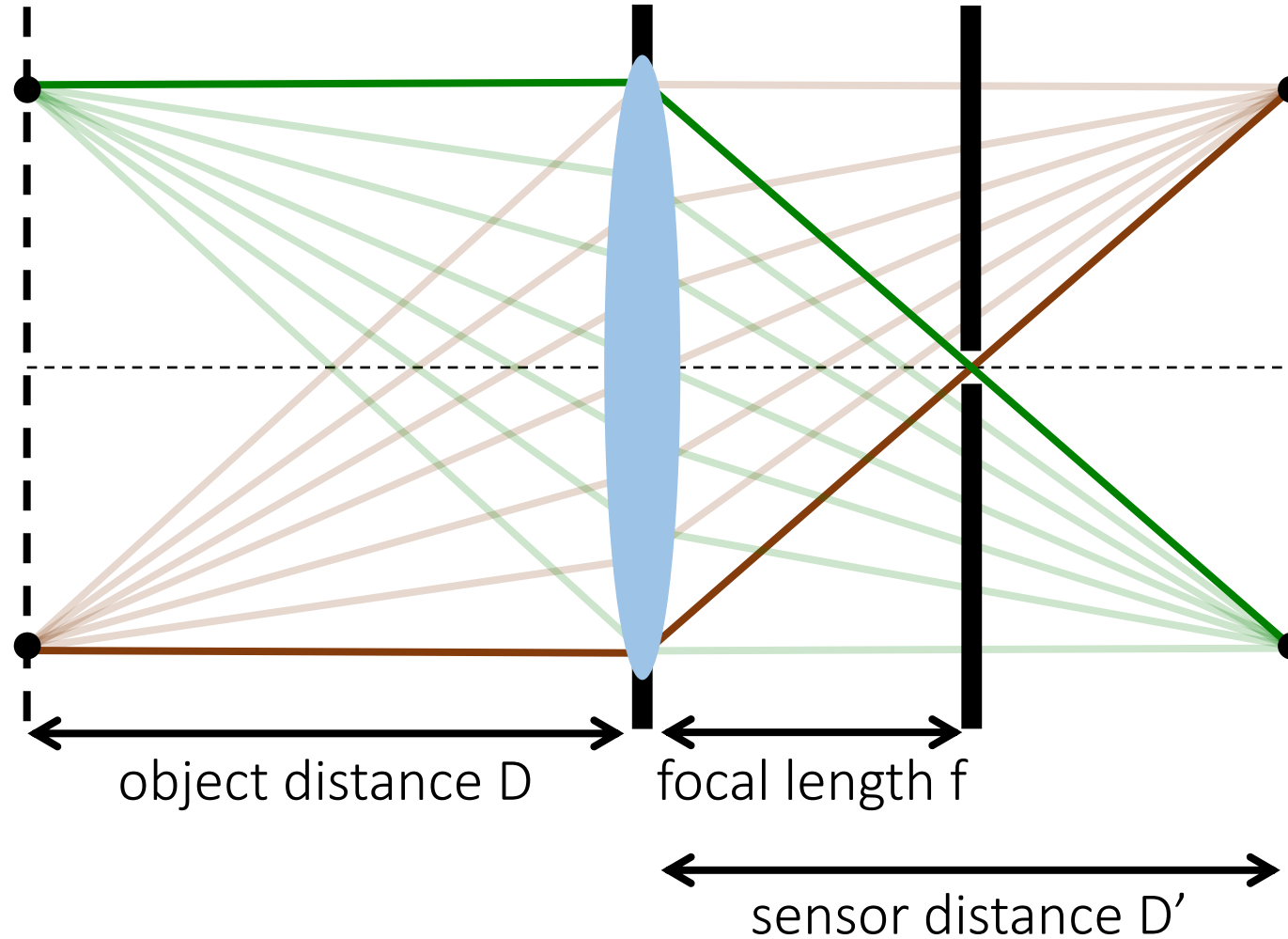


Weak perspective projection applies to the mountains.

# When can we assume a weak perspective camera?

2. When we use a telecentric lens.

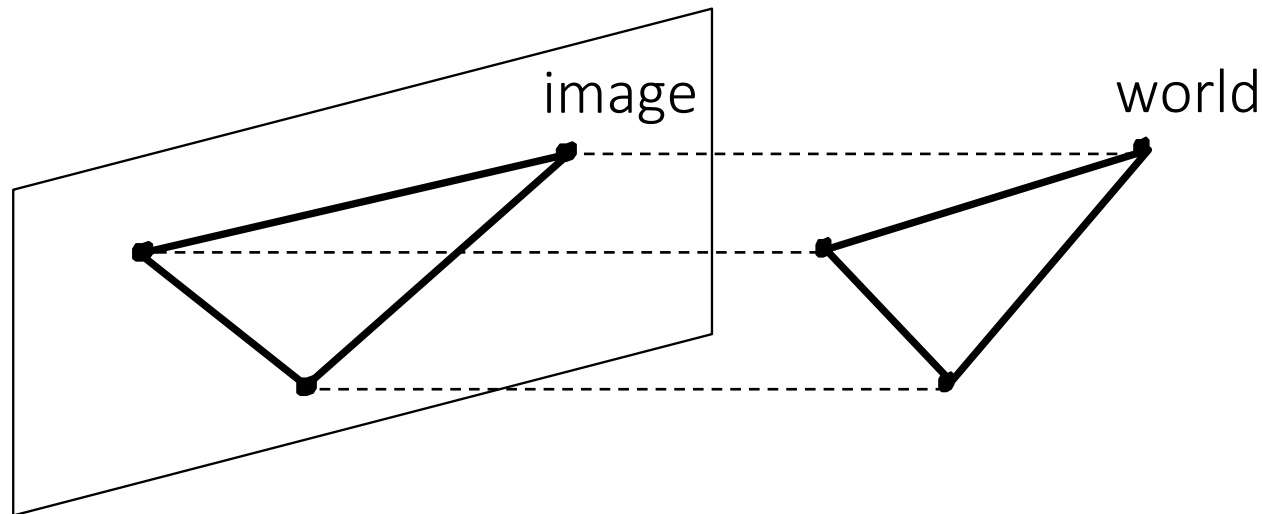
Place a pinhole at focal length, so that only rays parallel to primary ray pass through.



# Orthographic camera

Special case of weak perspective camera where:

- constant magnification is equal to 1.
- there is no shift between camera and image origins.
- the world and camera coordinate systems are the same.



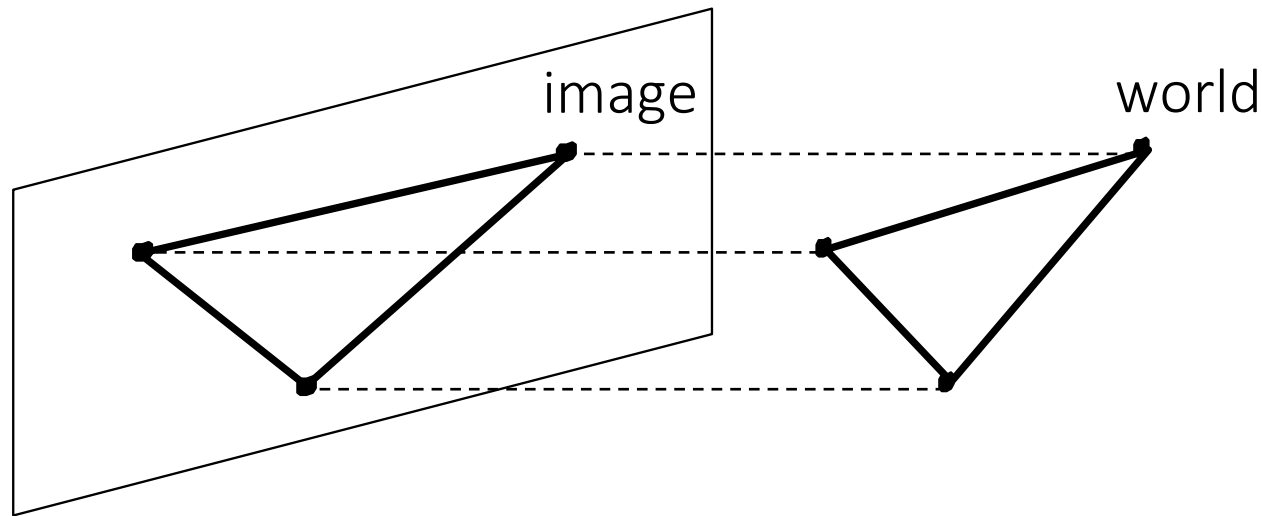
What is the camera matrix in this case?



# Orthographic camera

Special case of weak perspective camera where:

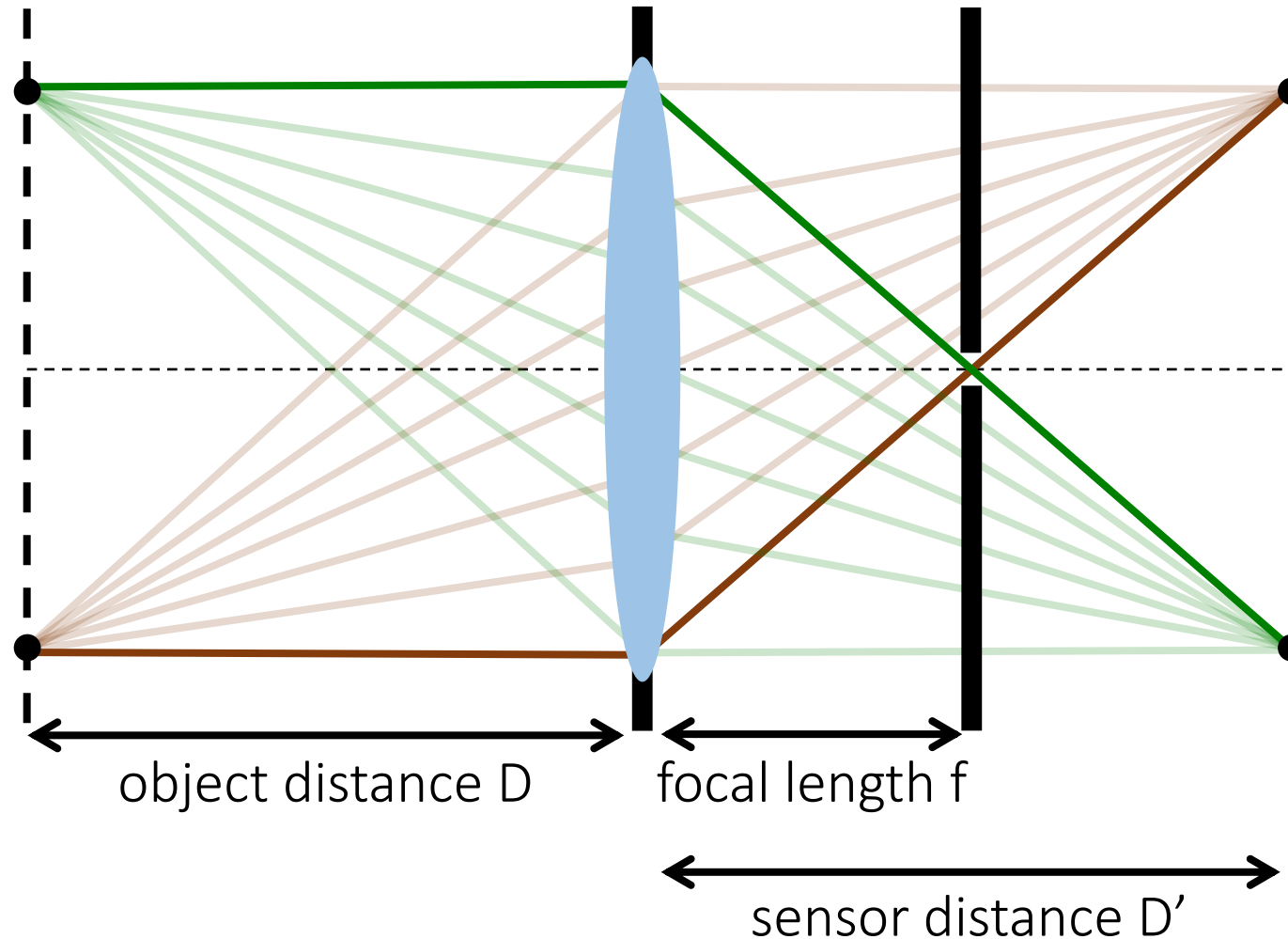
- constant magnification is equal to 1.
- there is no shift between camera and image origins.
- the world and camera coordinate systems are the same.



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

# Orthographic projection using a telecentric lens

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



We set the sensor distance as:

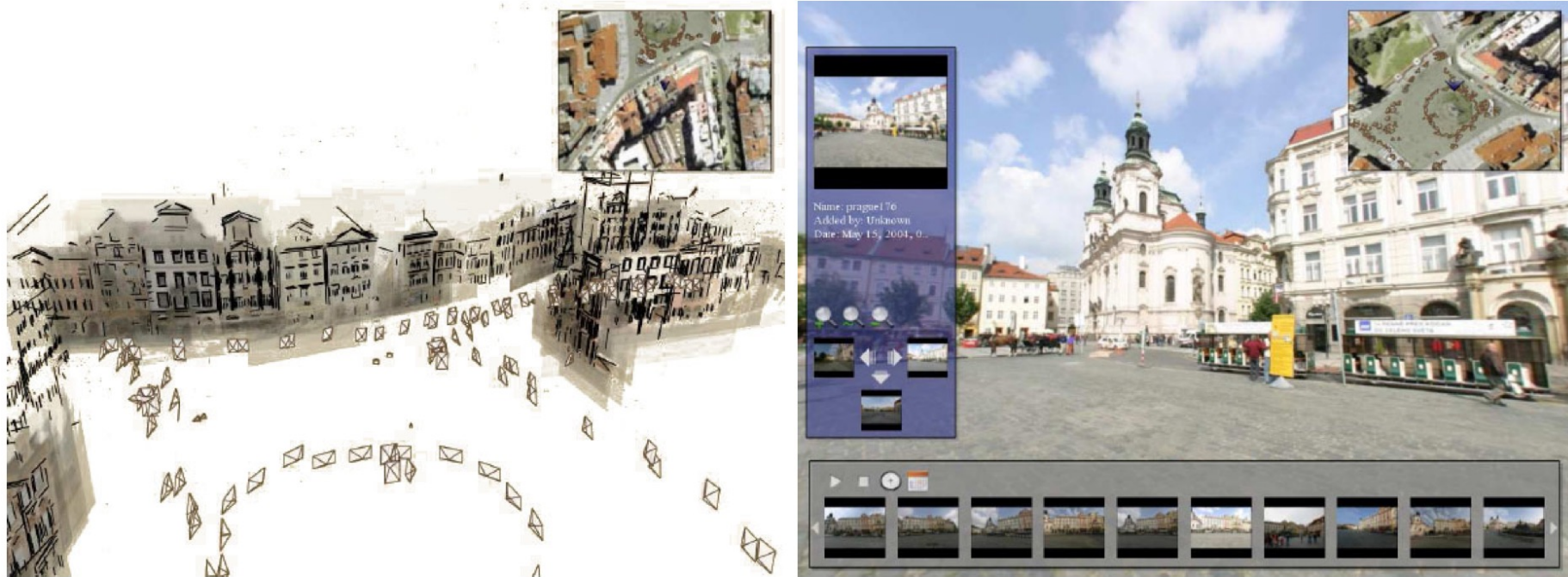
$$D' = 2f$$

in order to achieve unit magnification.

# Geometric camera calibration

	Structure (scene geometry)	Motion (camera geometry)	Measurements
Camera Calibration (a.k.a. Pose Estimation)	known	<b>estimate</b>	3D to 2D correspondences
Triangulation	<b>estimate</b>	known	2D to 2D coorespondences
Reconstruction	<b>estimate</b>	<b>estimate</b>	2D to 2D coorespondences

# Pose Estimation



Given a single image,  
estimate the exact position of the photographer

# Geometric camera calibration

Given a set of matched points

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

point in 3D  
space

point in the  
image

and camera model

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

projection  
model

parameters

Camera  
matrix

Find the (pose) estimate of

**P**

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

Same setup as homography estimation  
(slightly different derivation here)

## 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

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

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

Heterogeneous coordinates

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

(non-linear relation between coordinates)

*How can we make these relations linear?*

*How can we make these relations linear?*

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

Make them linear with algebraic manipulation...

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

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

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

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

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

*How do we proceed?*

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

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

In matrix form ...

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

*How do we proceed?*

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

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

In matrix form ...

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

For N points ...

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

*How do we solve  
this system?*

Solve for camera matrix by

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

$$\mathbf{A} = \begin{bmatrix} \mathbf{X}_1^\top & \mathbf{0} & -x' \mathbf{X}_1^\top \\ \mathbf{0} & \mathbf{X}_1^\top & -y' \mathbf{X}_1^\top \\ \vdots & \vdots & \vdots \\ \mathbf{X}_N^\top & \mathbf{0} & -x' \mathbf{X}_N^\top \\ \mathbf{0} & \mathbf{X}_N^\top & -y' \mathbf{X}_N^\top \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

**SVD!**

Solve for camera matrix by

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

$$\mathbf{A} = \begin{bmatrix} \mathbf{X}_1^\top & \mathbf{0} & -x' \mathbf{X}_1^\top \\ \mathbf{0} & \mathbf{X}_1^\top & -y' \mathbf{X}_1^\top \\ \vdots & \vdots & \vdots \\ \mathbf{X}_N^\top & \mathbf{0} & -x' \mathbf{X}_N^\top \\ \mathbf{0} & \mathbf{X}_N^\top & -y' \mathbf{X}_N^\top \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

Solution  $\mathbf{x}$  is the column of  $\mathbf{V}$   
corresponding to smallest singular  
value of

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

Solve for camera matrix by

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

$$\mathbf{A} = \begin{bmatrix} \mathbf{X}_1^\top & \mathbf{0} & -x' \mathbf{X}_1^\top \\ \mathbf{0} & \mathbf{X}_1^\top & -y' \mathbf{X}_1^\top \\ \vdots & \vdots & \vdots \\ \mathbf{X}_N^\top & \mathbf{0} & -x' \mathbf{X}_N^\top \\ \mathbf{0} & \mathbf{X}_N^\top & -y' \mathbf{X}_N^\top \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

Equivalently, solution  $\mathbf{x}$  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

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

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

## Decomposition of the Camera Matrix

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

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

Find the camera center  $\mathbf{C}$

*What is the projection of the camera center?*

Find intrinsic  $\mathbf{K}$  and rotation  $\mathbf{R}$

## Decomposition of the Camera Matrix

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

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

Find the camera center  $\mathbf{c}$

$$\mathbf{P}\mathbf{c} = \mathbf{0}$$

*How do we compute the camera center from this?*

Find intrinsic  $\mathbf{K}$  and rotation  $\mathbf{R}$

## Decomposition of the Camera Matrix

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

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

Find the camera center  $\mathbf{c}$

$$\mathbf{P}\mathbf{c} = \mathbf{0}$$

**SVD** of  $\mathbf{P}$ !

*$\mathbf{c}$  is the singular vector corresponding to the smallest singular value*

Find intrinsic  $\mathbf{K}$  and rotation  $\mathbf{R}$



## Decomposition of the Camera Matrix

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

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

Find the camera center  $\mathbf{c}$

$$\mathbf{P}\mathbf{c} = \mathbf{0}$$

SVD of  $\mathbf{P}$ !

*$\mathbf{c}$  is the singular vector corresponding to the smallest singular value*

Find intrinsic  $\mathbf{K}$  and rotation  $\mathbf{R}$

$$\mathbf{M} = \mathbf{K}\mathbf{R}$$

*Any useful properties of  $\mathbf{K}$  and  $\mathbf{R}$  we can use?*

# Decomposition of the Camera Matrix

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

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

Find the camera center  $\mathbf{c}$

$$\mathbf{P}\mathbf{c} = \mathbf{0}$$

SVD of P!

$\mathbf{c}$  is the singular vector corresponding to the smallest singular value

Find intrinsic  $\mathbf{K}$  and rotation  $\mathbf{R}$

$$\mathbf{M} = \mathbf{K}\mathbf{R}$$

right upper triangle      orthogonal

*How do we find K and R?*

## Decomposition of the Camera Matrix

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

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

Find the camera center  $\mathbf{c}$

$$\mathbf{P}\mathbf{c} = \mathbf{0}$$

SVD of  $\mathbf{P}$ !

*$\mathbf{c}$  is the singular vector corresponding to the smallest singular value*

Find intrinsic  $\mathbf{K}$  and rotation  $\mathbf{R}$

$$\mathbf{M} = \mathbf{K}\mathbf{R}$$

QR decomposition

# Geometric camera calibration

Given a set of matched points

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

point in 3D space    point in the image

*Where do we get these matched points from?*

and camera model

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

projection model

parameters

Camera matrix

Find the (pose) estimate of

# $\mathbf{P}$

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

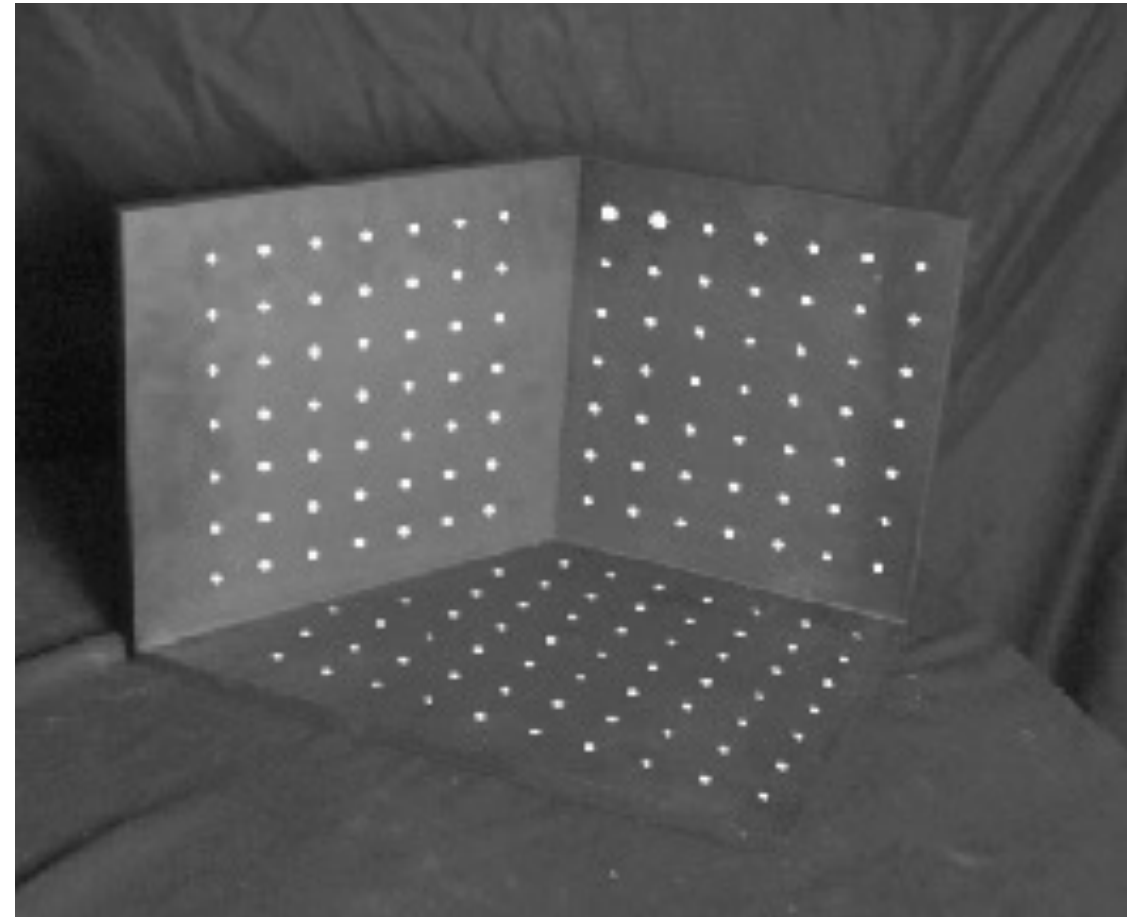
# Calibration using a reference object

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.

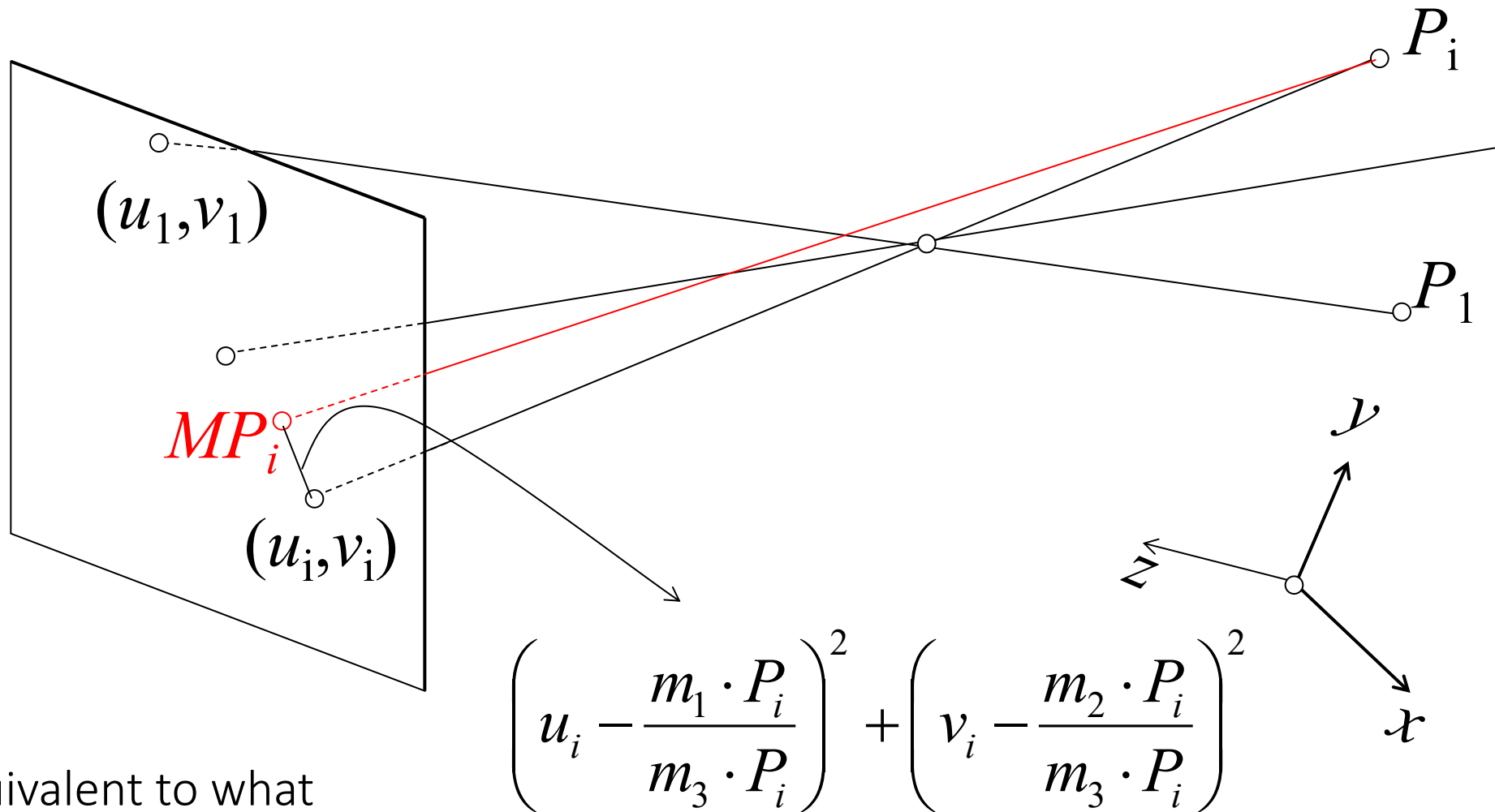
## Disadvantages:

- 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

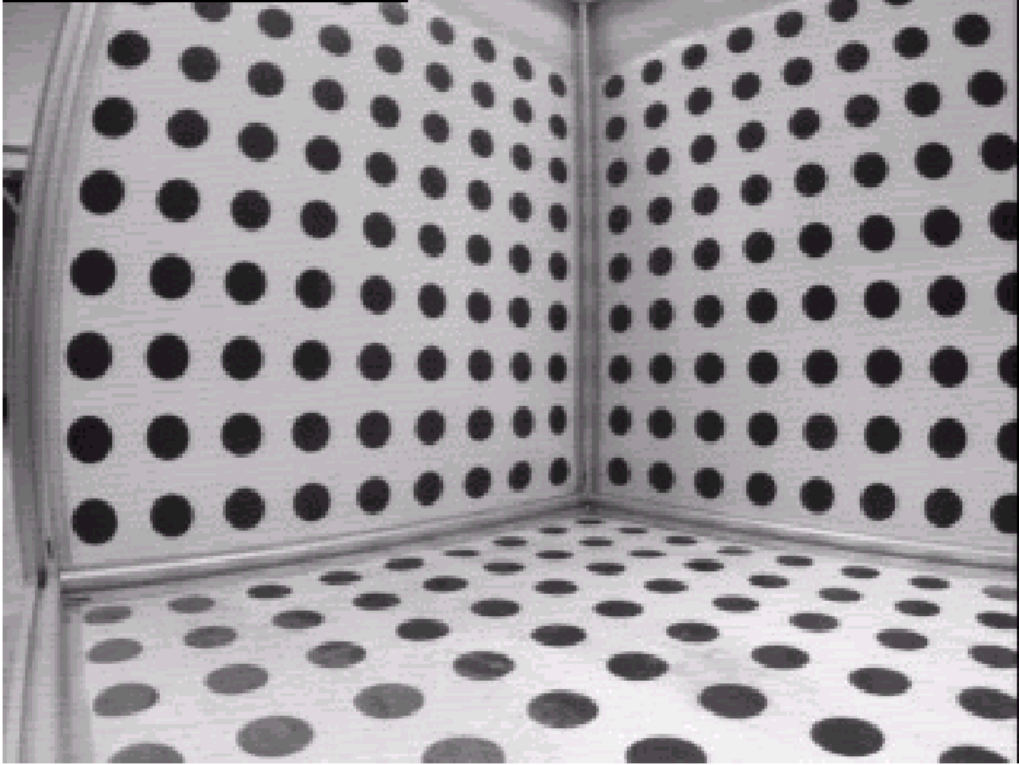
- Define error function  $E$  between projected 3D points and image positions
  - $E$  is nonlinear function of intrinsics, extrinsics, radial distortion
- Minimize  $E$  using nonlinear optimization techniques

# Minimizing reprojection error

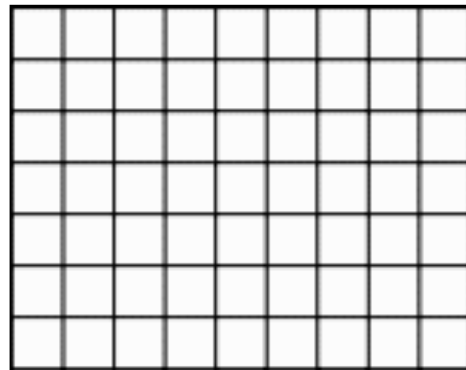


Is this equivalent to what we were doing previously?

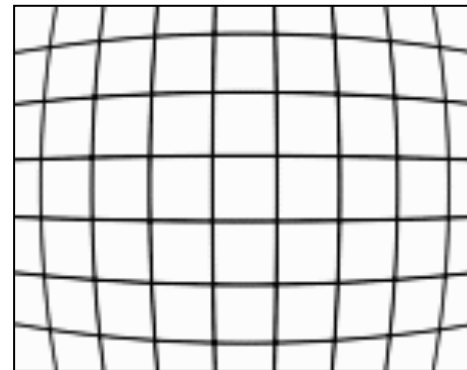
# Radial distortion



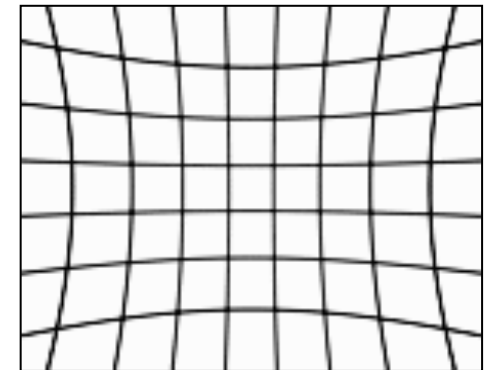
What causes this distortion?



no distortion



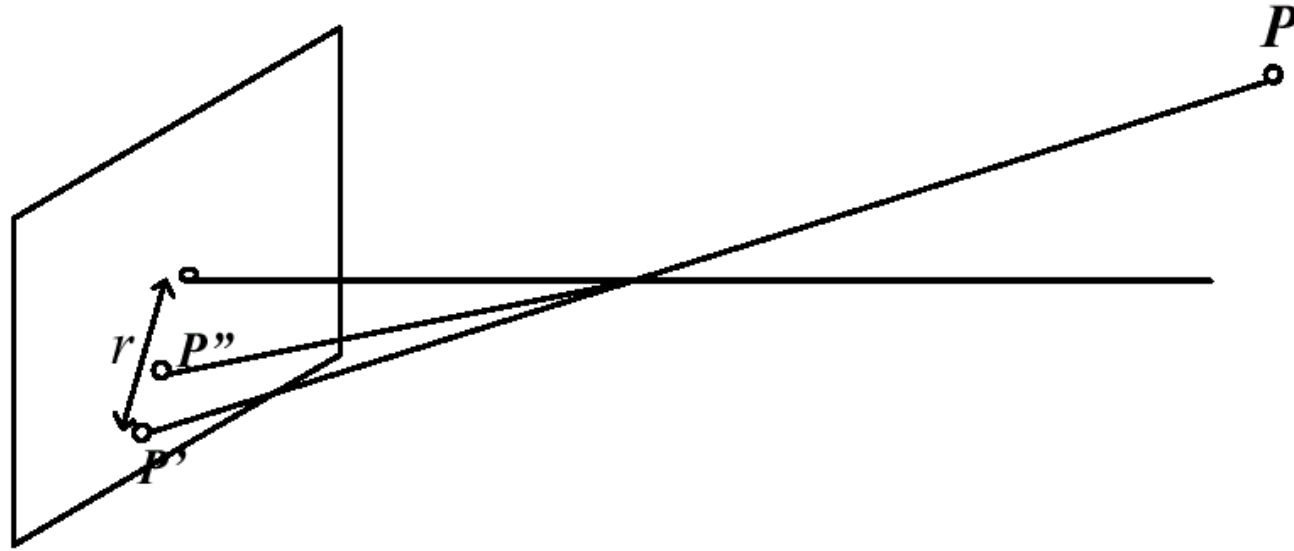
barrel distortion



pincushion distortion



# Radial distortion model



Ideal:

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

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

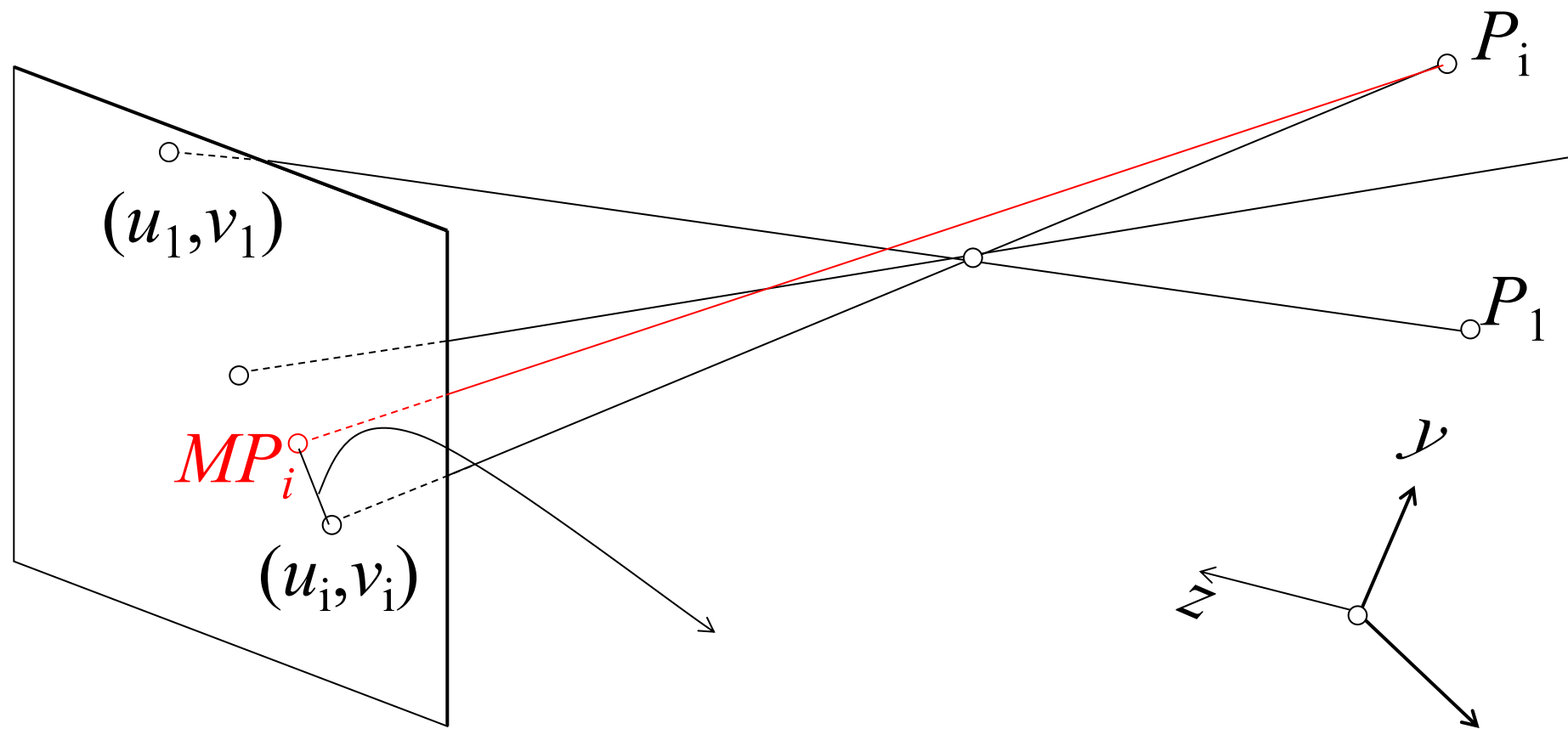
Distorted:

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

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

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

# 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

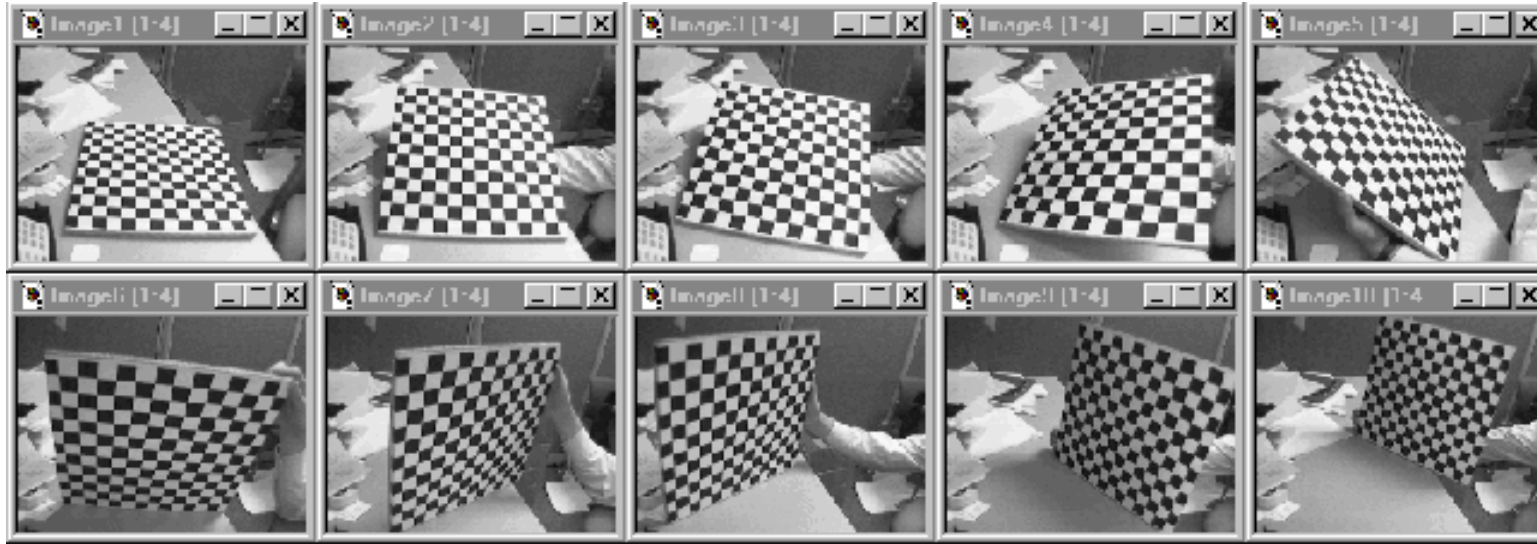


before



after

# Alternative: Multi-plane calibration



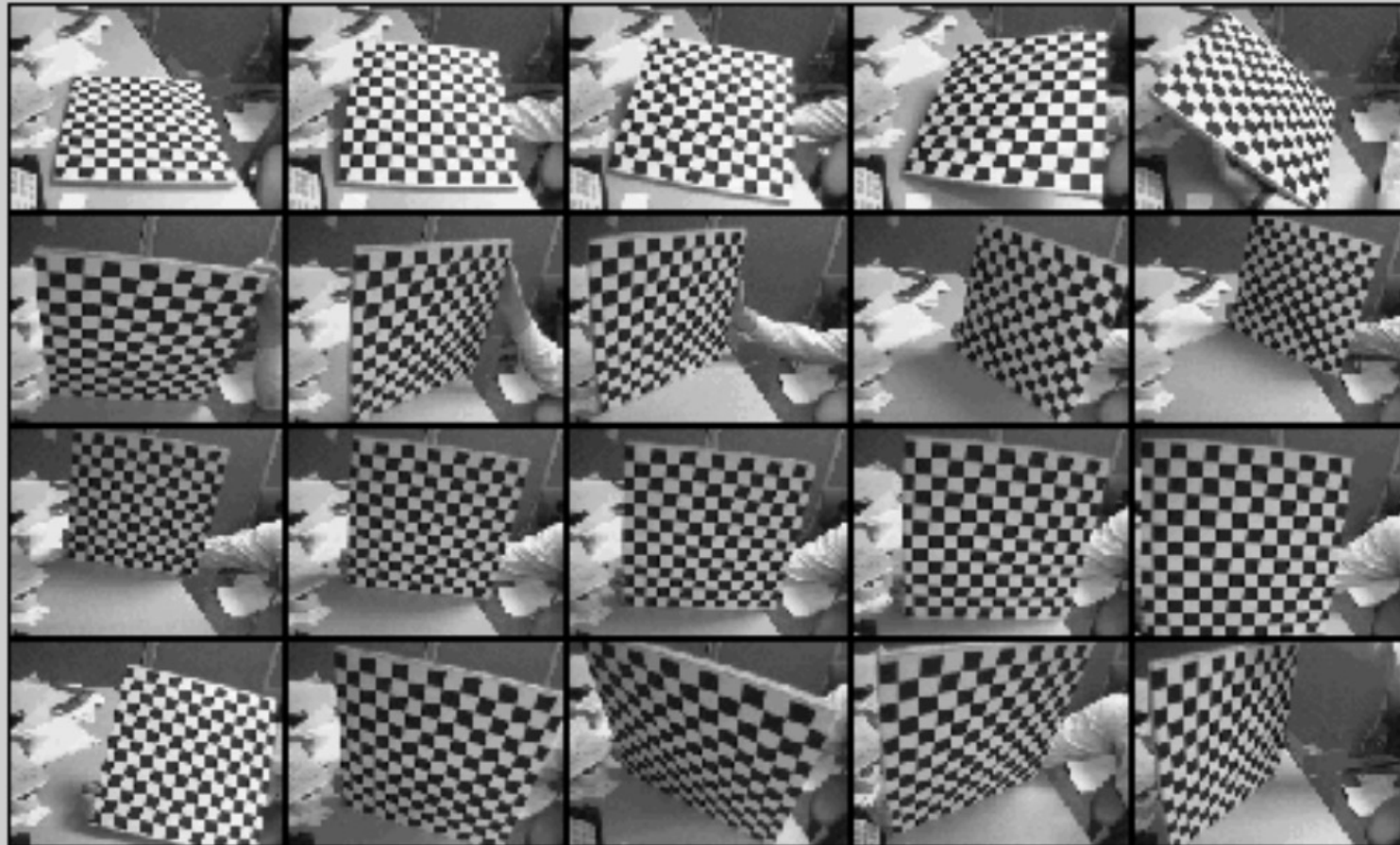
## Advantages:

- 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](http://www.vision.caltech.edu/bouguetj/calib_doc/index.html)
  - Also available on OpenCV.

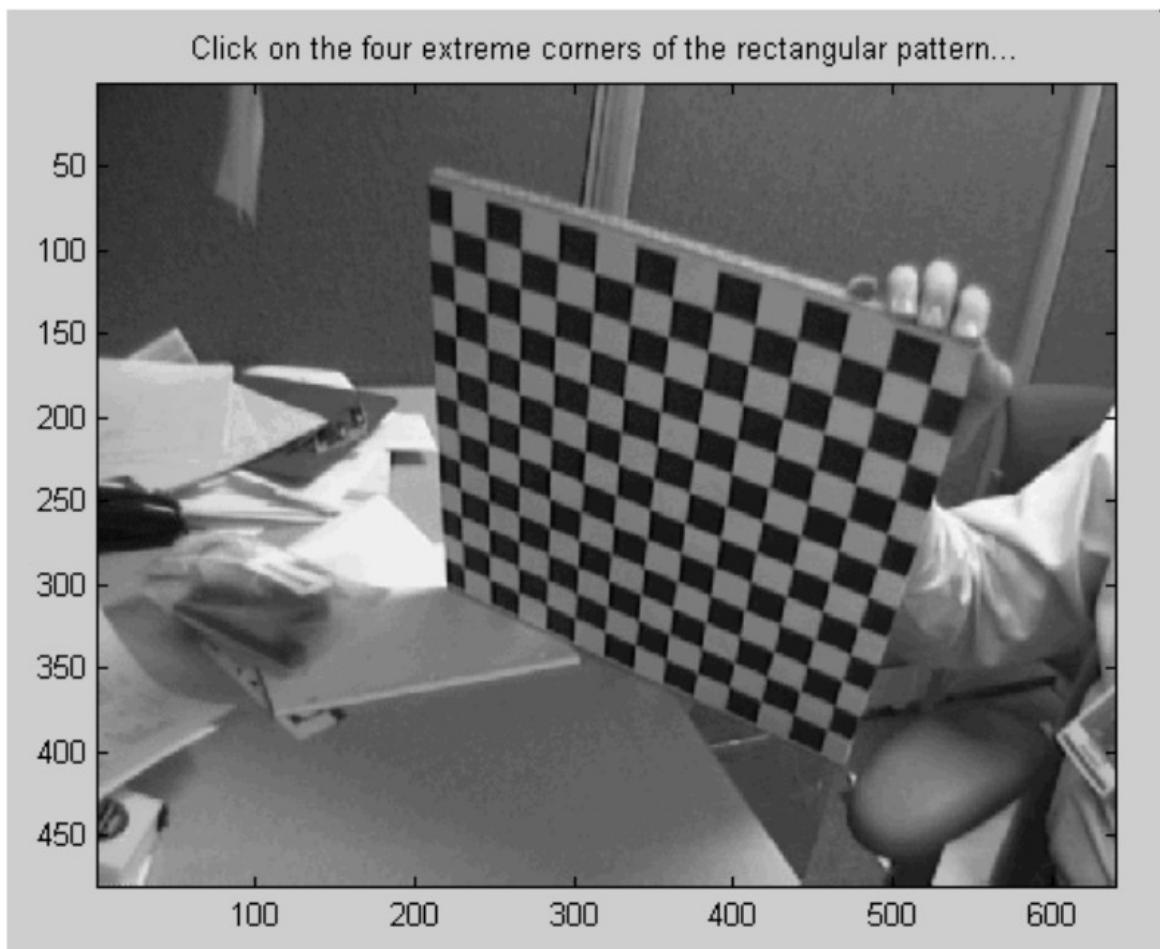
Disadvantage: Need to solve non-linear optimization problem.

# Step-by-step demonstration

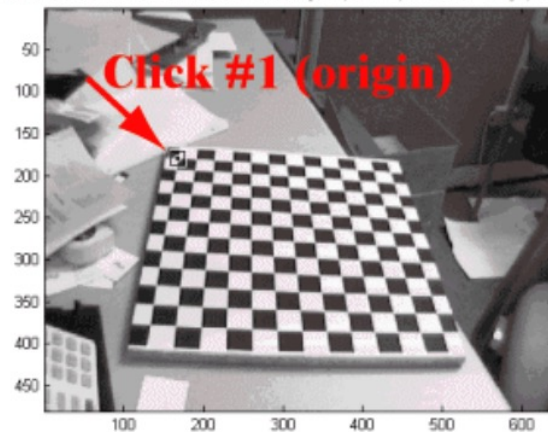
Calibration images



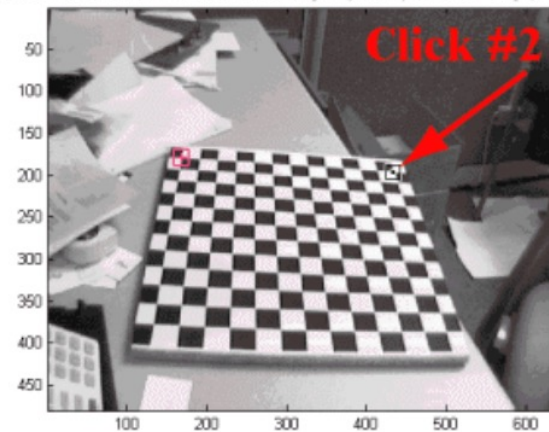
# Step-by-step demonstration



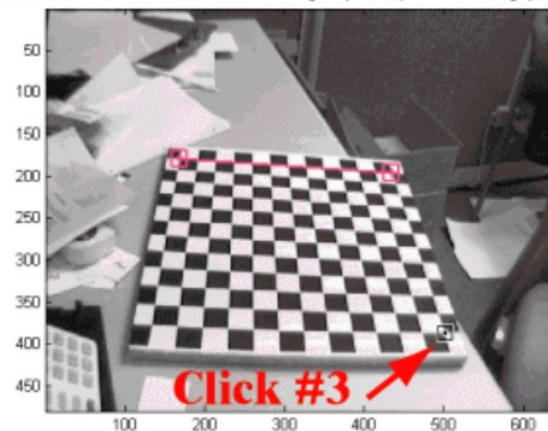
Click on the four extreme corners of the rectangular pattern (first corner = origin)... Image 1



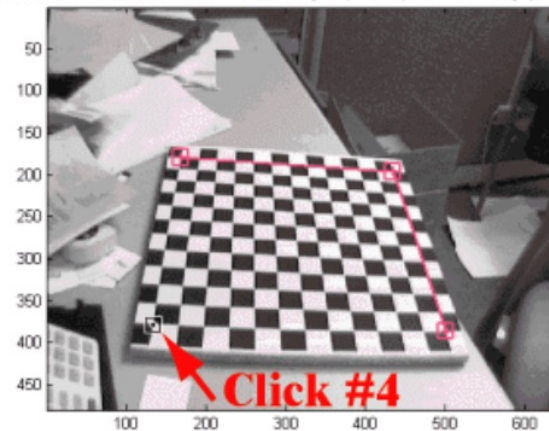
Click on the four extreme corners of the rectangular pattern (first corner = origin)... Image 1



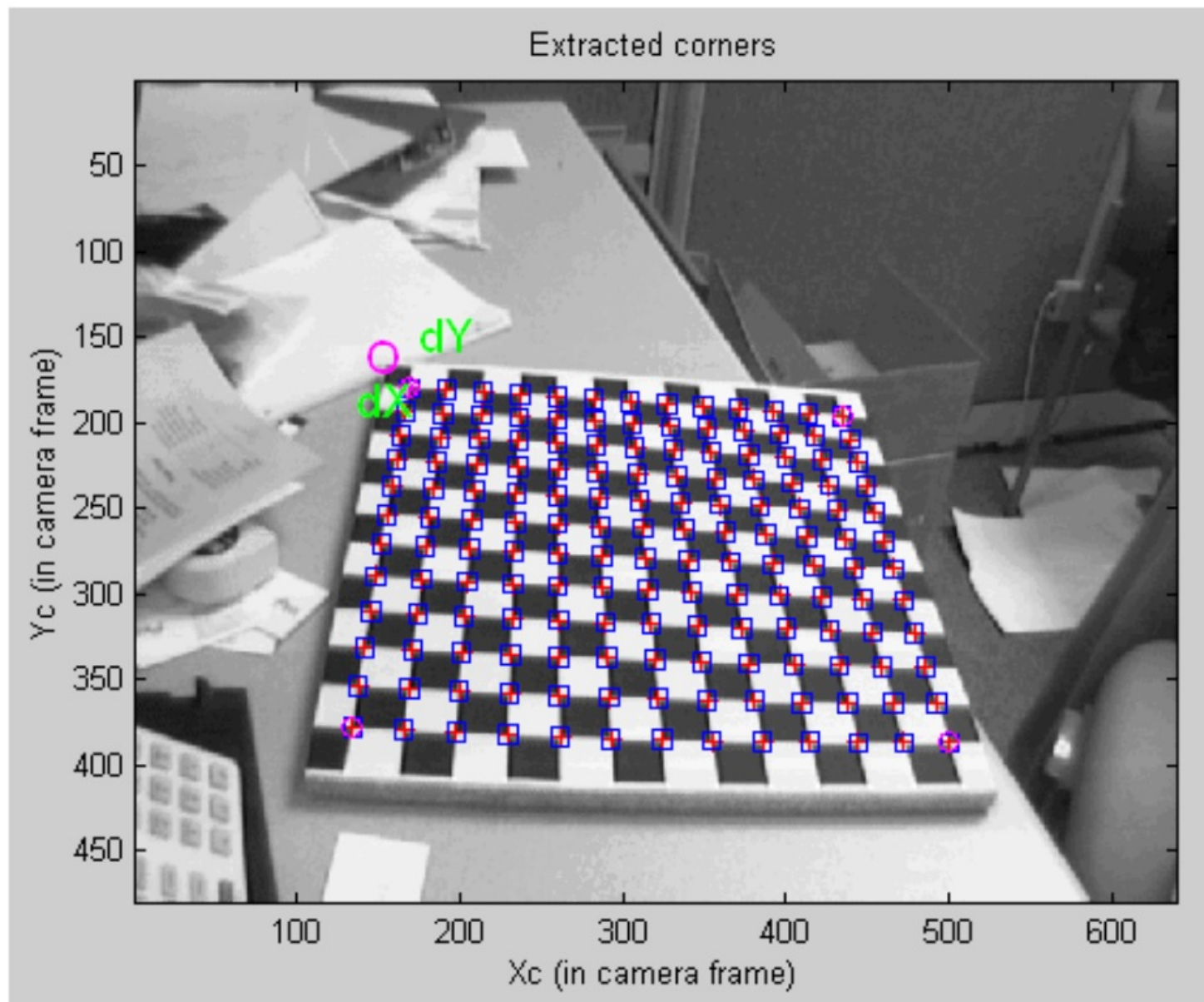
Click on the four extreme corners of the rectangular pattern (first corner = origin)... Image 1



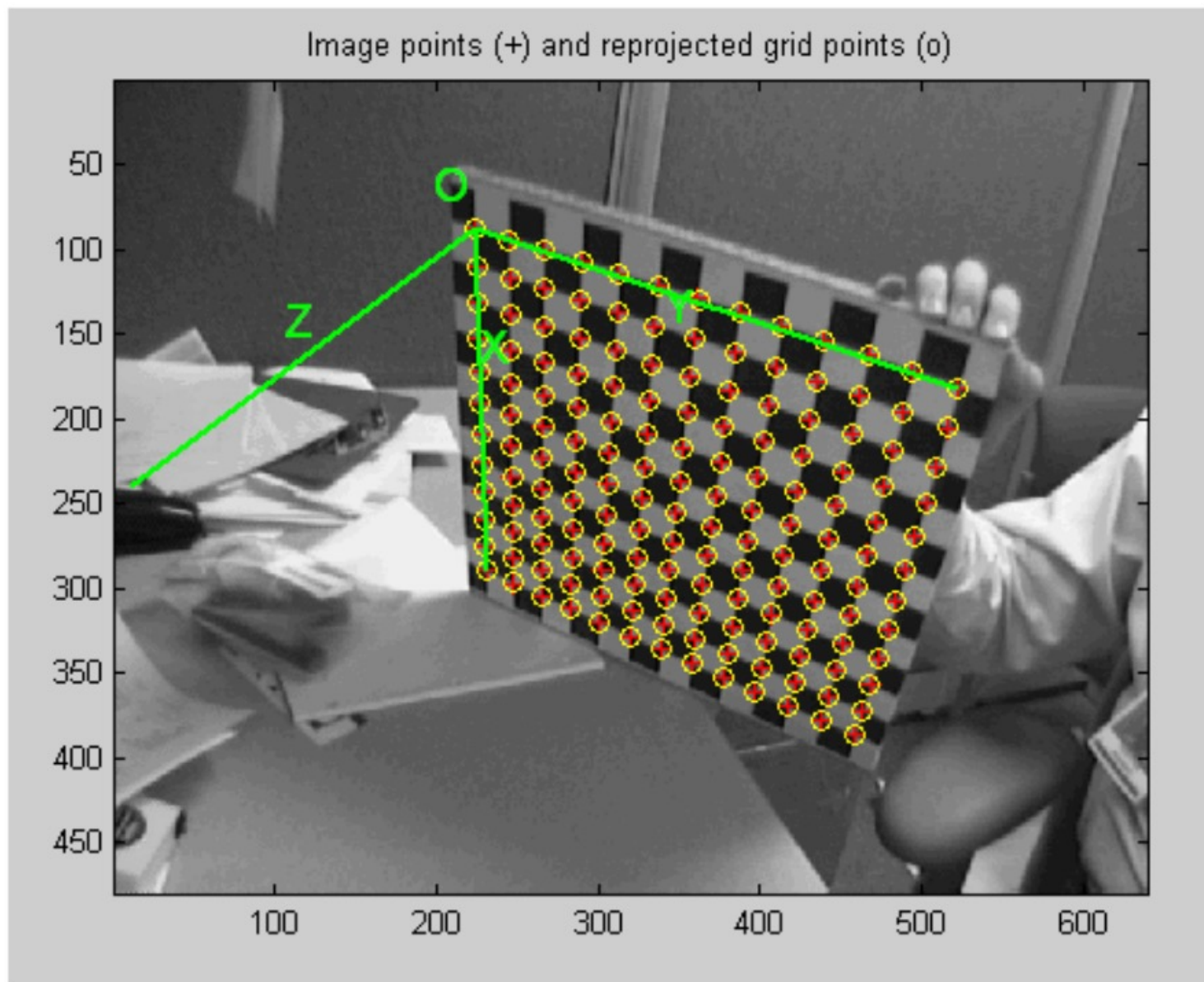
Click on the four extreme corners of the rectangular pattern (first corner = origin)... Image 1



# Step-by-step demonstration

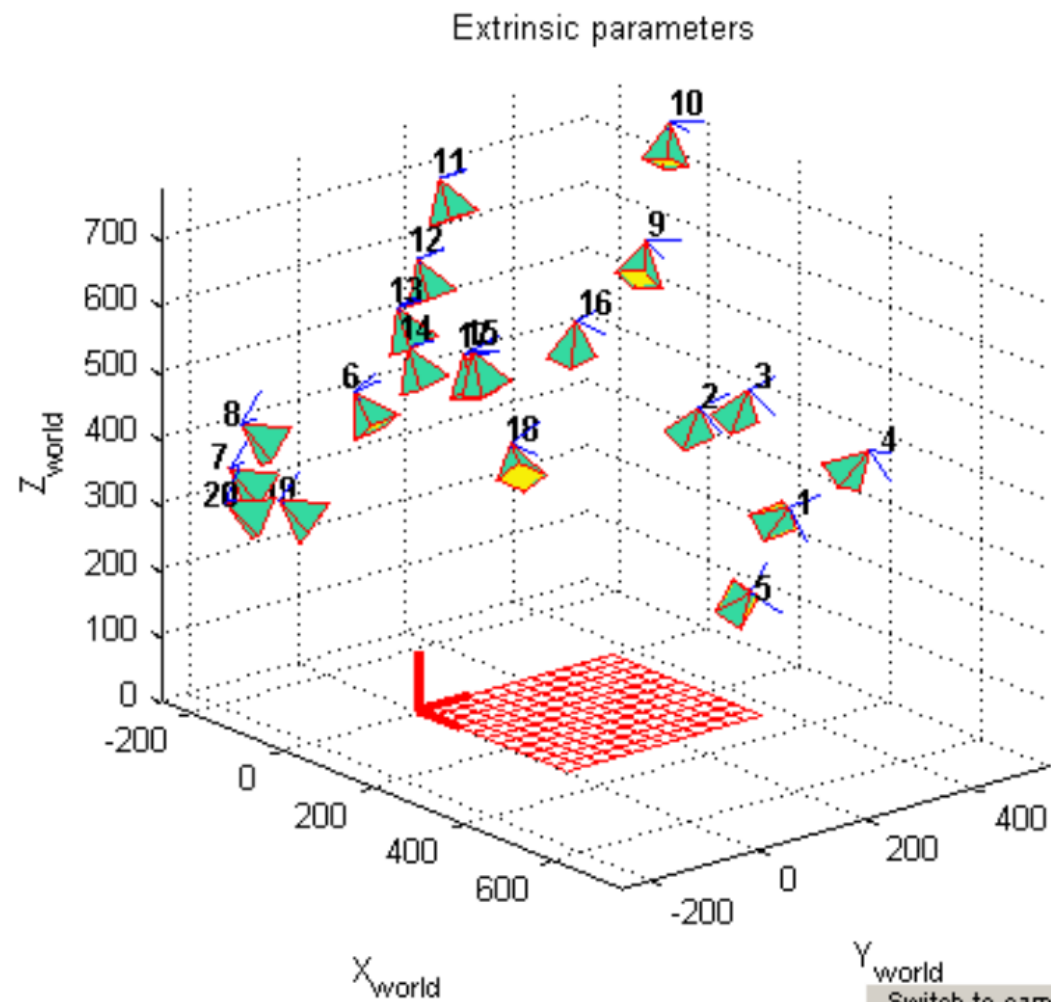
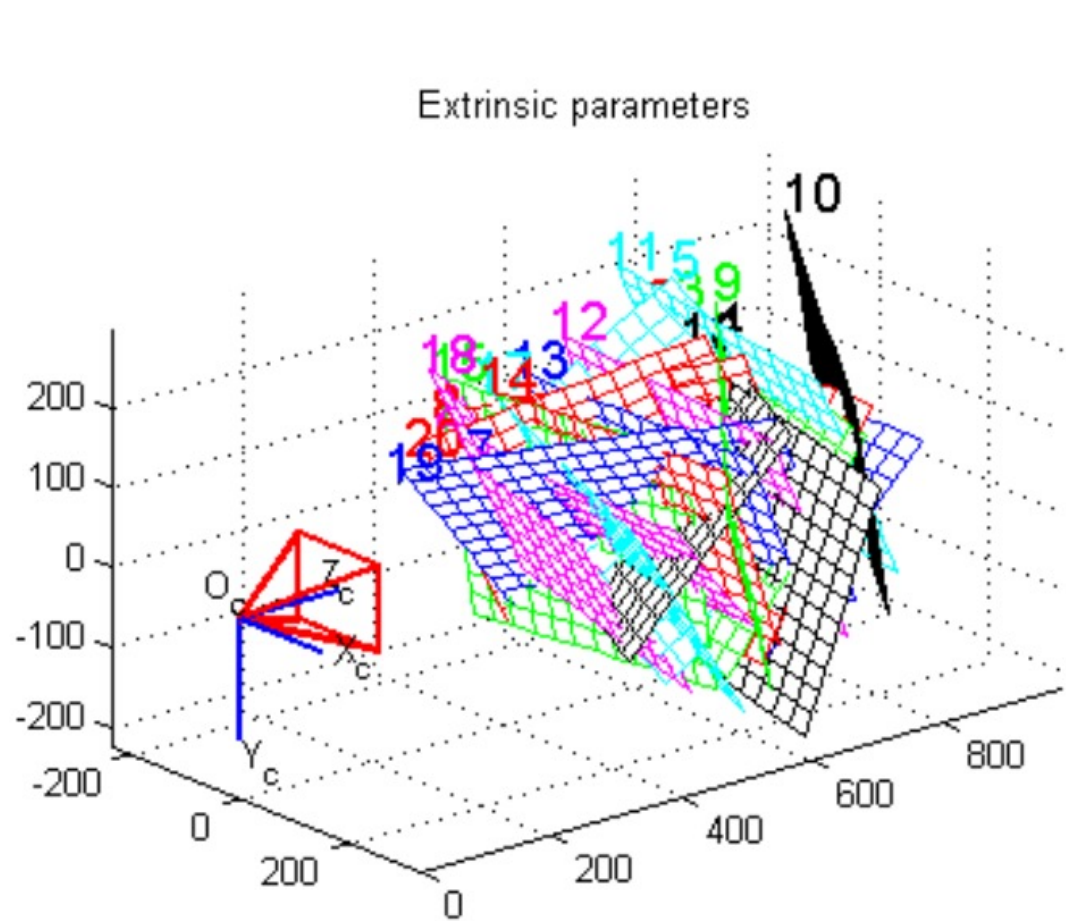


# Step-by-step demonstration





# Step-by-step demonstration



Switch to camera-centered view

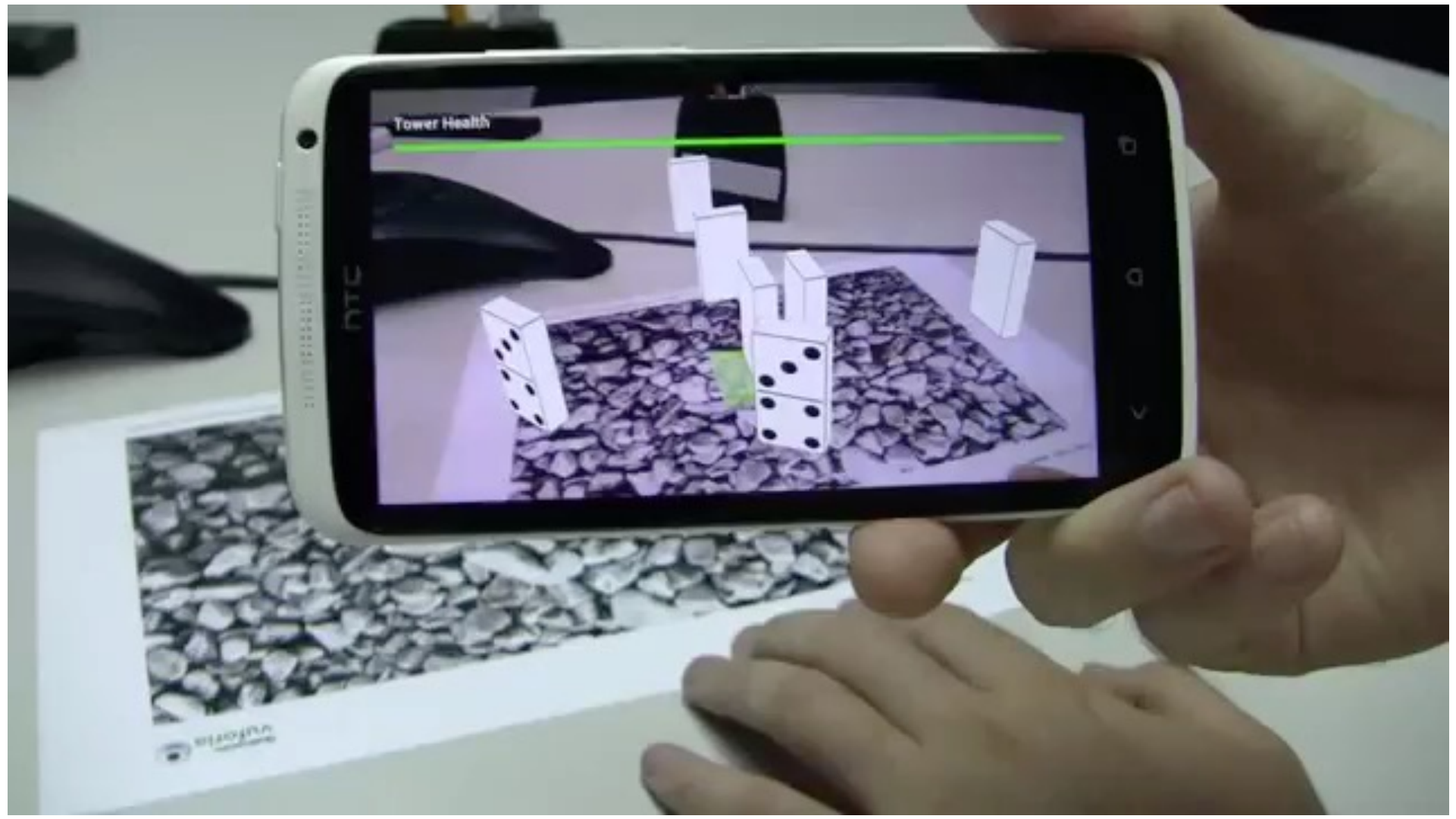
What does it mean to “calibrate a camera”?

# What does it mean to “calibrate a camera”?

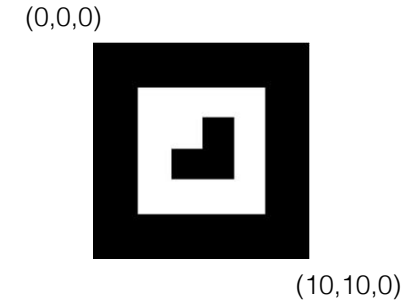
Many different ways to calibrate a camera:

- Radiometric calibration.
- Color calibration.
- Geometric calibration.
- Noise calibration.
- Lens (or aberration) calibration.

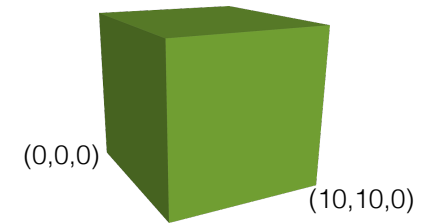
We'll briefly discuss radiometric and color calibration in later lectures. For the rest, see 15-463/663/862.



3D locations of planar marker features are known in advance



3D content prepared in advance



## Simple AR program

1. Compute point correspondences (2D and AR tag)
2. Estimate the pose of the camera **P**
3. Project 3D content to image plane using **P**

