AI70601/CSE61101

3D Vision and Machine Perception

Prof. Kyungdon Joo

3D Vision & Robotics Lab.

AI Graduate School (AIGS) & Computer Science and Engineering (CSE)

Recap: The camera as a coordinate transformation

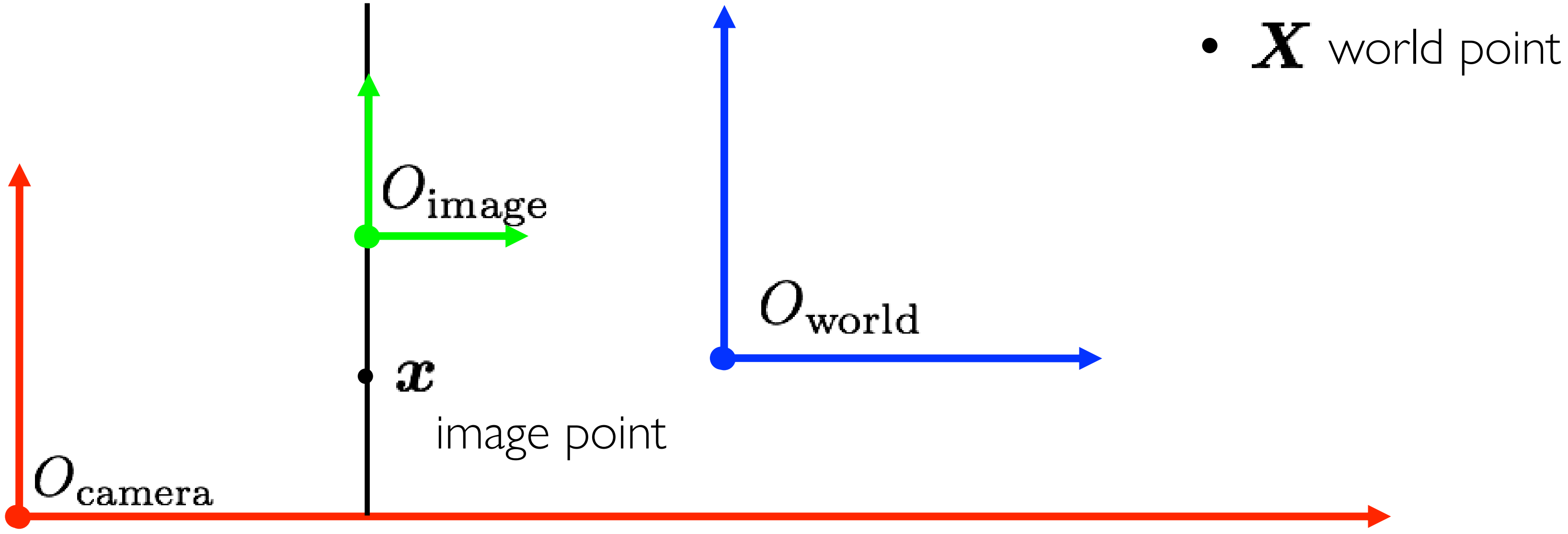
$$\mathbf{x} = \mathbf{P}\mathbf{X}$$

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

homogeneous image coordinates 3 x 1	camera matrix 3 x 4	homogeneous world coordinates 4 x 1

Recap: Generalizing the camera matrix

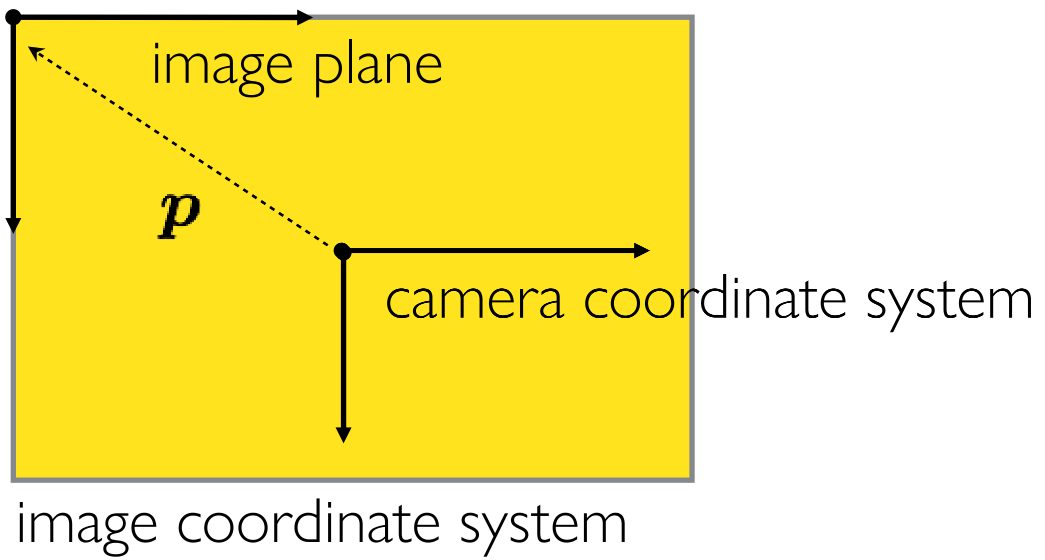
- In general, there are three, generally different, coordinate systems



- We need to know the transformations between them.

Recap: Generalizing the camera matrix

- In particular, the camera origin and image origin may be different:

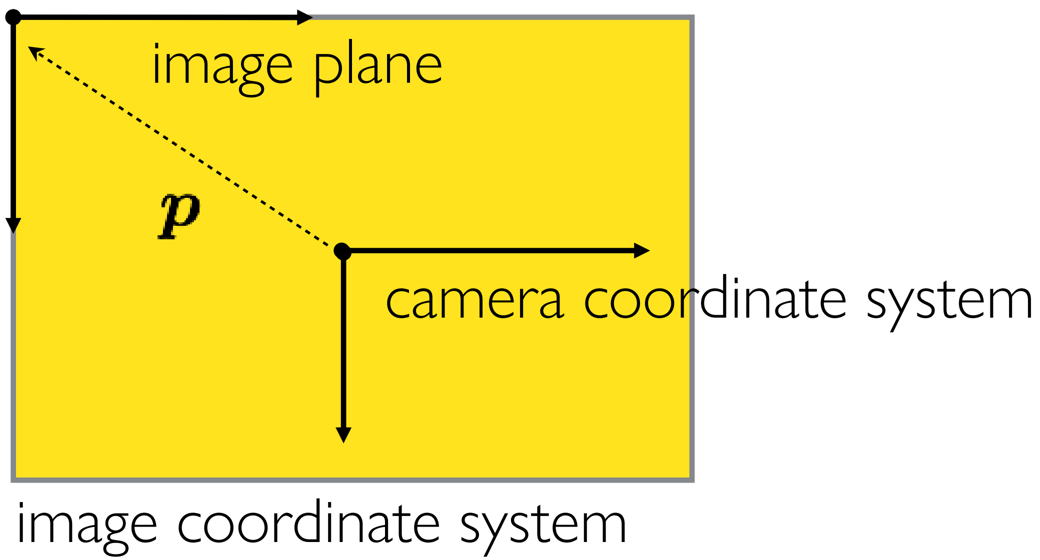


- How does the camera matrix change?

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

Recap: Generalizing the camera matrix

- In particular, the camera origin and image origin may be different:



- How does the camera matrix change?

shift vector transforming camera origin to image origin

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

Recap: Camera matrix decomposition

- We can decompose the camera matrix like this:

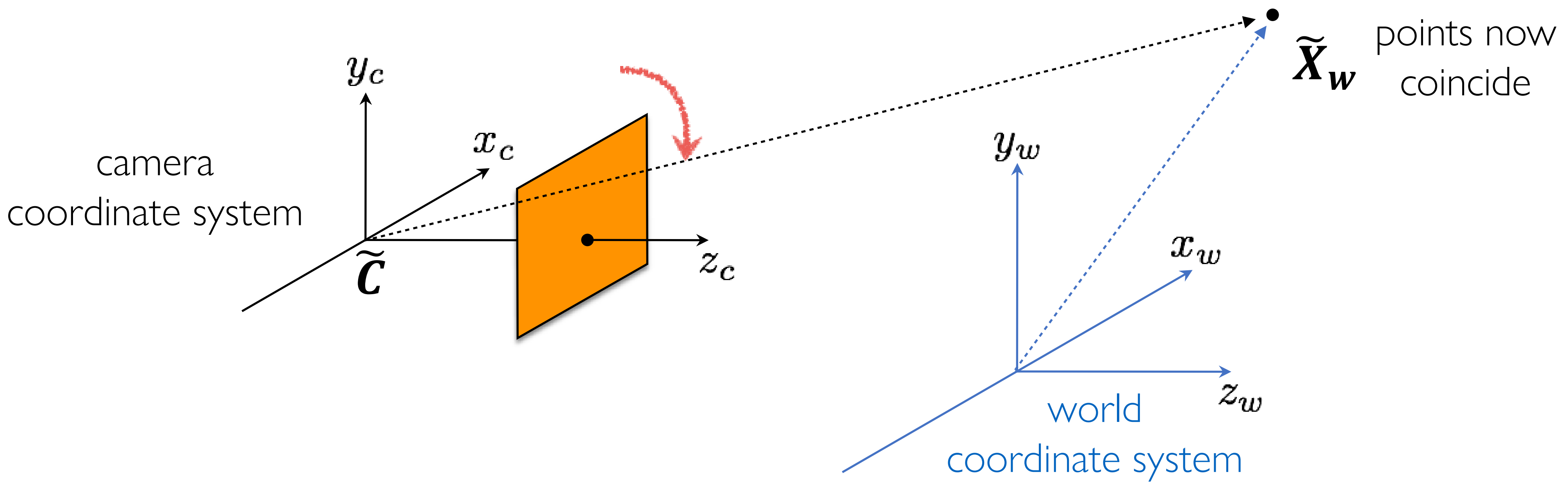
$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc:c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right]$$

(homogeneous) transformation from 2D to 2D, accounting for not 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: $\mathbf{P} = \mathbf{K}[\mathbf{I}|\mathbf{0}]$ where $\mathbf{K} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}$

Recap: World-to-camera coordinate system transformation



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

rotate translate

Recap: Modeling the coordinate system transformation

- In heterogeneous coordinates, we have:

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

- In homogeneous coordinates, we have:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{RC} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$ or $$\mathbf{X_c} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{X_w}$$

Recap: Incorporating the transform in the camera matrix

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

$$\mathbf{x} = \mathbf{P}\mathbf{X_c} = \mathbf{K}[\mathbf{I}|\mathbf{0}]\mathbf{X_c}$$

- We also just derived:

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

Recap: Putting it all together

- We can write everything into a single projection:

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

- The camera matrix now looks like:

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

intrinsic and extrinsic parameters

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{I} \quad | \quad \mathbf{0}] \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix}$$

intrinsic parameters (3 x 3):
correspond to camera internals
(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
(world-to-camera transformation)

Perspective distortion

Finite projective camera

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

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

Finite projective camera

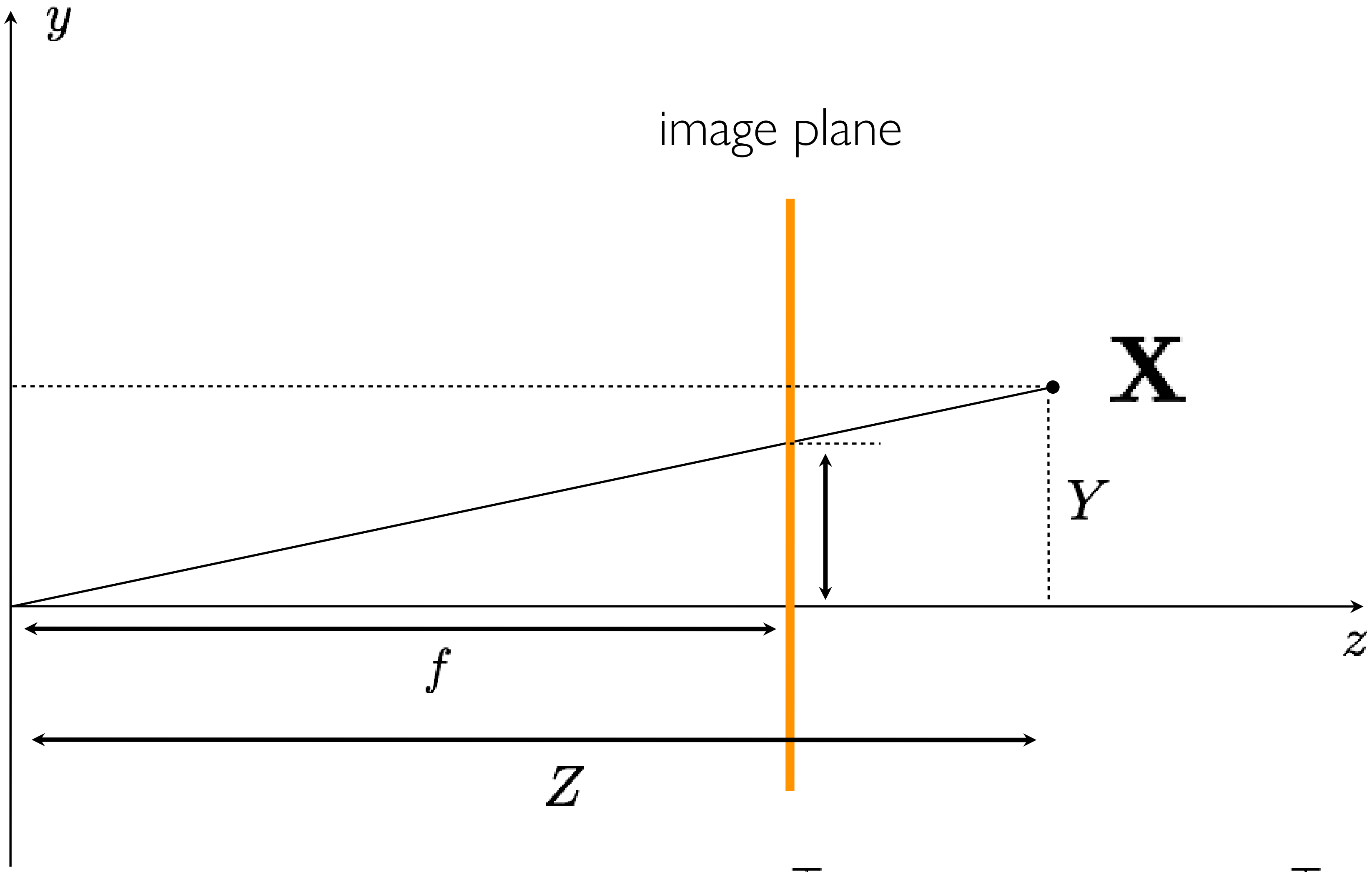
- The pinhole camera and all of the more general cameras we have seen so far have “perspective distortion”.

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

perspective projection from
(homogeneous) 3D to 2D coordinates

The 2D view of the (rearranged) pinhole camera

- Perspective projection in 2D



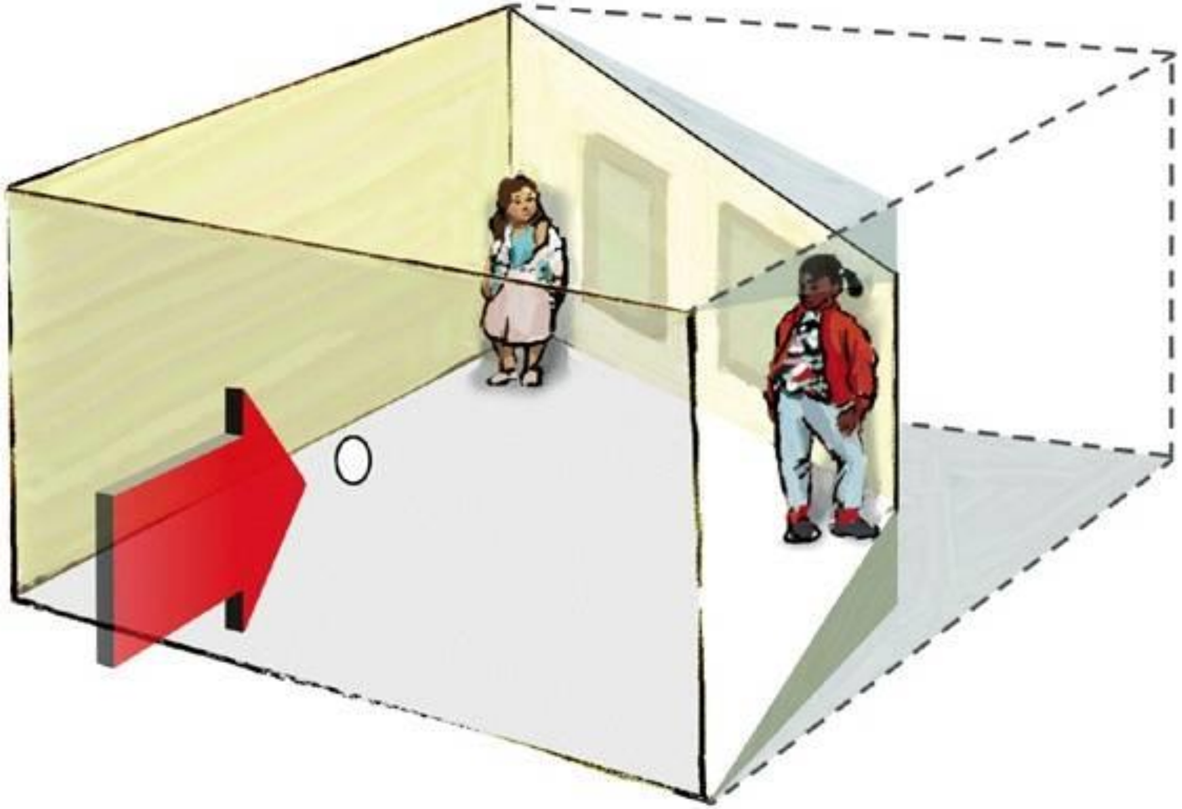
perspective distortion:
magnification changes
with depth

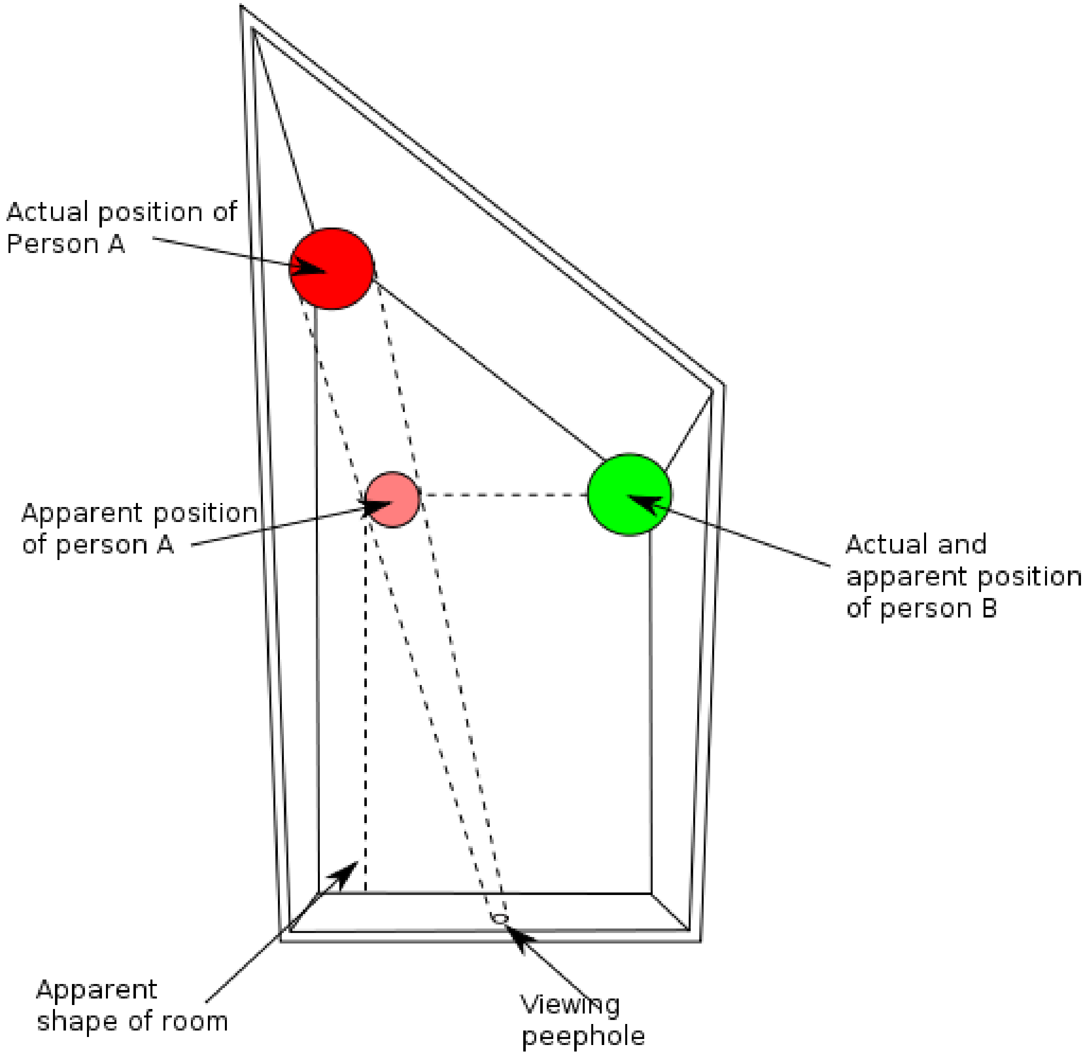
$$[X \quad Y \quad Z]^\top \mapsto [fX/Z \quad fY/Z]^\top$$

Forced perspective

The Ames room illusion

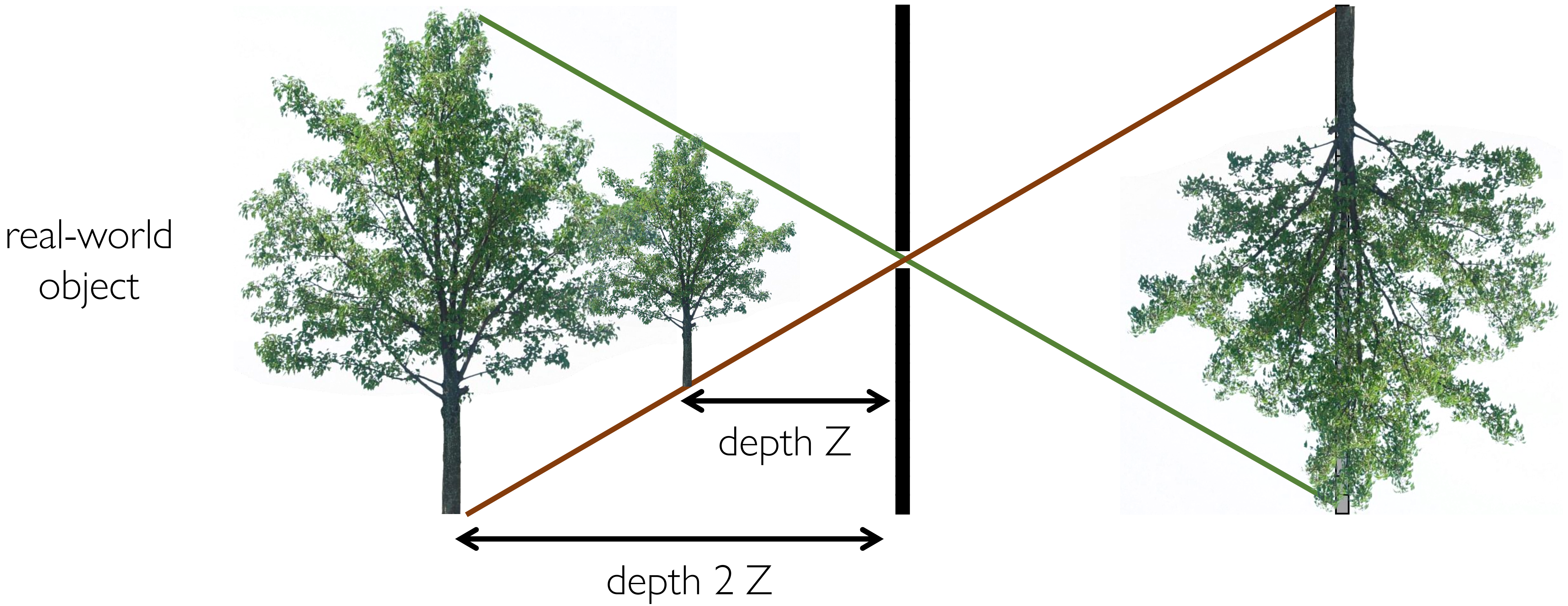
The Ames room illusion





Magnification depends on depth

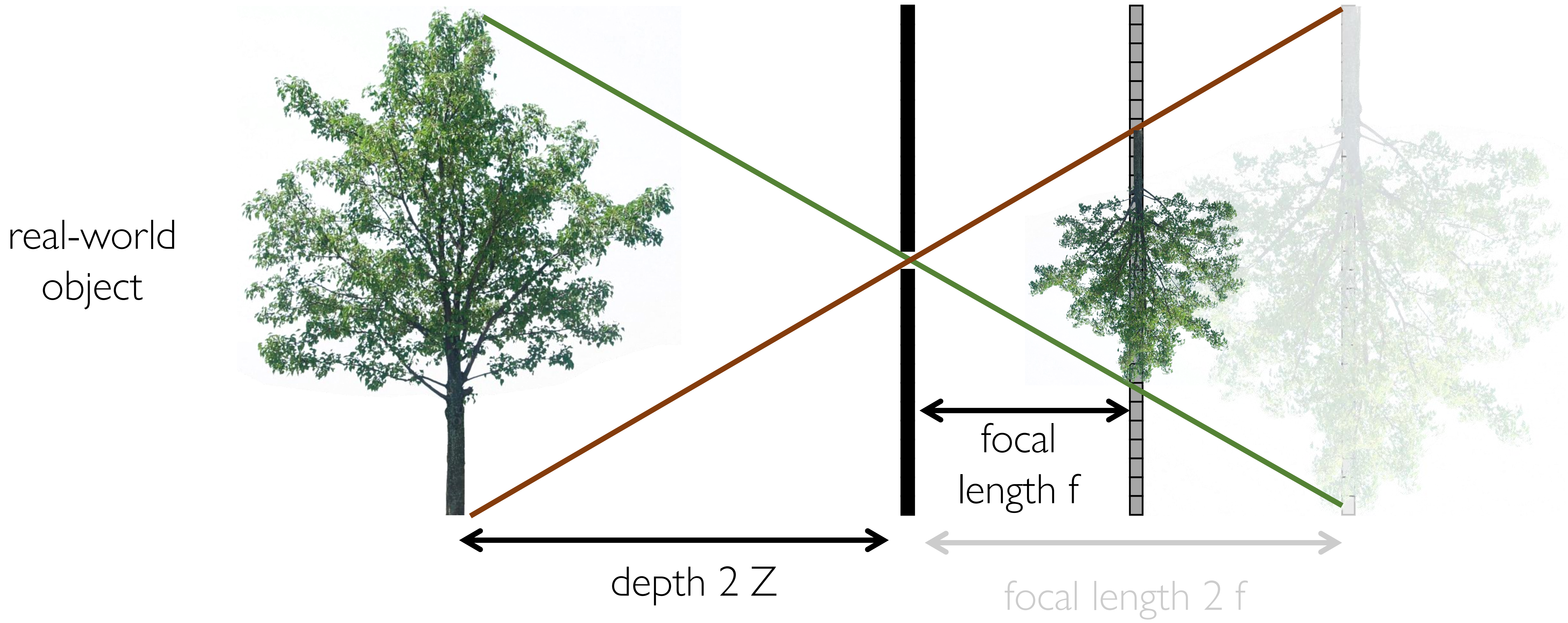
- What happens as we change the focal length?



Magnification depends on focal length

What if…

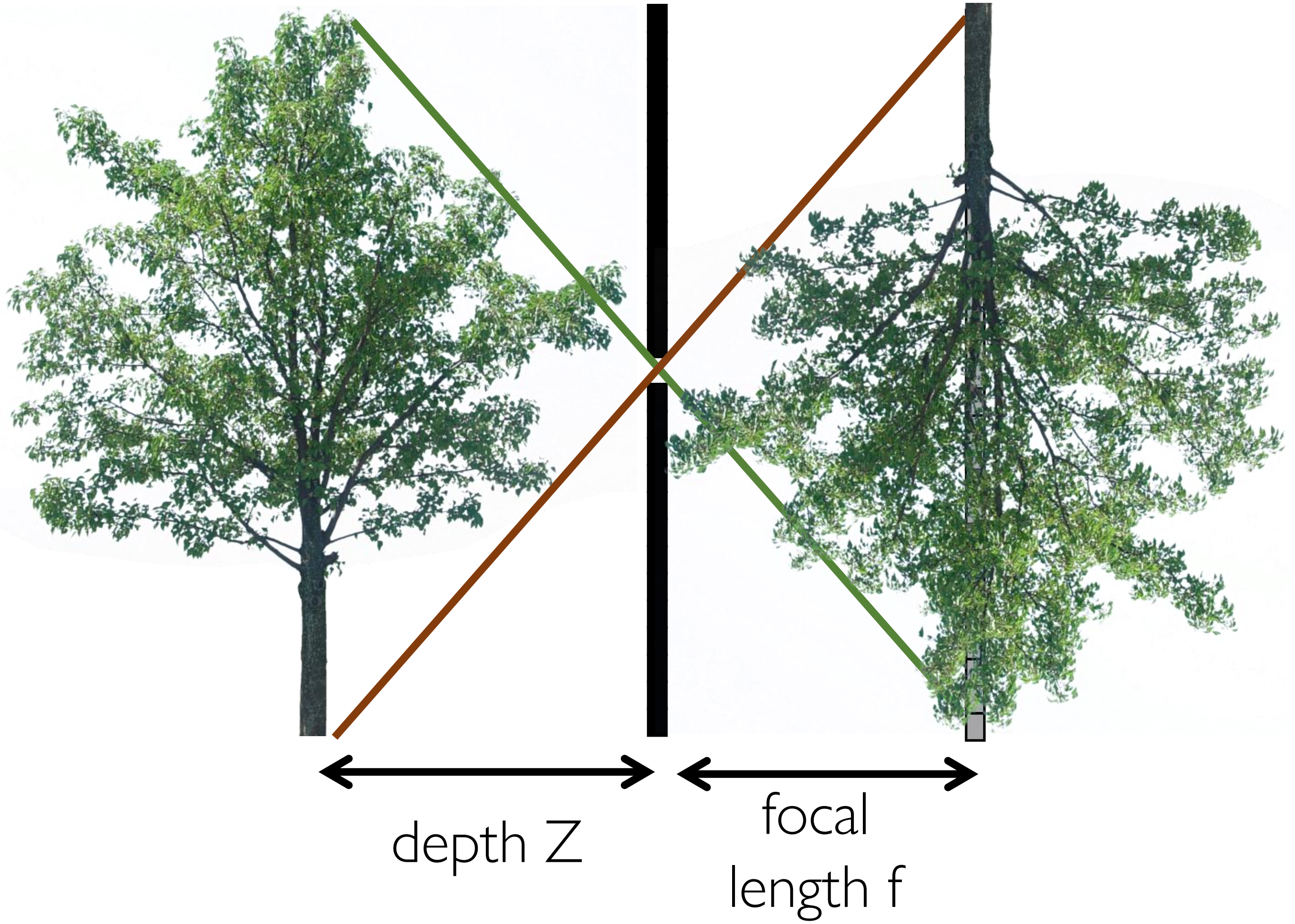
1. Set focal length to half.



What if…

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

real-world object



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

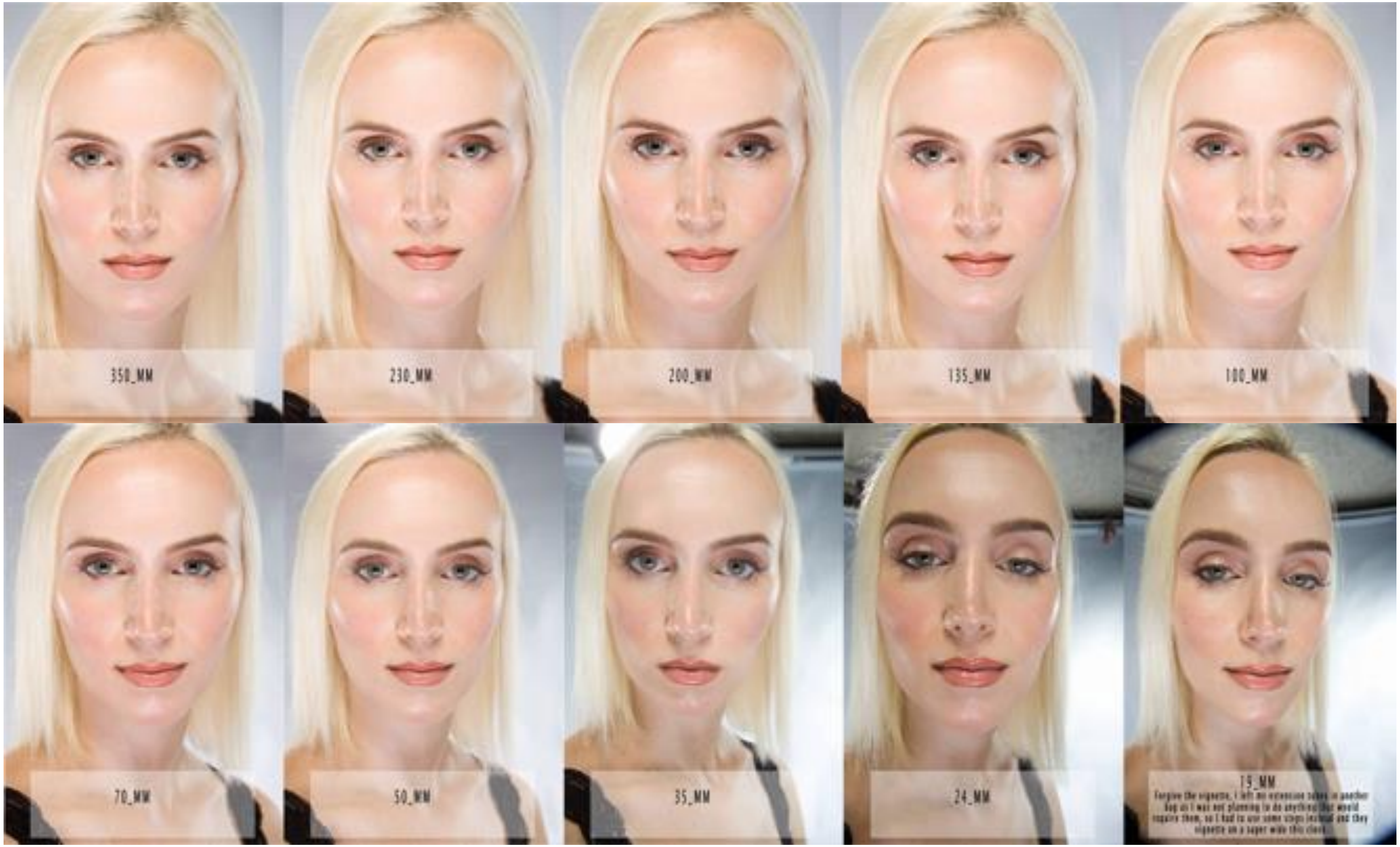
Perspective distortion

long focal length

mid focal length

short focal length

Perspective distortion



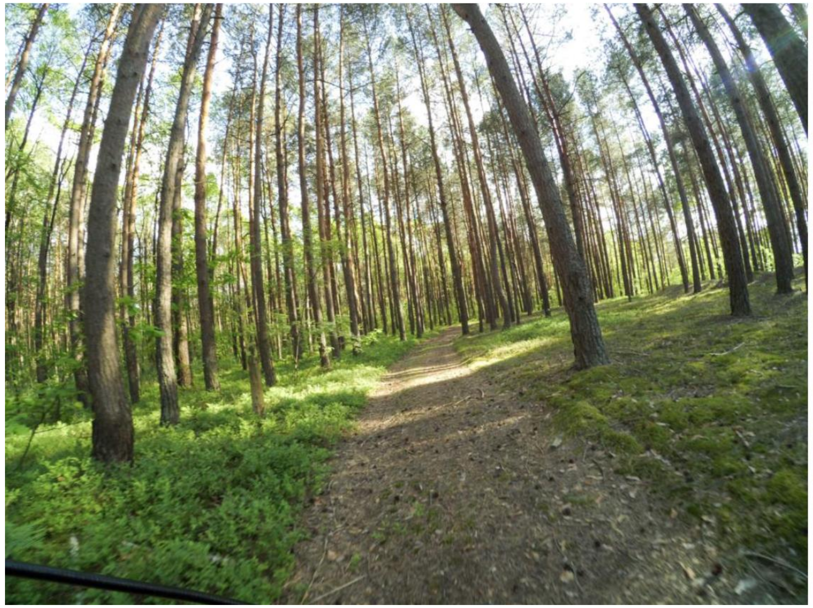
Vertigo effect

- How would you create this effect?

More intrinsic

More Intrinsic parameter

- Radial distortion
  - We assume that the basic camera model follows pinhole model.
  - Unfortunately, cameras do use lens



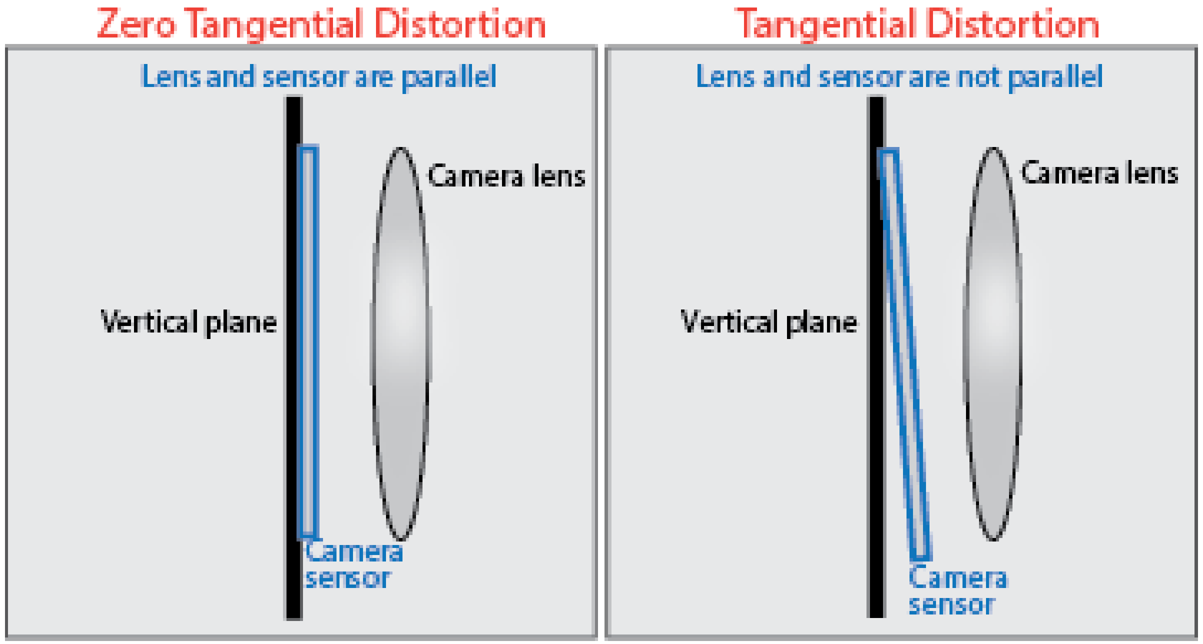
distorted image

Undistorted image

https://research.mapillary.com/publication/cvpr19d/

More Intrinsic parameter

- Tangential distortion
  - Tangential distortion occurs when the lens and image plane are not parallel.



Calibration

- Camera calibration is to estimate both
  - Camera intrinsic parameter
  - Extrinsic parameter (pose estimation)

- Why the calibration is necessary?
  - If we know the origin direction of ray,
  - We can back projection → 3D reconstruction

Type of camera calibration

Camera-base sensor systems

Single camera system

Multi-camera system (stereo system)

Camera-LiDAR system

Calibration of each system

Single camera system

- Calibration goal:
  - intrinsic parameters

$$\mathbf{K}$$

$$x_u = x + \underbrace{(x - x_c) \cdot (k_1 \cdot r^2 + k_2 \cdot r^4 + \cdots)}_{\text{radial terms}}$$

$$y_u = y + \underbrace{(y - y_c) \cdot (k_1 \cdot r^2 + k_2 \cdot r^4 + \cdots)}_{\text{radial terms}}$$

Multi-camera system (stereo system)

- Calibration goal:
  - Intrinsic parameters (for each)
  - Extrinsic parameters btw them

$$\mathbf{K}$$

$$x_u = x + \underbrace{(x - x_c) \cdot (k_1 \cdot r^2 + k_2 \cdot r^4 + \cdots)}_{\text{radial terms}}$$

$$y_u = y + \underbrace{(y - y_c) \cdot (k_1 \cdot r^2 + k_2 \cdot r^4 + \cdots)}_{\text{radial terms}}$$

$$[\mathbf{R}|\mathbf{t}]$$

btw cameras

Camera-LiDAR system

- Calibration goal:
  - Intrinsic parameters
  - Extrinsic parameters btw them

$$\mathbf{K}$$

$$x_u = x + \underbrace{(x - x_c) \cdot (k_1 \cdot r^2 + k_2 \cdot r^4 + \cdots)}_{\text{radial terms}}$$

$$y_u = y + \underbrace{(y - y_c) \cdot (k_1 \cdot r^2 + k_2 \cdot r^4 + \cdots)}_{\text{radial terms}}$$

$$[\mathbf{R}|\mathbf{t}]$$

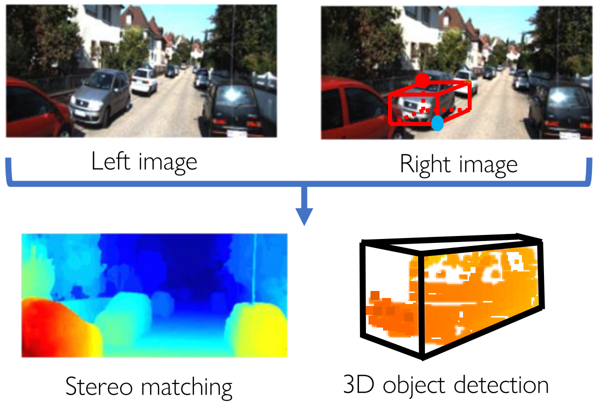
btw camera and LiDAR

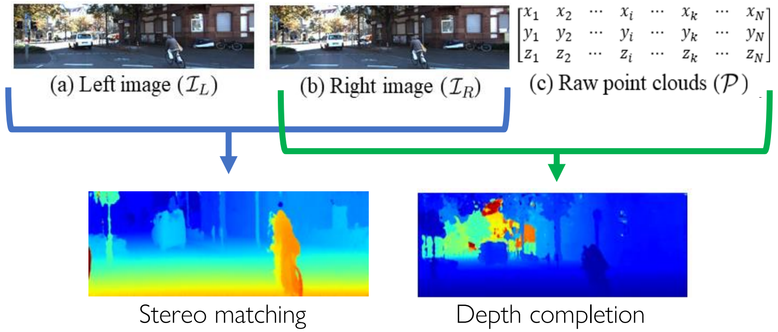
Application of each system

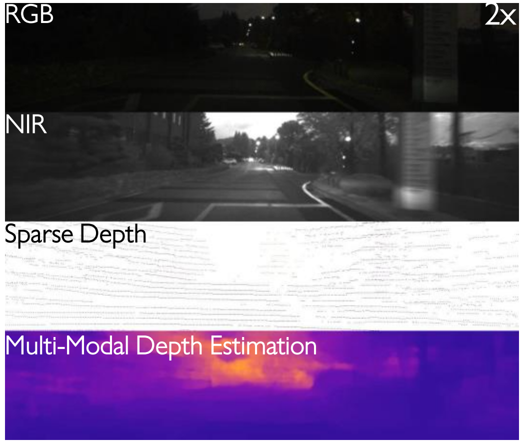
RGB Sensor System

RGB-LiDAR Sensor System

RGB-NIR-LiDAR Sensor System







- Stereo object matching network (ICRA'21)

- Stereo-LiDAR fusion for depth estimation (RA-Letter + ICRA'21)

- Adaptive Cost Volume Fusion Network for Multi-Modal Depth Estimation in Changing Environments (RA-Letter + ICRA'22)

[1] Choe et al. "Stereo Object Matching Network." ICRA 2021

[1] Choe et al. "Volumetric Propagation Network: Stereo-LiDAR Fusion for High-Quality Depth Estimation." RA-L (with ICRA presentation) 2021

[1] Park et al. "Adaptive Cost Volume Fusion Network for Multi-Modal Depth Estimation in Changing Environment." RA-L (with ICRA presentation) 2021

Geometric camera calibration

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

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

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

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

- Heterogeneous coordinates

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

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

(non-linear relation between coordinates)

- How can we make these relations linear?

Mapping between 3D point and image points

- 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

Mapping between 3D point and image points

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

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

- How do we proceed?

Mapping between 3D point and image points

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

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

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?

Mapping between 3D point and image points

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

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

in matrix form …

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

for N points …

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

- How do we solve this system?

Solve for camera matrix by

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

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

SVD!

Solve for camera matrix by

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

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

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

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

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

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

Solve for camera matrix by

- 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?

Solve for camera matrix by

- 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

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

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

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**

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

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 **c**

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

SVD of P!
c is the Eigenvector corresponding to smallest Eigenvalue

Find intrinsic K and rotation R

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

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

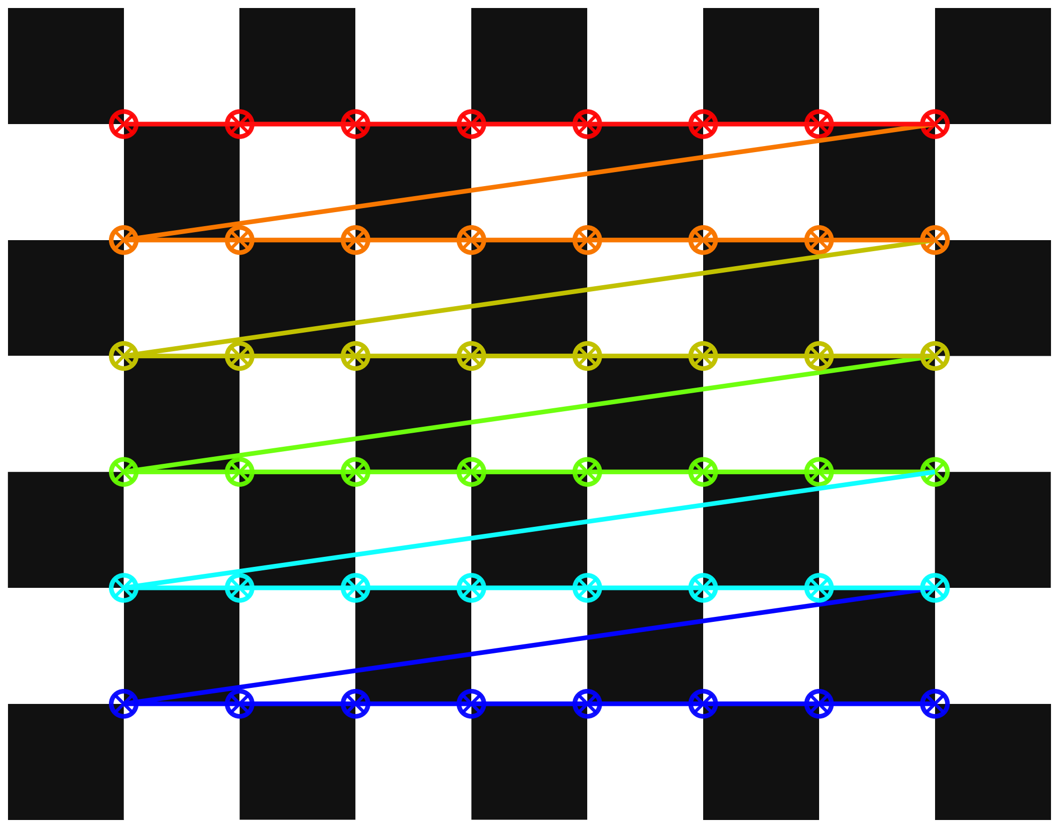
- Find the (pose) estimate of

$$\mathbf{P}$$

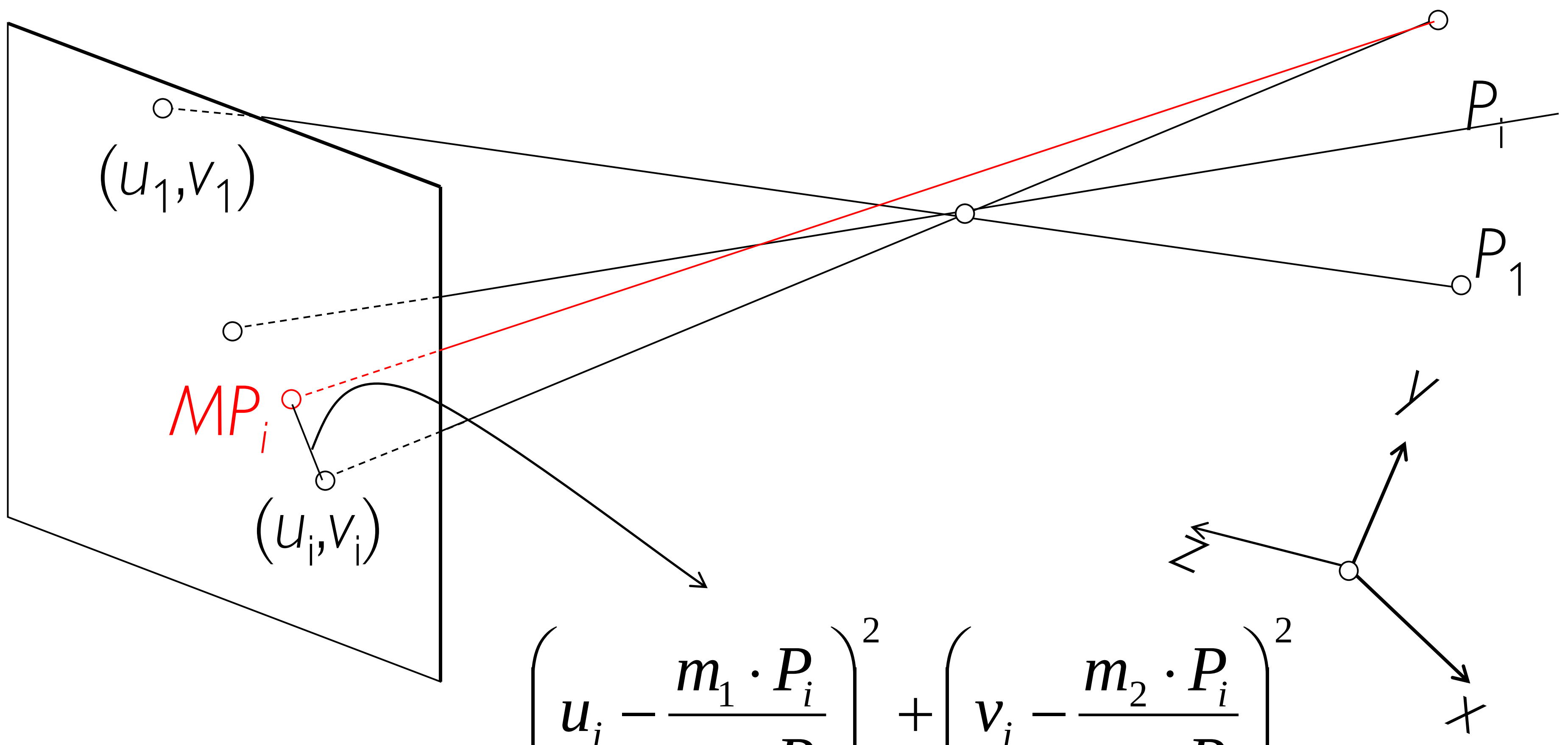
we'll use a **perspective** camera model for pose estimation

Checkerboard-based calibration

- Checkerboard pattern on a plane
- **Known 3D points:**
  - M x N corners (e.g., 8 x 6 corners)
  - Step: fixed step size (e.g., 10mm)
- **2D points (on image domain)**
  - Easy to detect

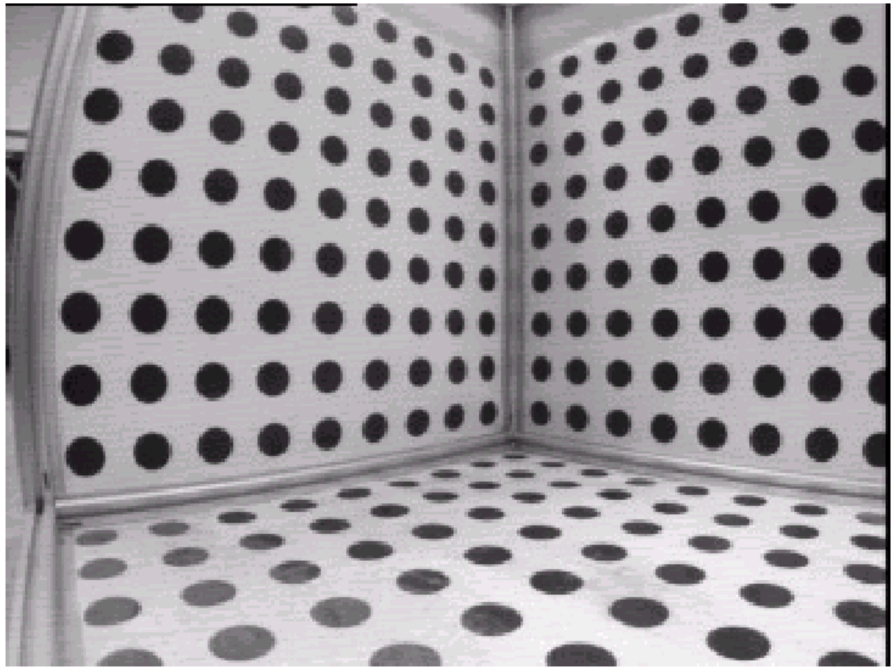


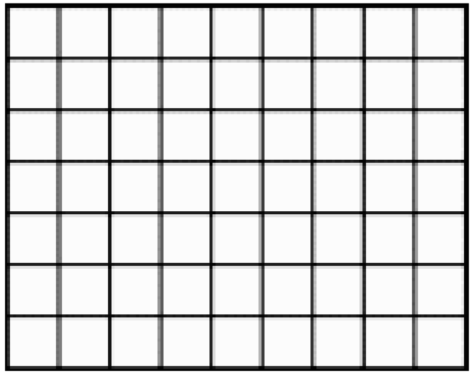
Minimizing reprojection error



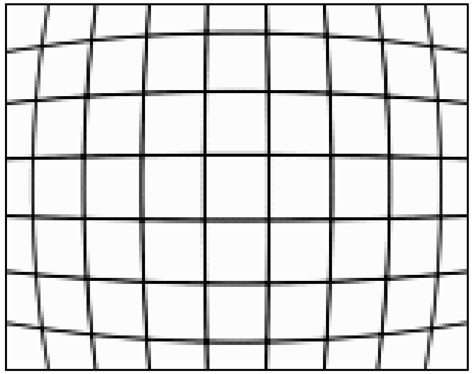
$$\left(u_i - \frac{m_1 \cdot P_i}{m_3 \cdot P_i}\right)^2 + \left(v_i - \frac{m_2 \cdot P_i}{m_3 \cdot P_i}\right)^2$$

Radial distortion

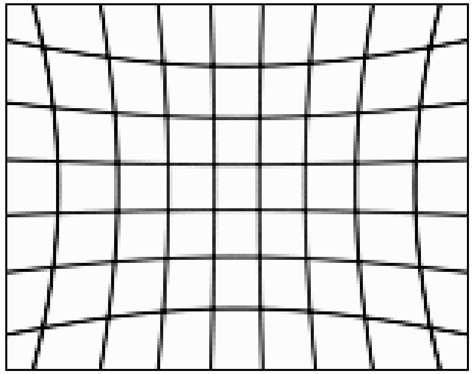




no distortion



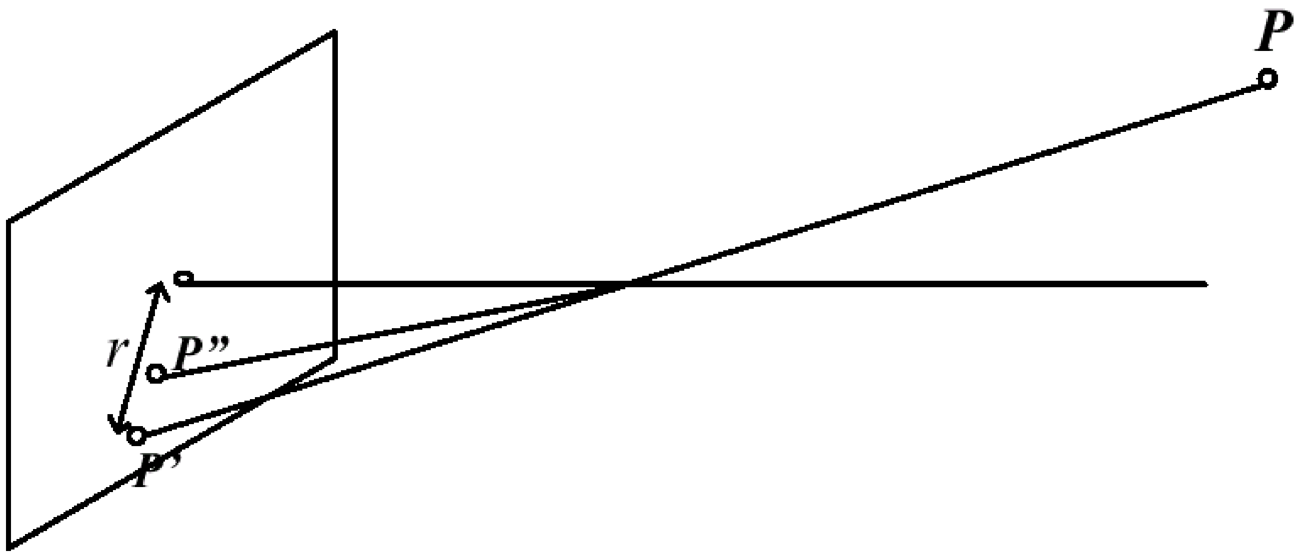
barrel distortion

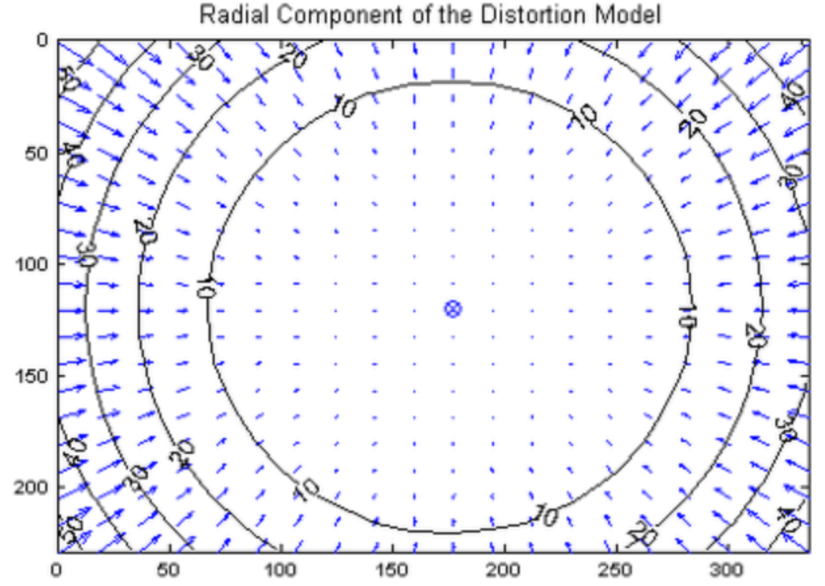


pincushion distortion

- What causes this distortion?

Radial distortion model





Pixel error	= [0.2688, 0.277]	
Focal Length	= (181.995, 164.699)	+/- [0.4468, 0.4092]
Principal Point	= (175.5, 119.5)	+/- [0, 0]
Skew	= 0	+/- 0
Radial coefficients	= (-0.289, 0.08213, -0.01014)	+/- [0.002255, 0.001728, 0.0003671]
Tangential coefficients	= (-0.0002611, -0.0002235)	+/- [0.0002153, 0.0001831]

Ideal:

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

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

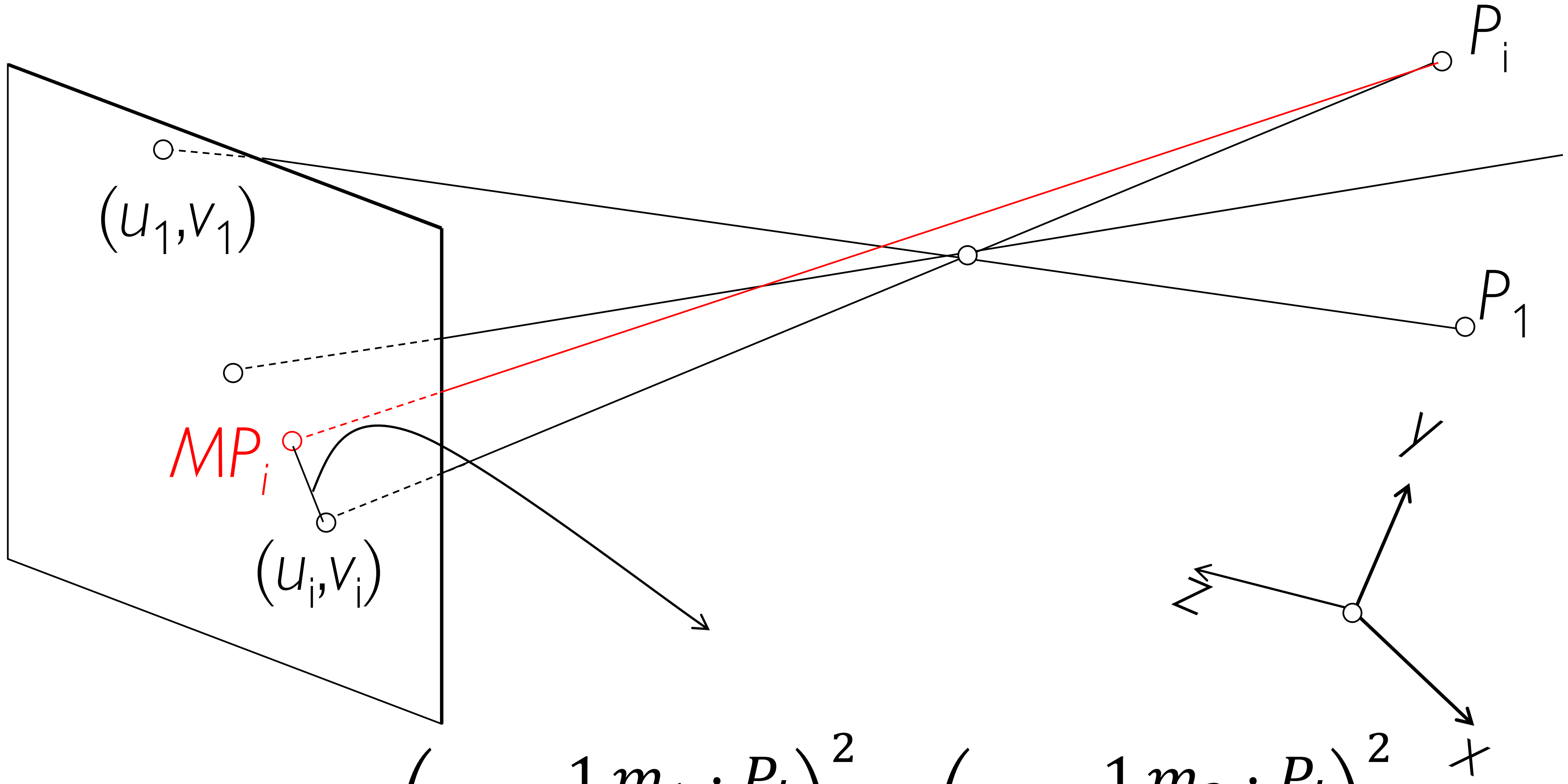
Distorted:

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

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

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

Minimizing reprojection error with radial distortion



Add distortions to reprojection error:

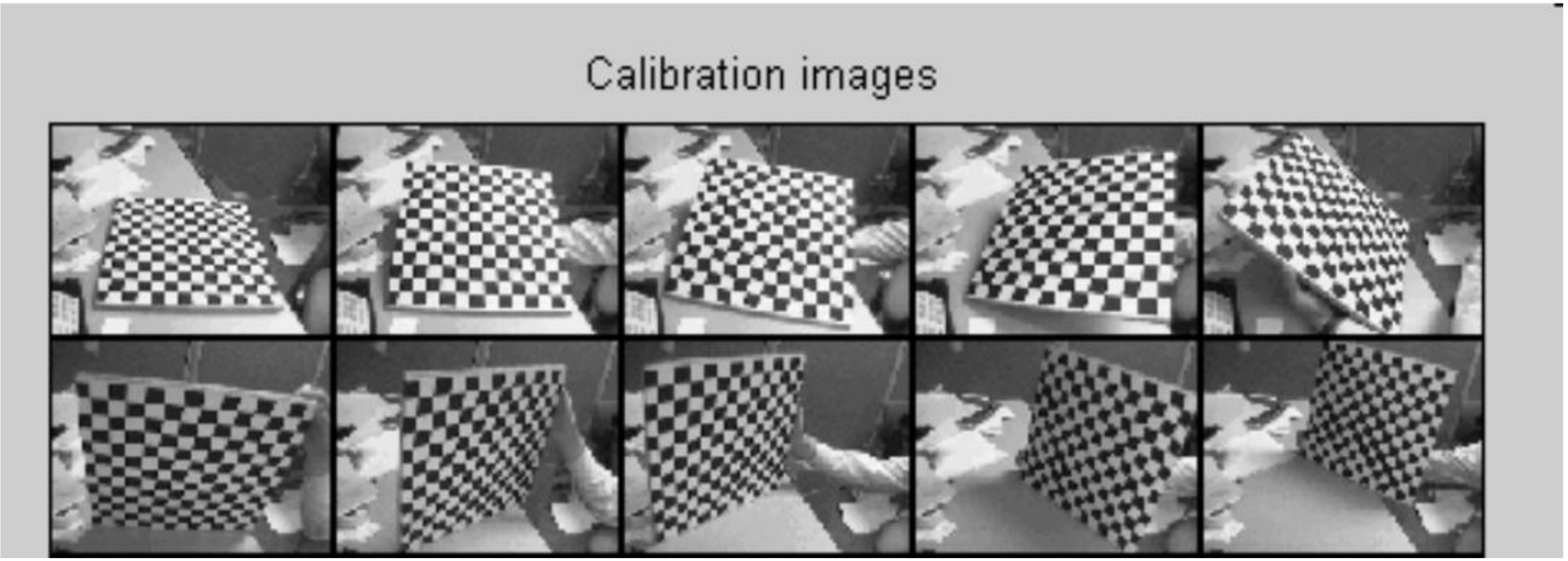
$$\left(u_i - \frac{1}{\lambda}\frac{m_1 \cdot P_i}{m_3 \cdot P_i}\right)^2 + \left(v_i - \frac{1}{\lambda}\frac{m_2 \cdot P_i}{m_3 \cdot P_i}\right)^2$$

Correcting radial distortion

before

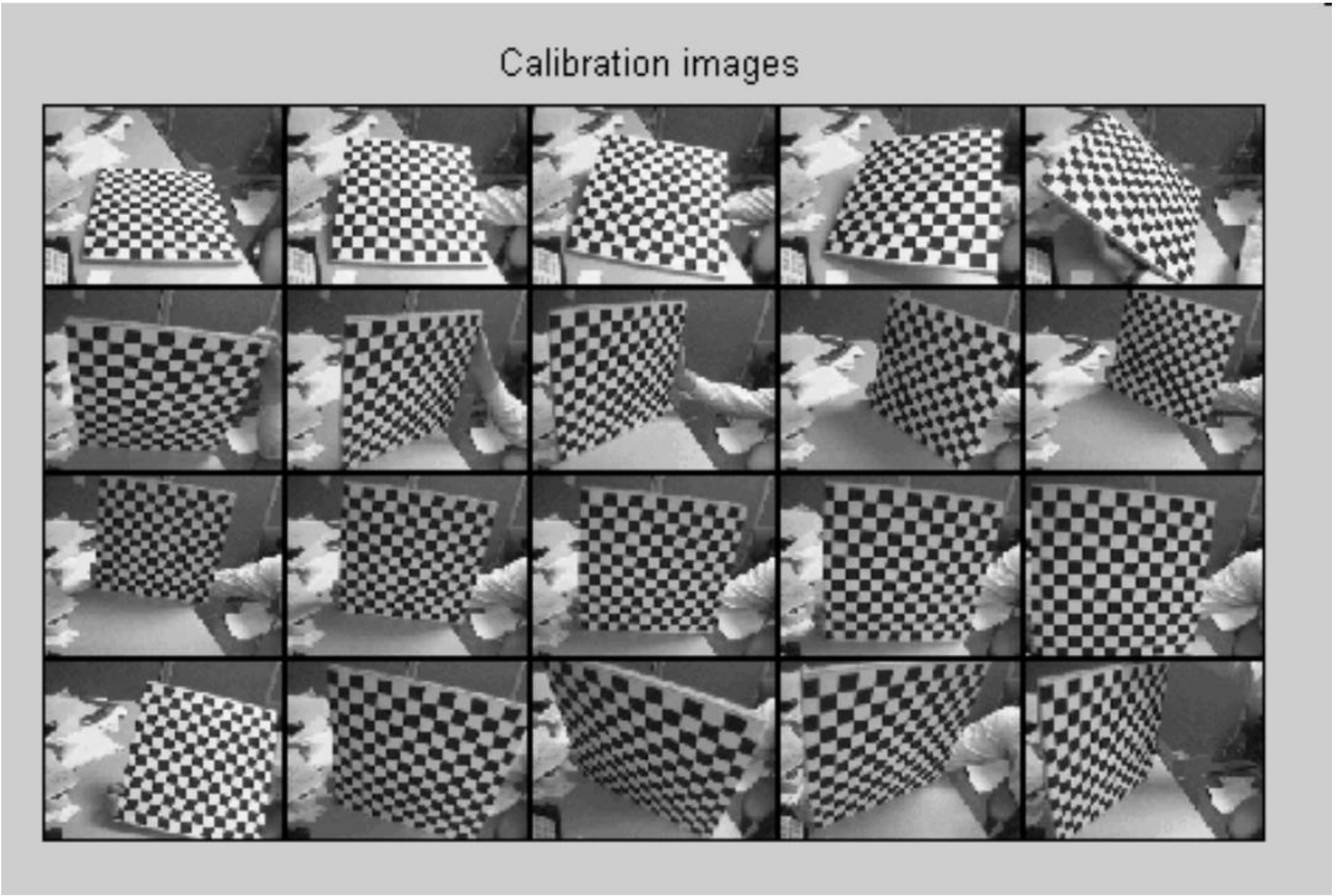
after

Checkerboard-based calibration

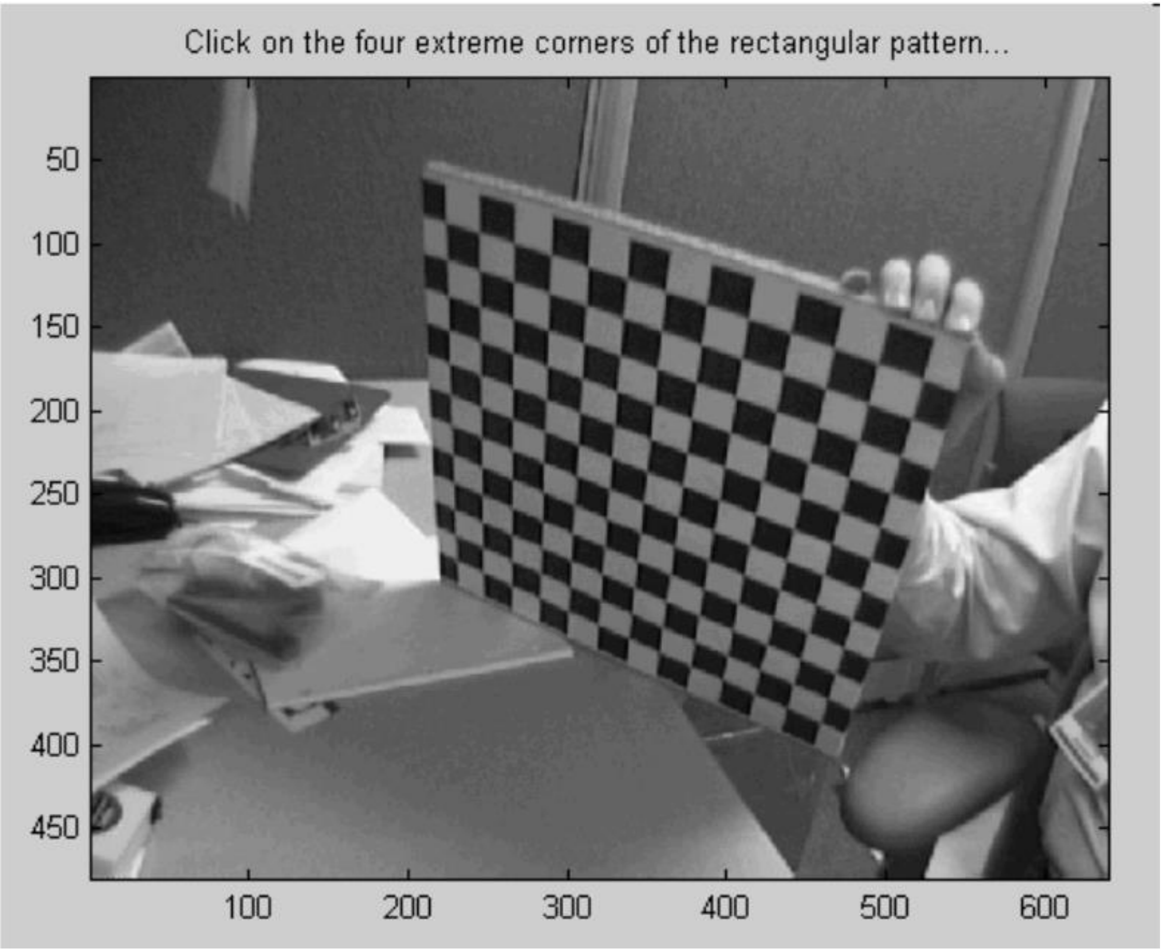


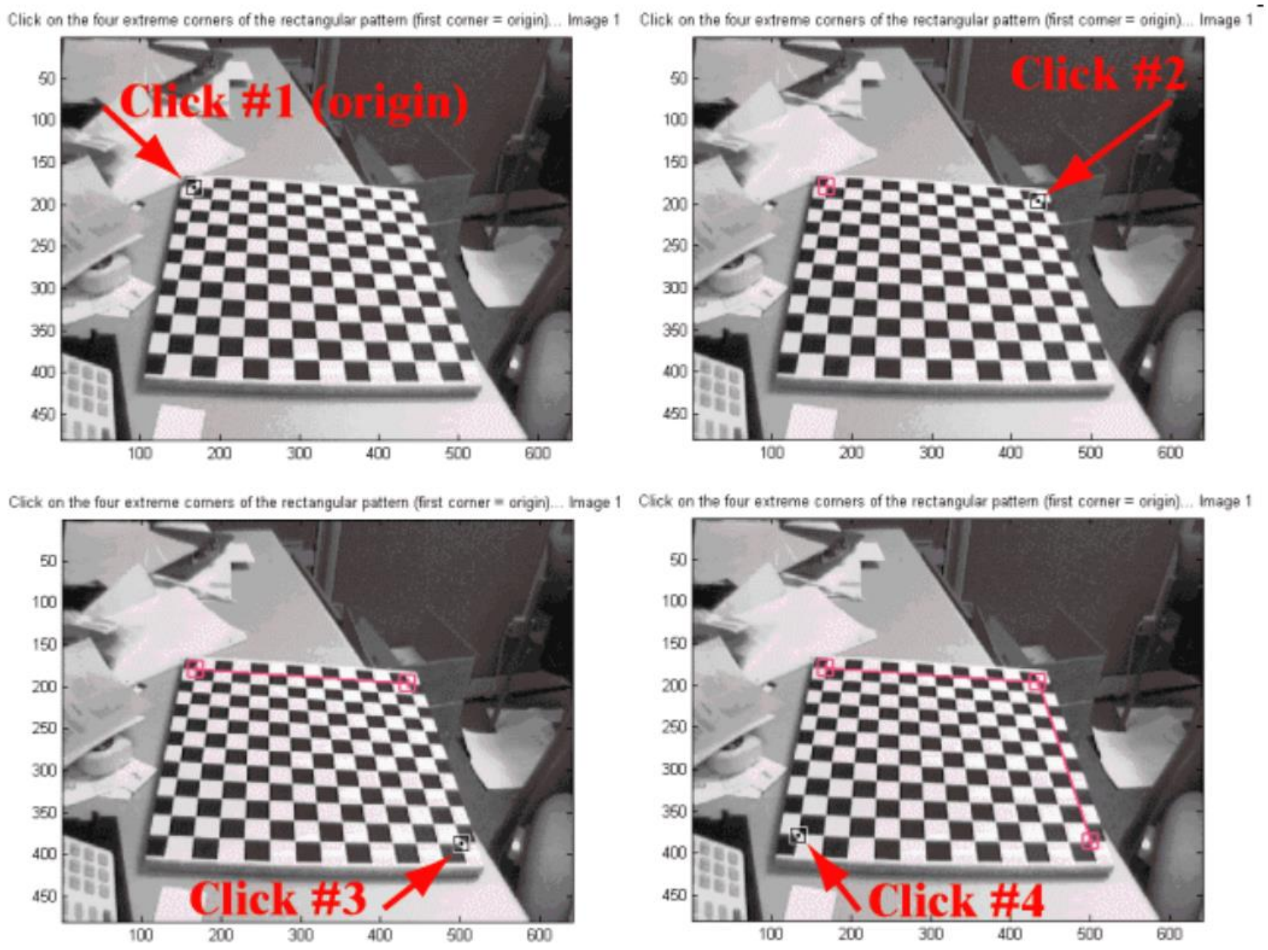
- Advantages:
  - Only requires a plane.
  - Don't have to know positions/orientations.
  - Great code available online!
    - Matlab version: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html
    - Also available on OpenCV.
- Disadvantages: need to solve non-linear optimization problem.

Step-by-step demonstration

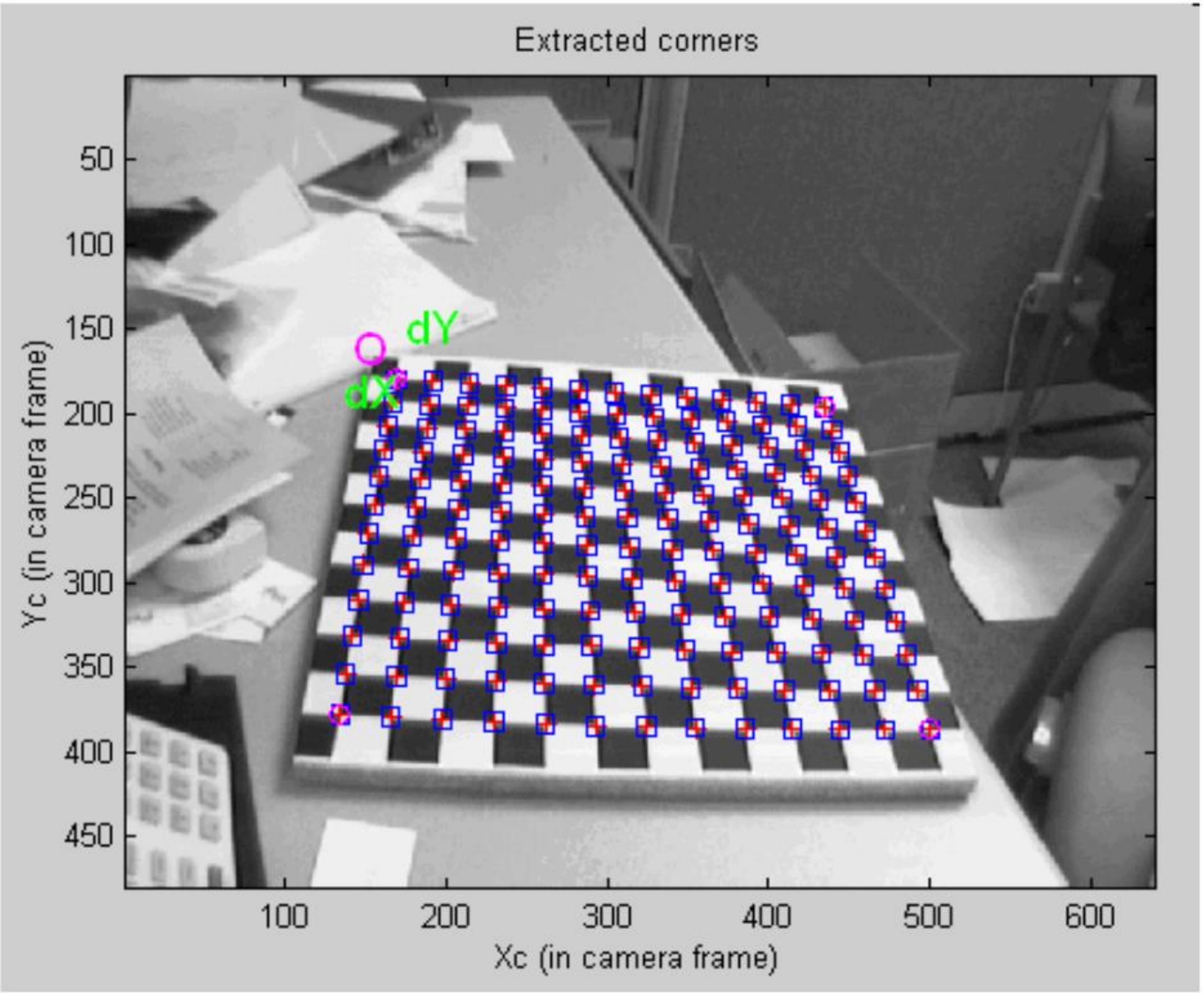


Step-by-step demonstration

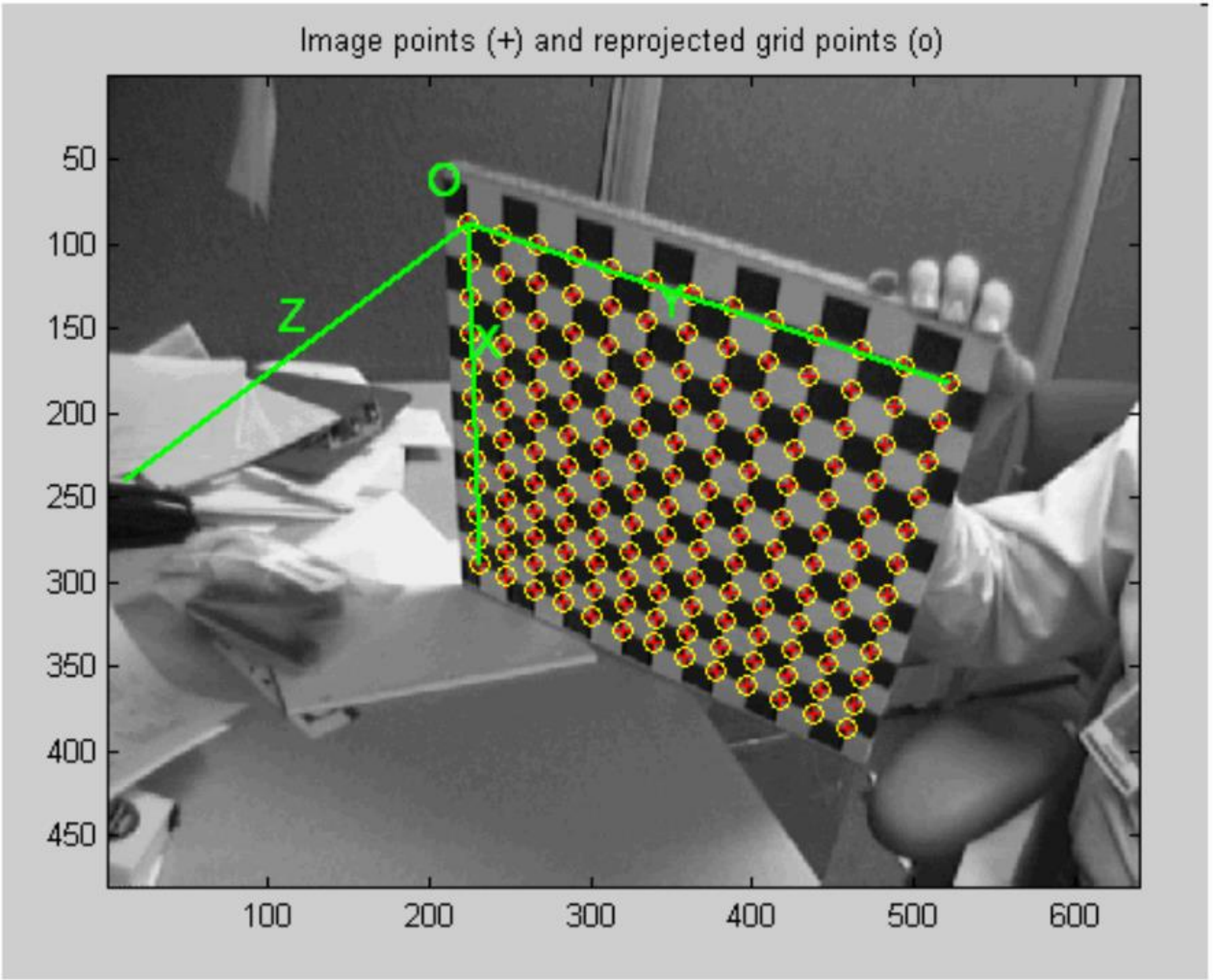




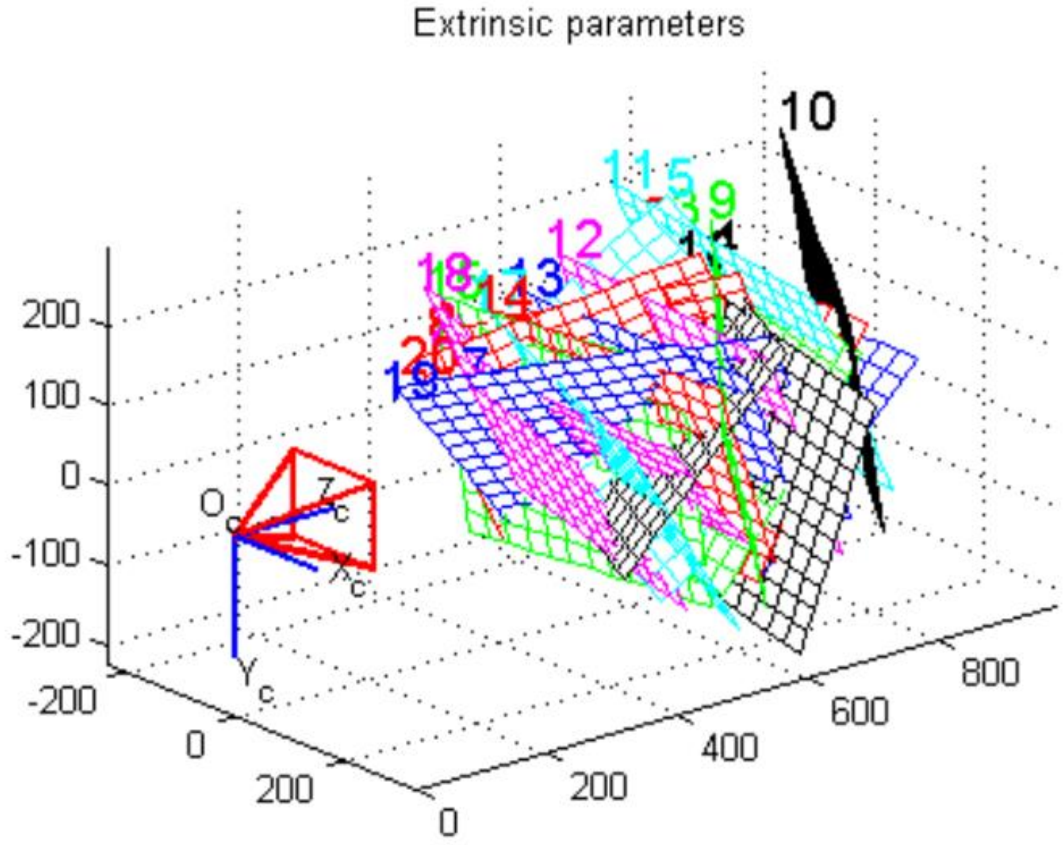
Step-by-step demonstration

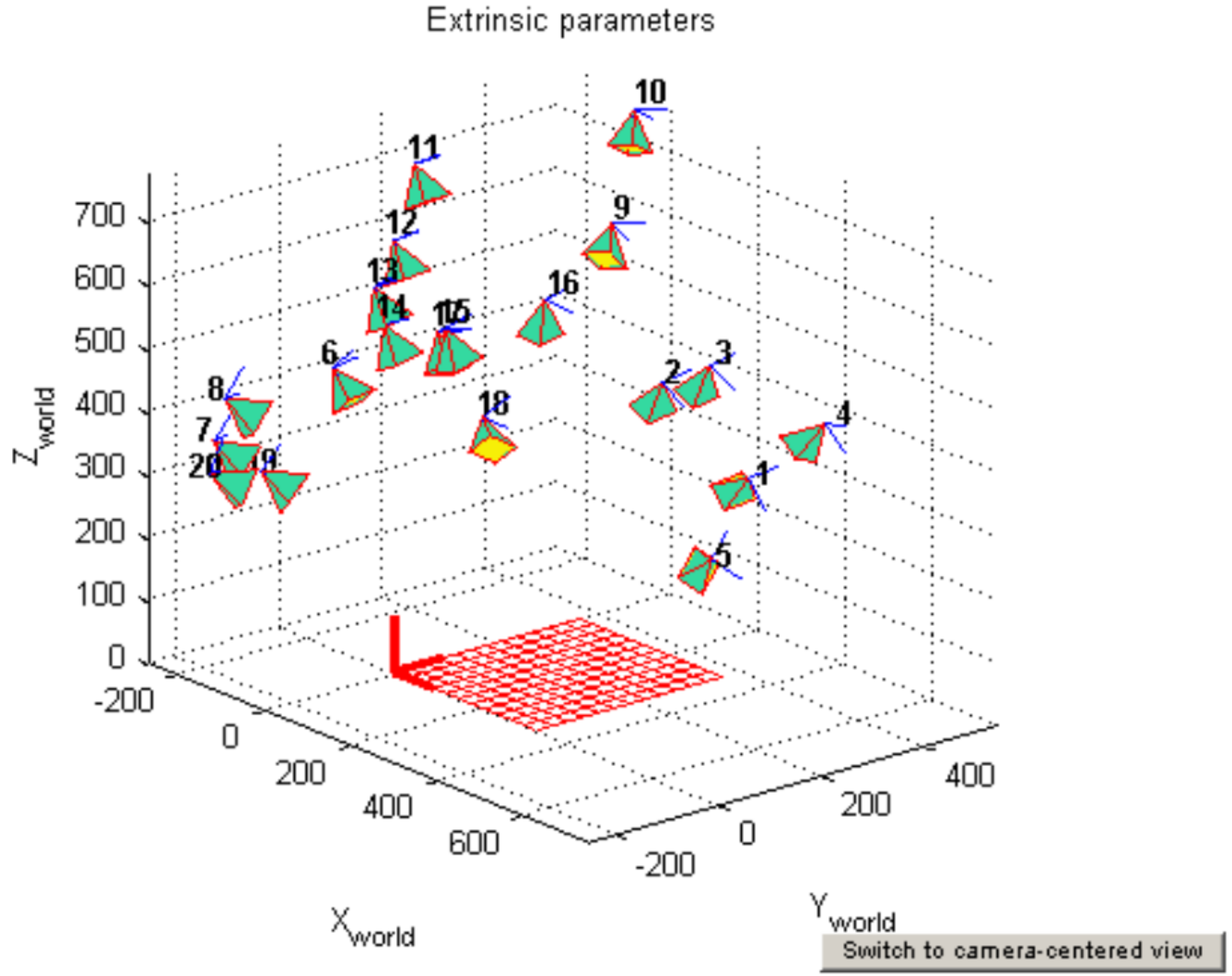


Step-by-step demonstration



Step-by-step demonstration






Tower Health

Back-Projection by $\mathbf{K}^{-1}$

Back-projection

[Projection]

$$\mathbf{x} = \mathbf{PX}$$
$$= \mathbf{K}[\mathbf{I}|\mathbf{0}]\mathbf{X}$$

- Back-projection by $\mathbf{K}^{-1}$

$$\mathbf{K}^{-1}\mathbf{x} = \mathbf{K}^{-1}\mathbf{PX}$$
$$= \mathbf{K}^{-1}\mathbf{K}[\mathbf{I}|\mathbf{0}]\mathbf{X}$$

$$\Downarrow$$

$$\mathbf{K}^{-1}\mathbf{x} \cong \mathbf{X}$$

Inverse of intrinsic parameters $\mathbf{K}^{-1}$

- $\mathbf{K}$ vs. $\mathbf{K}^{-1}$

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

vs.

$$\mathbf{K}^{-1} = \begin{bmatrix} 1/f_x & 0 & -c_x/f_x \\ 0 & 1/f_y & -c_y/f_y \\ 0 & 0 & 1 \end{bmatrix}$$

Back-projection

- Convert pixel to mm

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \hat{\mathbf{X}} = \mathbf{K}^{-1}\mathbf{x} = \begin{bmatrix} 1/f_x & 0 & -c_x/f_x \\ 0 & 1/f_y & -c_y/f_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

- [Example] X value only

$$\mathrm{X} = \frac{u - c_x}{f_x}$$

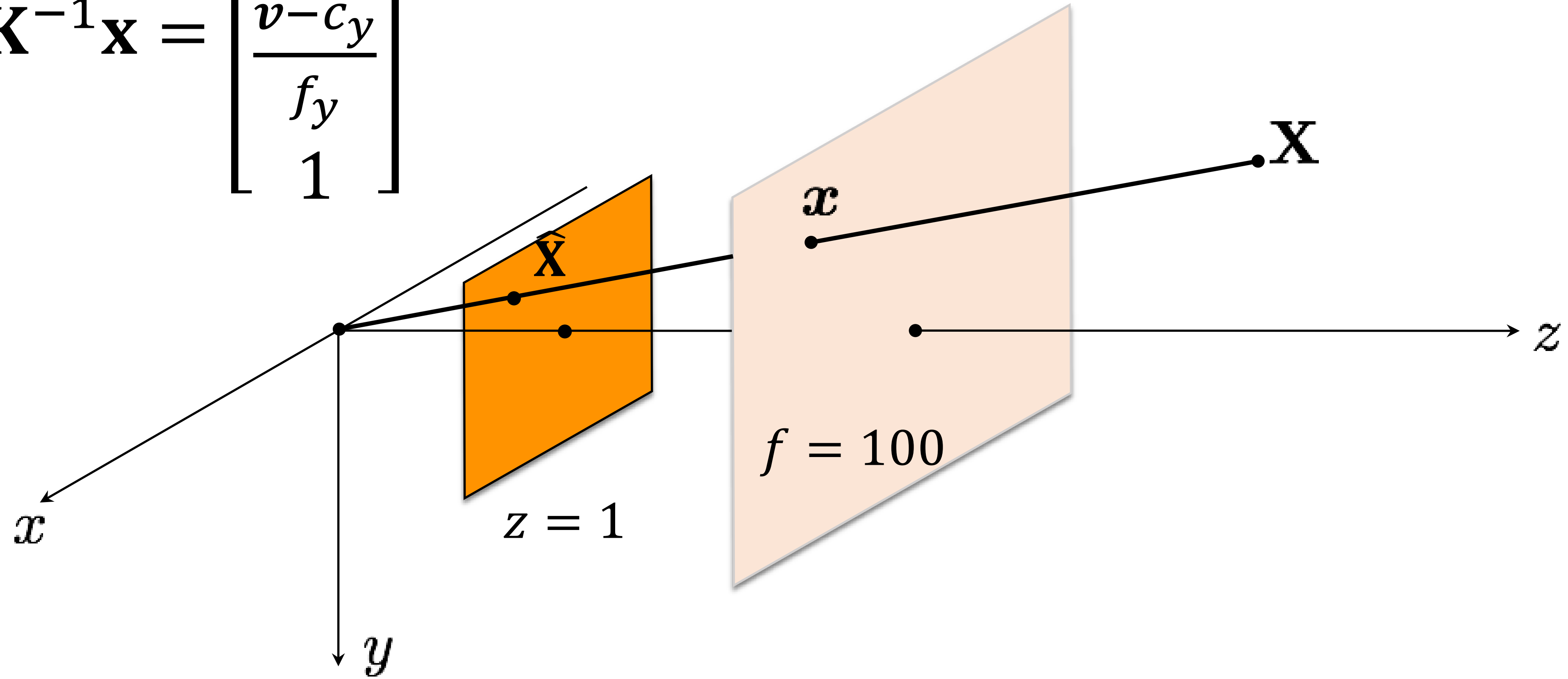
[Transformation]

?

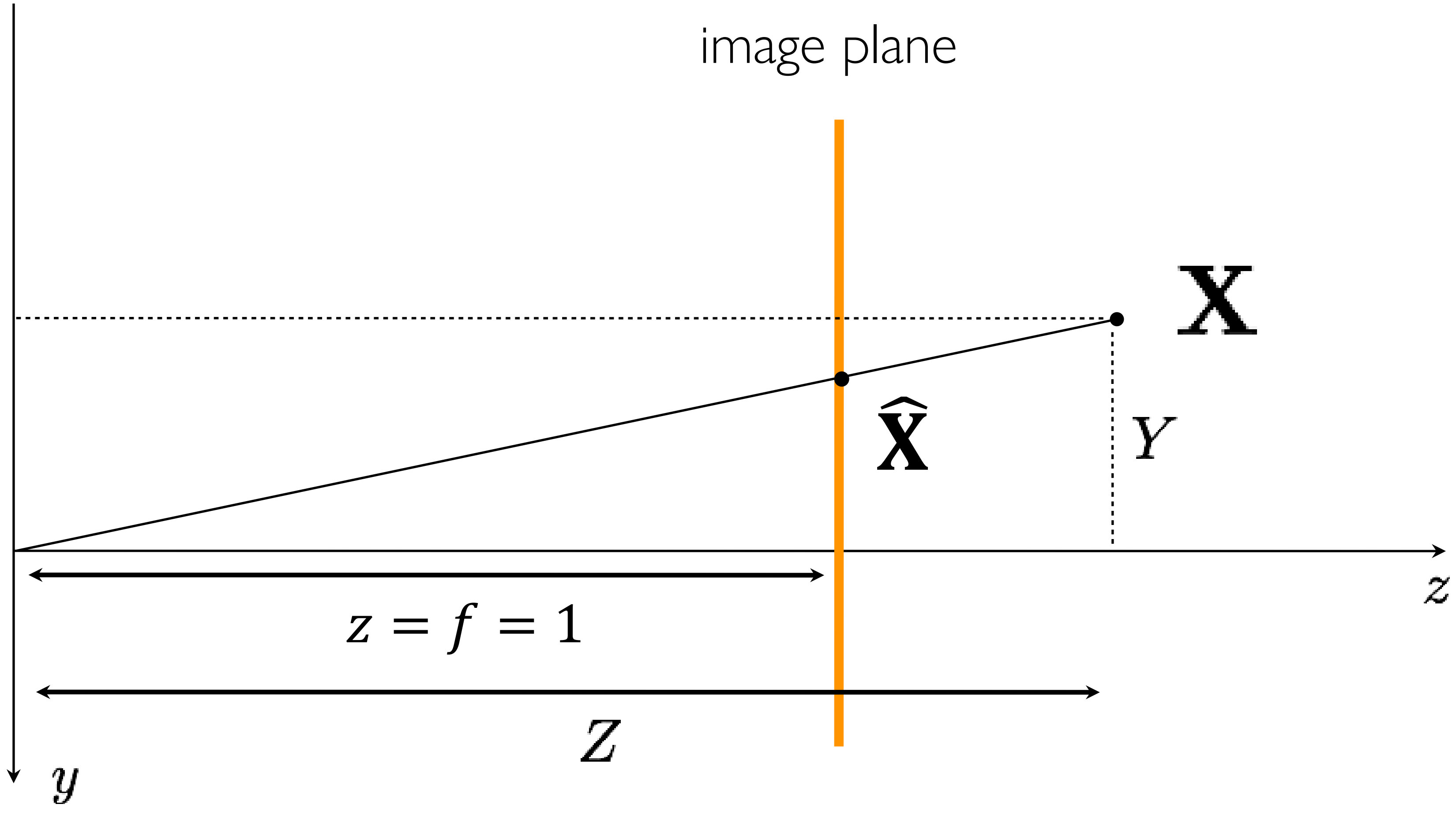
Visualization of back-projection

- Normalized coordinate

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \hat{\mathbf{X}} = \mathbf{K}^{-1}\mathbf{x} = \begin{bmatrix} \frac{u-c_x}{f_x} \\ \frac{v-c_y}{f_y} \\ 1 \end{bmatrix}$$



Visualization of back-projection on 2D (y-z) space



Connection to depth

Property of normalized coordinate

- Last element equals to 1 (NOT unit norm)

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \hat{\mathbf{X}} = \mathbf{K}^{-1}\mathbf{x} = \begin{bmatrix} \frac{u - c_x}{f_x} \\ \frac{v - c_y}{f_y} \\ 1 \end{bmatrix}$$

- Relationship to depth map
  - Depth map encodes only Z value of 3D points on 2D image domain
  - Last elements (z-value) of normalized coordinate equals to 1

Depth to 3D point clouds