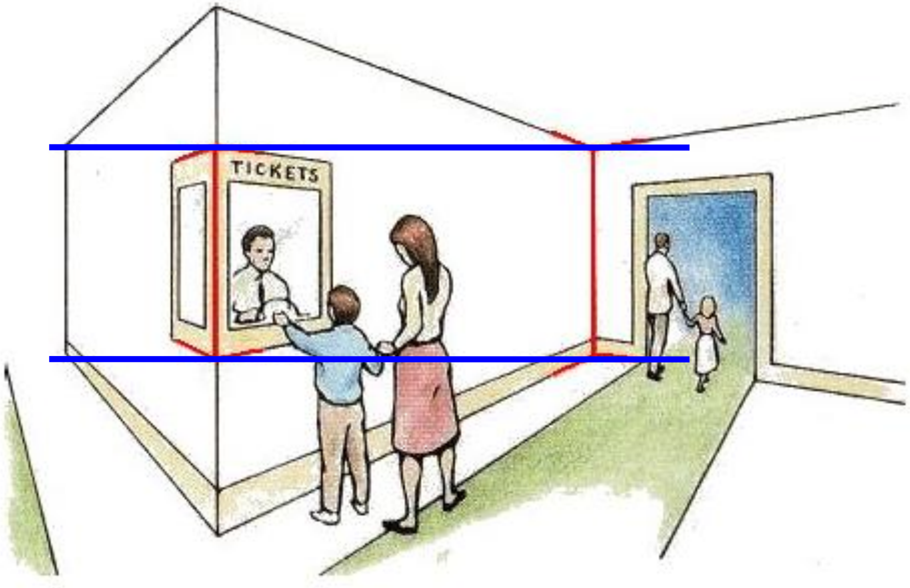
Cameras and Stereo

EE/CSE 576

Linda Shapiro

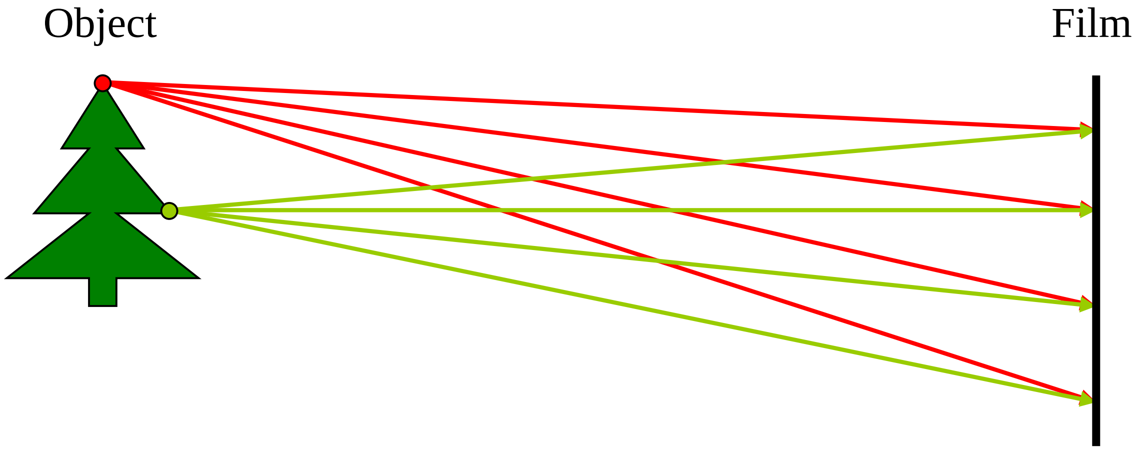
Müller-Lyer Illusion



http://www.michaelbach.de/ot/sze_muelue/index.html

- What do you know about perspective projection?
- Vertical lines?
- Other lines?

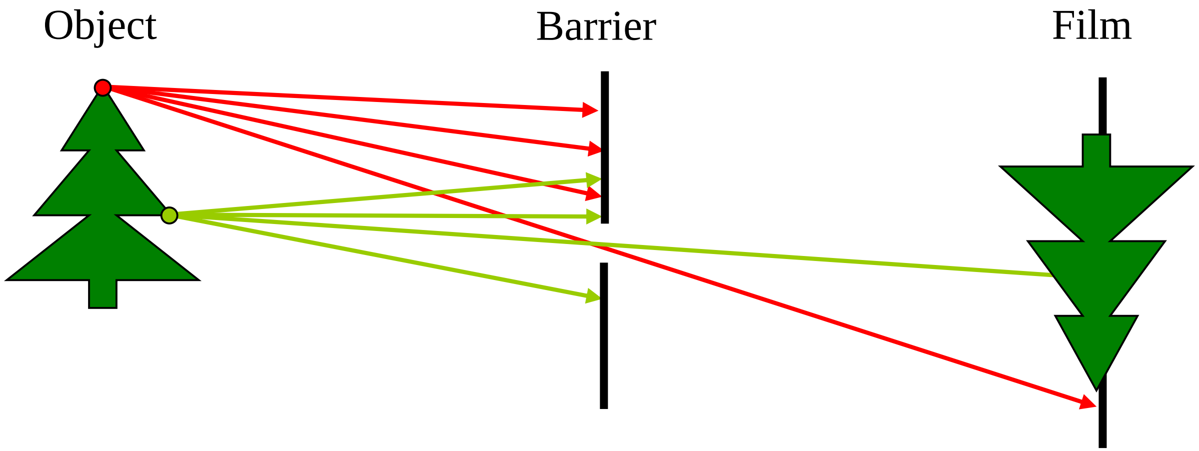
Image formation



Let’s design a camera

- Idea 1: put a piece of film in front of an object
- Do we get a reasonable image?

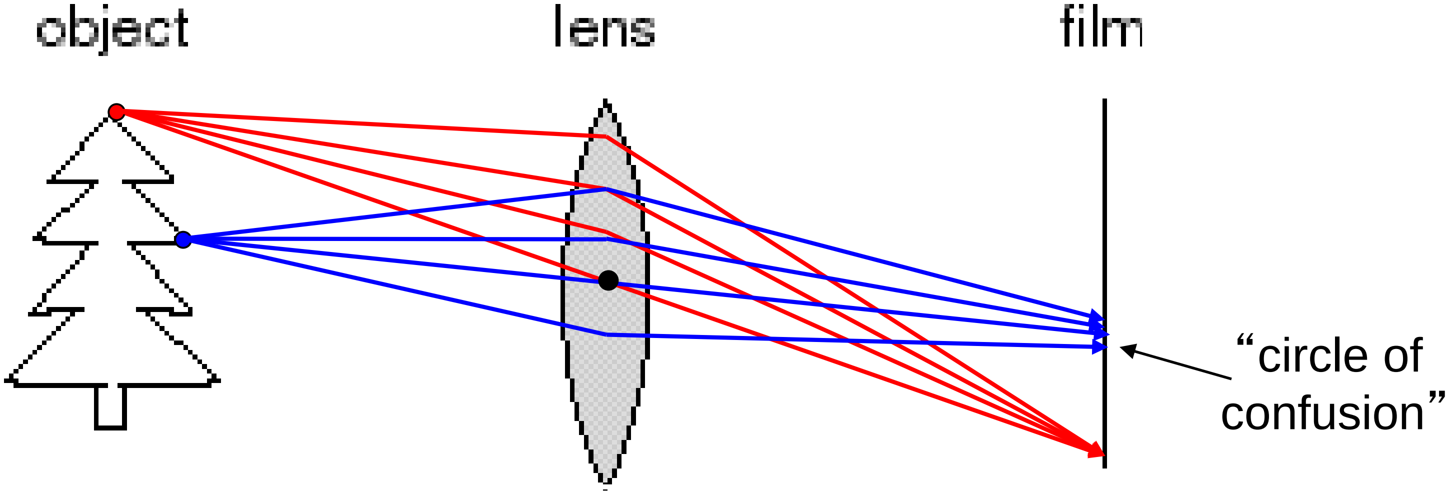
Pinhole camera



Add a barrier to block off most of the rays

- This reduces blurring
- The opening known as the **aperture**
- How does this transform the image?

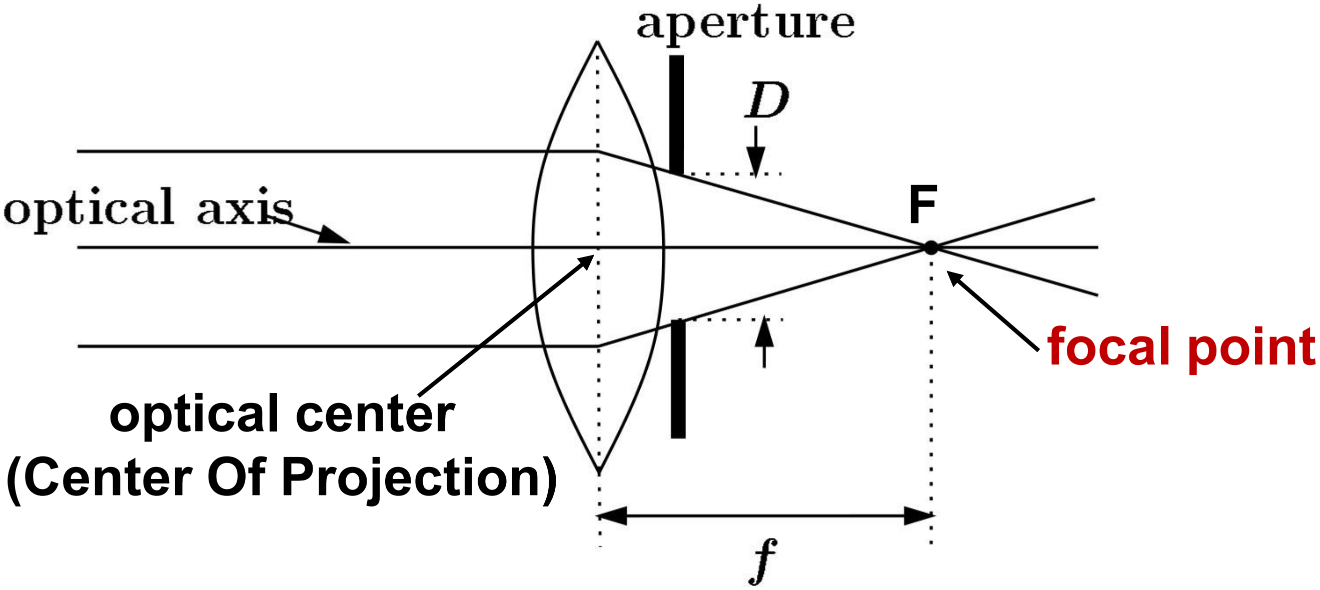
Adding a lens



A lens focuses light onto the film

- There is a specific distance at which objects are “in focus”
  - other points project to a “circle of confusion” in the image
- Changing the shape of the lens changes this distance

Lenses



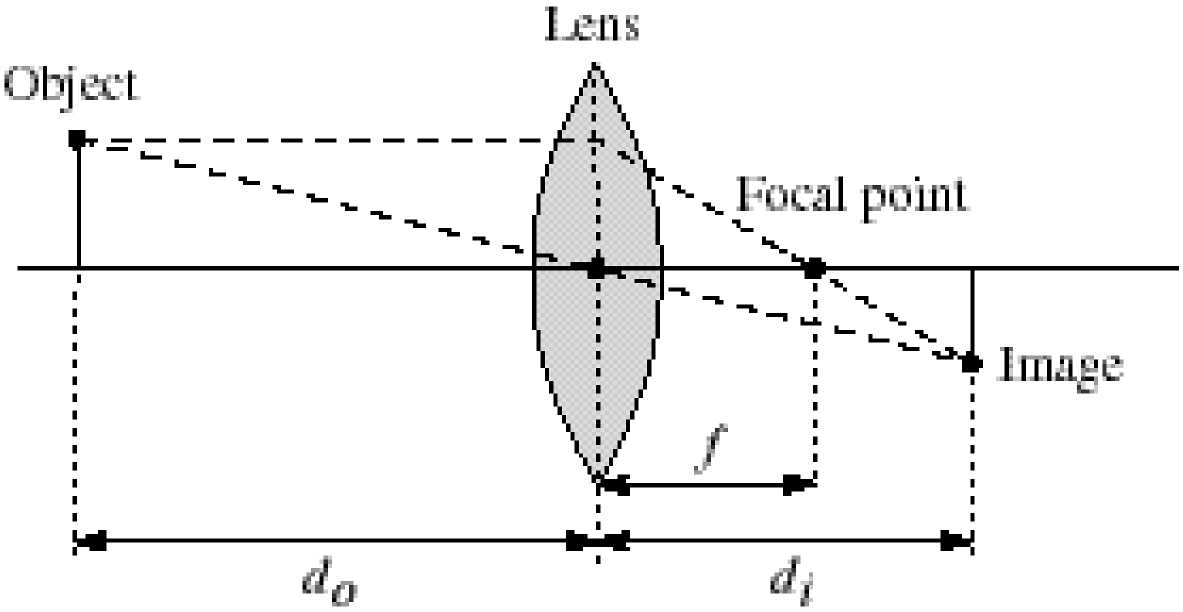
optical center
(Center Of Projection)

focal point

A lens focuses parallel rays onto a single focal point

- focal point at a distance *f* beyond the plane of the lens
  - *f* is a function of the shape and index of refraction of the lens
- Aperture of diameter D restricts the range of rays
  - aperture may be on either side of the lens
- Lenses are typically spherical (easier to produce)
- Real cameras use many lenses together (to correct for aberrations)

Thin lenses



Thin lens equation: $$\frac{1}{d_o} + \frac{1}{d_i} = \frac{1}{f}$$

- Any object point satisfying this equation is in focus

Digital camera

A digital camera replaces film with a sensor array

- Each cell in the array is a **C**harge **C**oupled **D**evice (CCD)
  - light-sensitive diode that converts photons to electrons
- CMOS is becoming more popular (esp. in cell phones)
  - http://electronics.howstuffworks.com/digital-camera.htm

Issues with digital cameras

Noise

- big difference between consumer vs. SLR-style cameras
- low light is where you most notice noise

Compression

- creates artifacts except in uncompressed formats (tiff, raw)

Color

- color fringing artifacts from Bayer patterns

Blooming

- charge overflowing into neighboring pixels

In-camera processing

- oversharpening can produce halos

Interlaced vs. progressive scan video

- even/odd rows from different exposures

Are more megapixels better?

- requires higher quality lens
- noise issues

Stabilization

- compensate for camera shake (mechanical vs. electronic)

More info online, e.g.,

- http://electronics.howstuffworks.com/digital-camera.htm
- http://www.dpreview.com/

Projection

Mapping from the world (3d) to an image (2d)

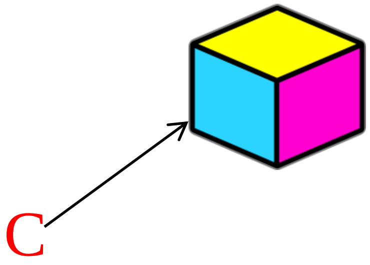
- Can we have a 1-to-1 mapping?
- How many possible mappings are there?

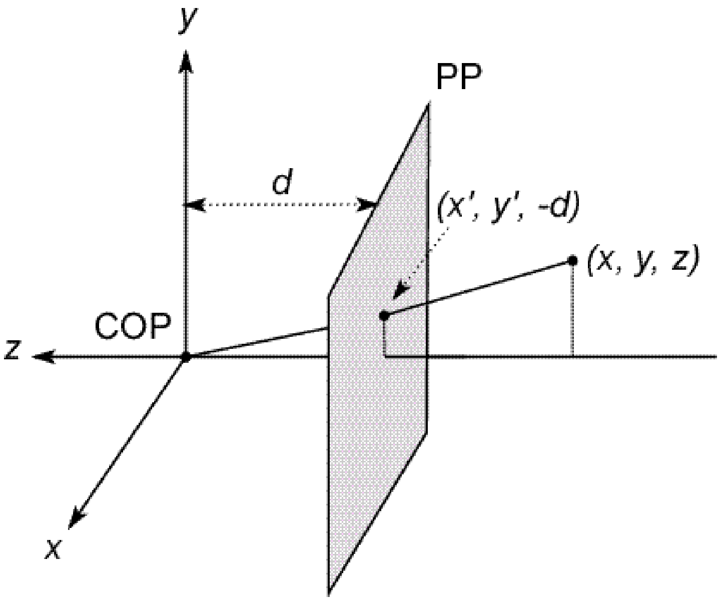
An optical system defines a particular projection. We’ll talk about 2:

1. Perspective projection (how we see “normally”)

2. Orthographic projection (e.g., telephoto lenses)

Modeling projection





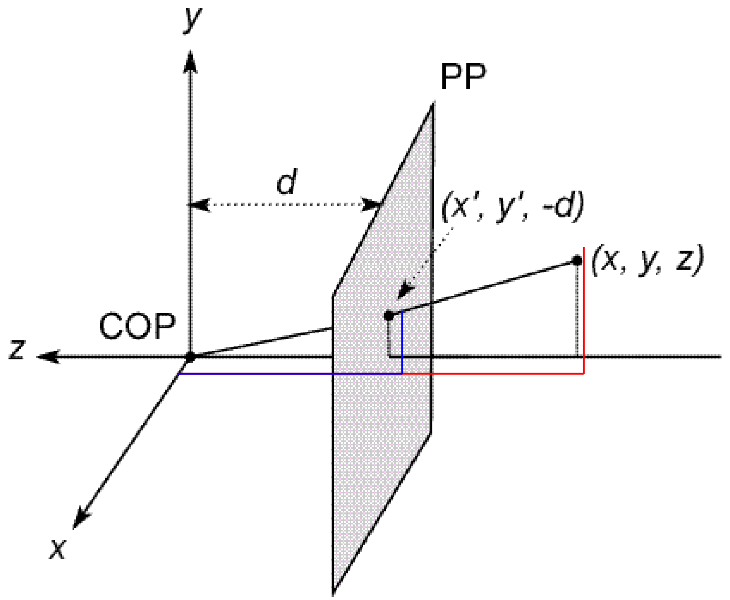
3D point

negative z axis

The coordinate system

- We will use the pin-hole model as an approximation
- Put the optical center (**C**enter **O**f **P**rojection) at the origin
- Put the image plane (**P**rojection **P**lane) *in front* of the COP
- The camera looks down the *negative* z axis
  - we need this if we want right-handed-coordinates

Modeling projection



y/z = y´/-d
y´ = -d(y/z)

Projection equations

- Compute intersection with PP of ray from (x,y,z) to COP
- Derived using similar triangles

$$(x, y, z) \rightarrow (-d\frac{x}{z},\ -d\frac{y}{z},\ -d)$$

- We get the projection by throwing out the last coordinate:

$$(x´,y´) \rightarrow (-d\frac{x}{z},\ -d\frac{y}{z})$$

Homogeneous coordinates

Is this a linear transformation?

- no—division by z is nonlinear

Trick: add one more coordinate:

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image coordinates

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

homogeneous scene coordinates

Converting *from* homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

Perspective Projection

Projection is a matrix multiply using homogeneous coordinates:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ -z/d \end{bmatrix} \Rightarrow \left(-d\frac{x}{z},\ \ -d\frac{y}{z}\right)$$

divide by third coordinate

projection matrix 3D point 2D point

This is known as **perspective projection**

- The matrix is the **projection matrix**

Perspective Projection Example

1. Object point at (10, 6, 4), d=2

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/2 & 0 \end{bmatrix} \begin{bmatrix} 10 \\ 6 \\ 4 \\ 1 \end{bmatrix} = \begin{bmatrix} 10 & 6 & -2 \end{bmatrix}$$

$$\boxed{\Rightarrow \; x' = -5, \; y' = -3}$$

2. Object point at (25, 15, 10)

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/2 & 0 \end{bmatrix} \begin{bmatrix} 25 \\ 15 \\ 10 \\ 1 \end{bmatrix} = \begin{bmatrix} 25 & 15 & -5 \end{bmatrix}$$

$$\boxed{\Rightarrow \; x' = -5, \; y' = -3}$$

Perspective projection is not 1-to-1!

Perspective Projection

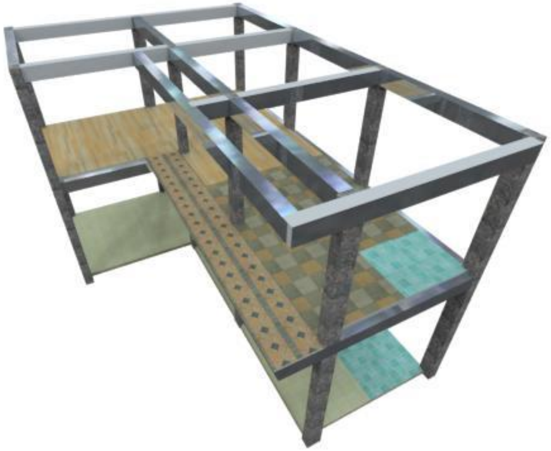
How does scaling the projection matrix change the transformation?

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ -z/d \end{bmatrix} \Rightarrow \left(-d\frac{x}{z}, \quad -d\frac{y}{z}\right)$$

$$\begin{bmatrix} -d & 0 & 0 & 0 \\ 0 & -d & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} -dx \\ -dy \\ z \end{bmatrix} \Rightarrow \left(-d\frac{x}{z}, \quad -d\frac{y}{z}\right)$$

SAME

Perspective Projection



- What happens to parallel lines?
- What happens to angles?
- What happens to distances?

Perspective Projection

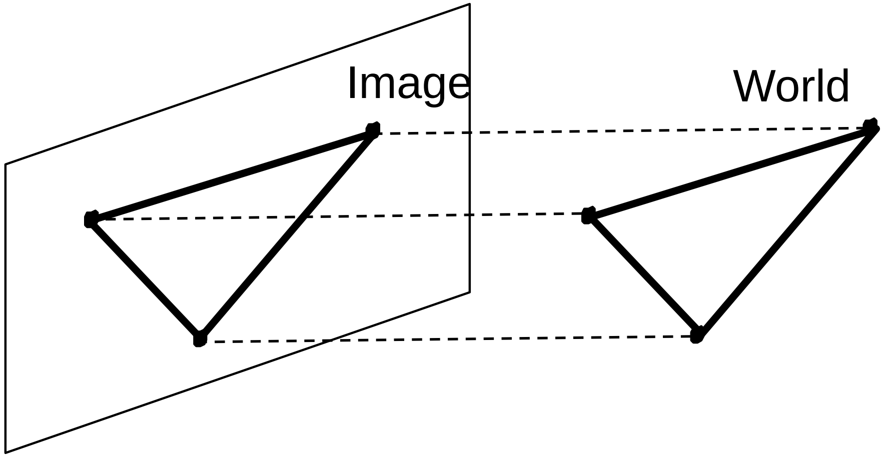
What happens when d➔∞?

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ -z/d \end{bmatrix} \Rightarrow (-d\frac{x}{z}, \quad -d\frac{y}{z})$$

Orthographic projection

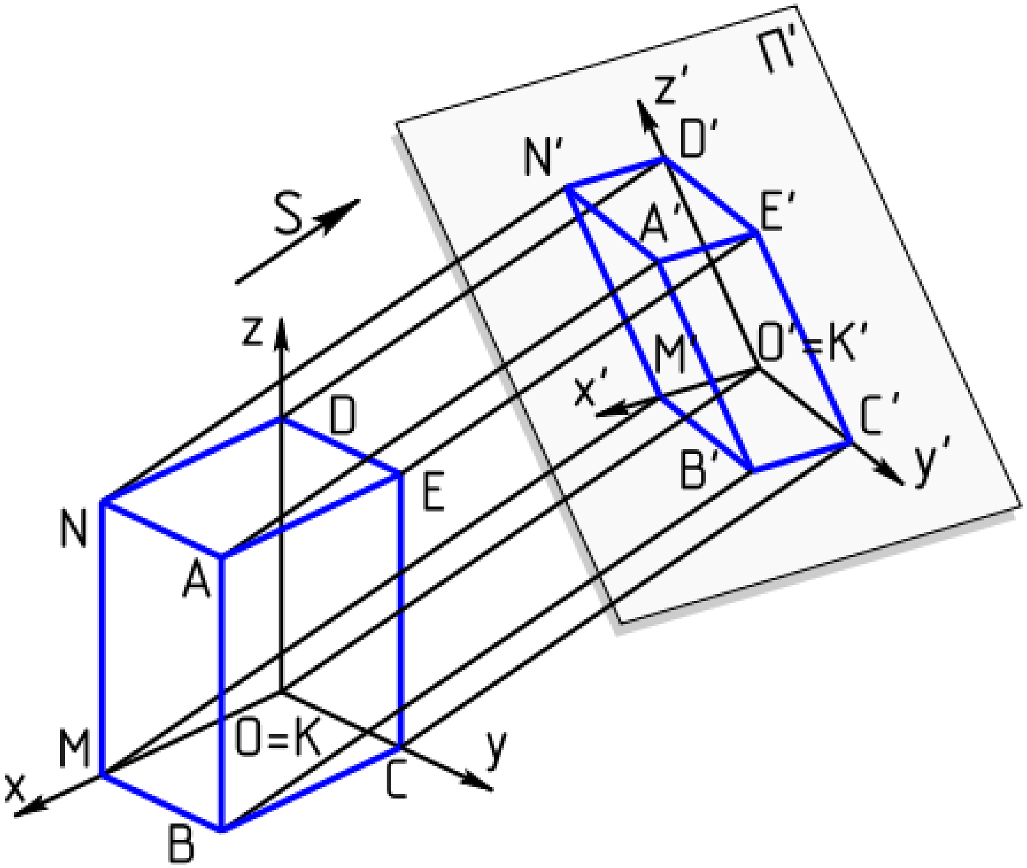
Special case of perspective projection

- Distance from the COP to the PP is infinite



- Good approximation for telephoto optics
- Also called “parallel projection”: (x, y, z) → (x, y)
- What’s the projection matrix?

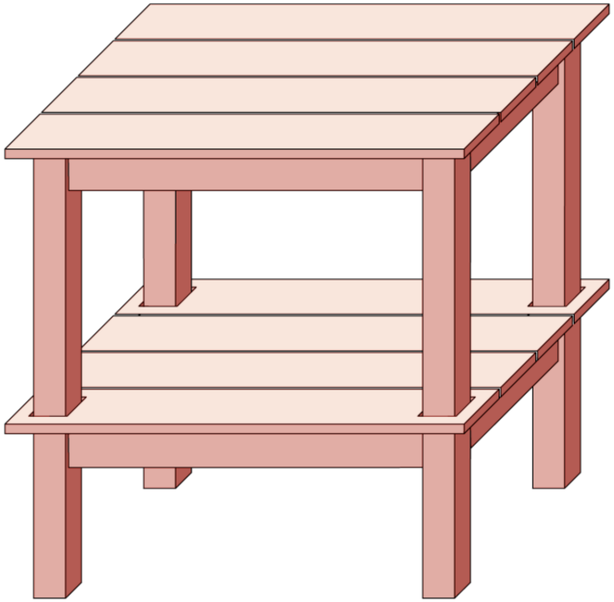
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow (x, y)$$

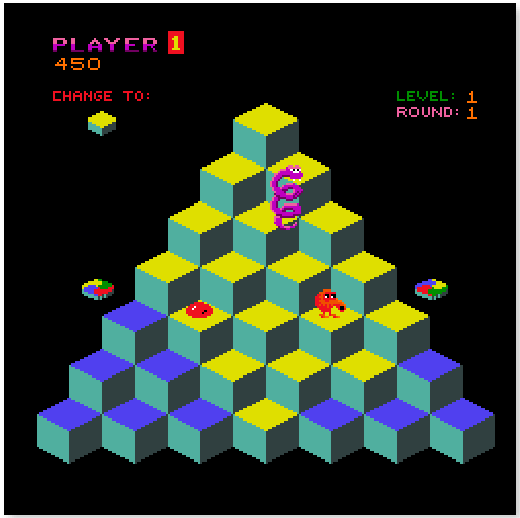


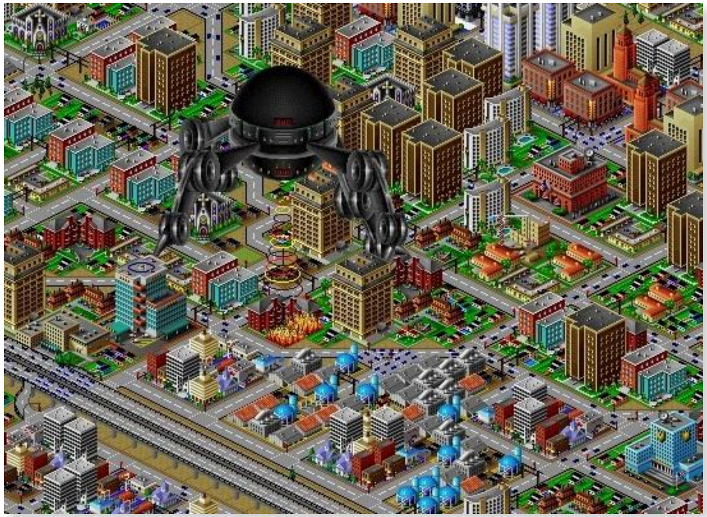
2D Parallel Projection

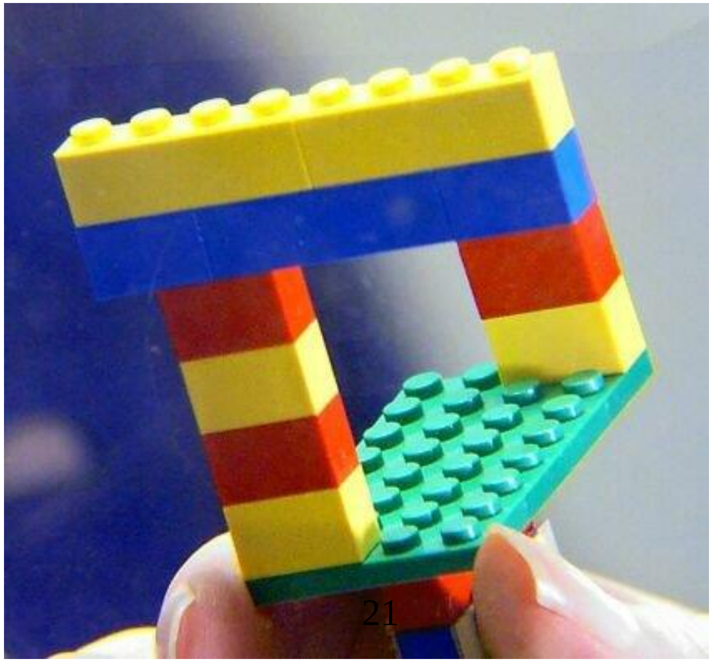
3D

Orthographic Projection









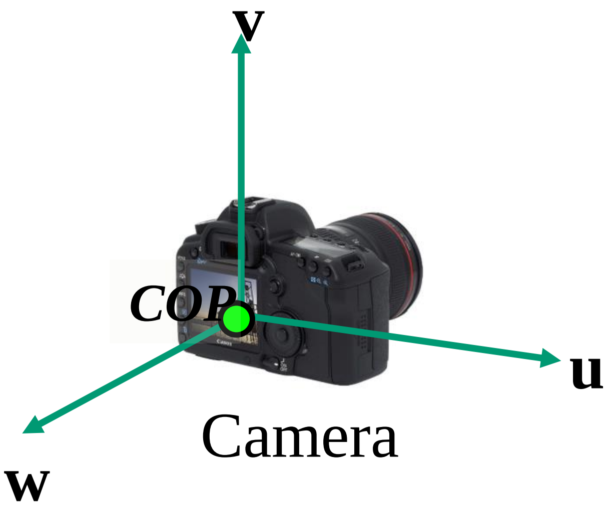
- What happens to parallel lines?
- What happens to angles?
- What happens to distances?

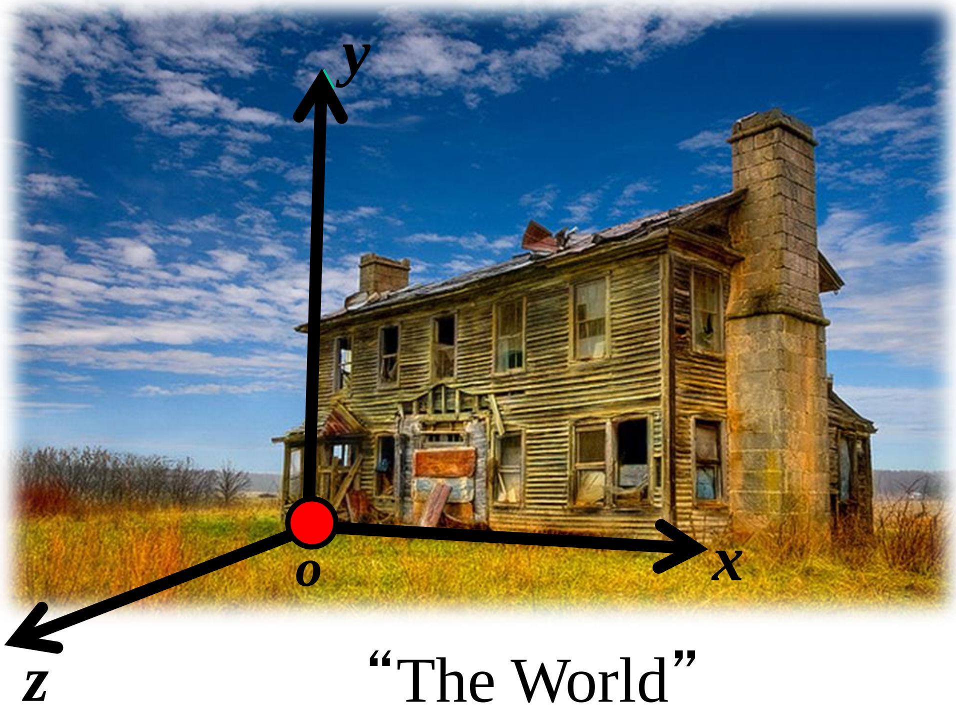
Camera parameters

How many numbers do we need to describe a camera?

- We need to describe its *pose* in the world
- We need to describe its internal *parameters*

A Tale of Two Coordinate Systems





Two important coordinate systems:

1. *World* coordinate system
2. *Camera* coordinate system

Camera parameters

- To project a point (*x*,*y*,*z*) in *world* coordinates into a camera
- First transform (*x*,*y*,*z*) into *camera* coordinates
- Need to know
  - Camera position (in world coordinates)
  - Camera orientation (in world coordinates)
- Then project into the image plane
  - Need to know camera *intrinsics*
- These can all be described with matrices

3D Translation

- 3D translation is just like 2D with one more coordinate

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$= [x+tx, y+ty, z+tz, 1]^T$$

3D Rotation (just the 3 x 3 part shown)

About X axis: $\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$

About Y: $\begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$

About Z axis: $\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

General (orthonormal) rotation matrix used in practice:

$$\begin{bmatrix} r11 & r12 & r13 \\ r21 & r22 & r23 \\ r31 & r32 & r33 \end{bmatrix}$$

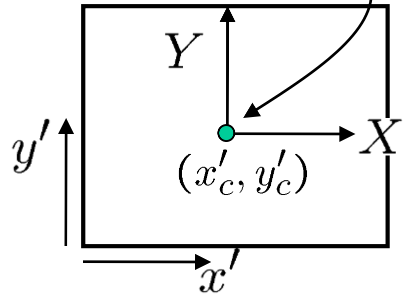
Camera parameters

A camera is described by several parameters

- Translation T of the optical center from the origin of world coords
- Rotation R of the image plane
- focal length f, principal point (x'_c, y'_c), pixel size (s_x, s_y)
- blue parameters are called “extrinsics,” red are “intrinsics”

Projection equation

$$\mathbf{x} = \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{\Pi X}$$



- The projection matrix models the cumulative effect of all parameters
- Useful to decompose into a series of operations

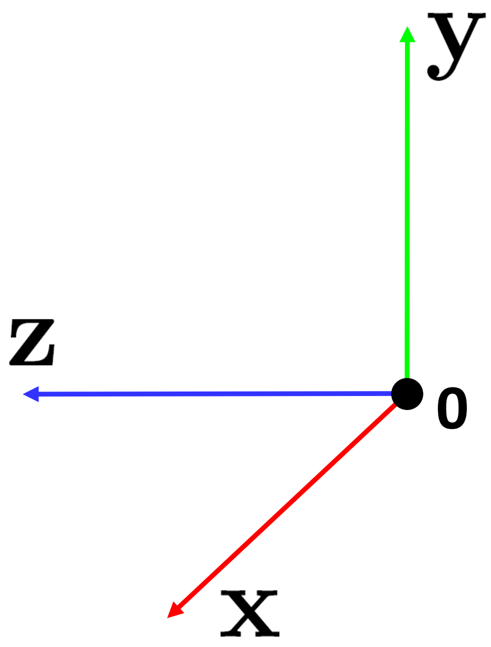
$$\mathbf{\Pi} = \underbrace{\begin{bmatrix} -fs_x & 0 & x'_c \\ 0 & -fs_y & y'_c \\ 0 & 0 & 1 \end{bmatrix}}_{\text{intrinsics}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{0}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} \overset{\text{identity matrix}}{\mathbf{I}_{3x3}} & \mathbf{T}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}}_{\text{translation}} \leftarrow [tx, ty, tz]^T$$

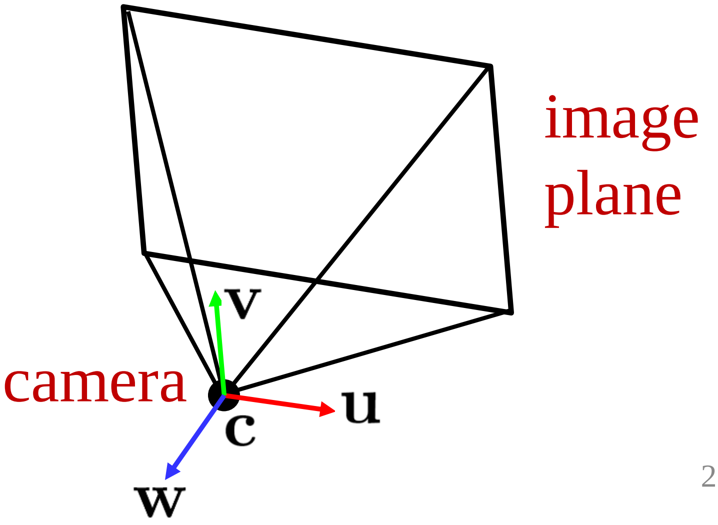
- The definitions of these parameters are **not** completely standardized
  - especially intrinsics—varies from one book to another

Extrinsics

- How do we get the camera to “canonical form”?
  - (Center of projection at the origin, x-axis points right, y-axis points up, z-axis points backwards)

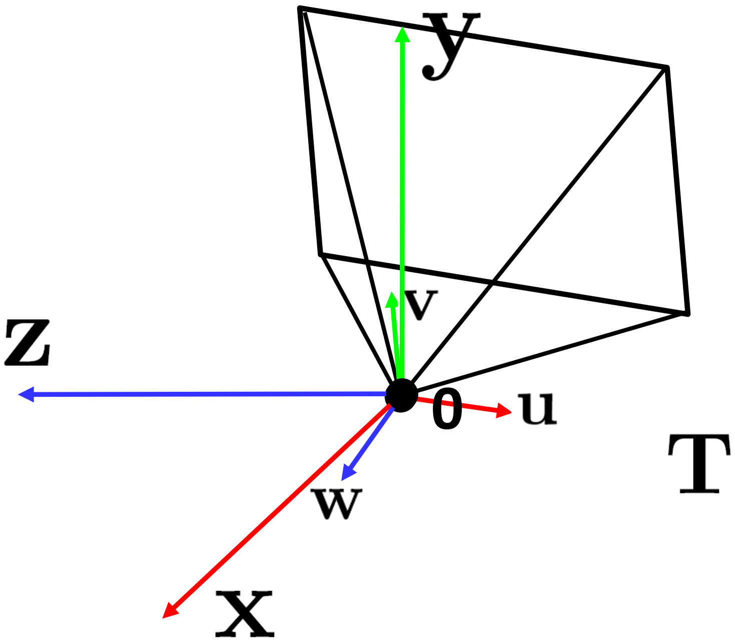
Step 1: Translate by -**c**





Extrinsics

- How do we get the camera to "canonical form"?
  - (Center of projection at the origin, x-axis points right, y-axis points up, z-axis points backwards)



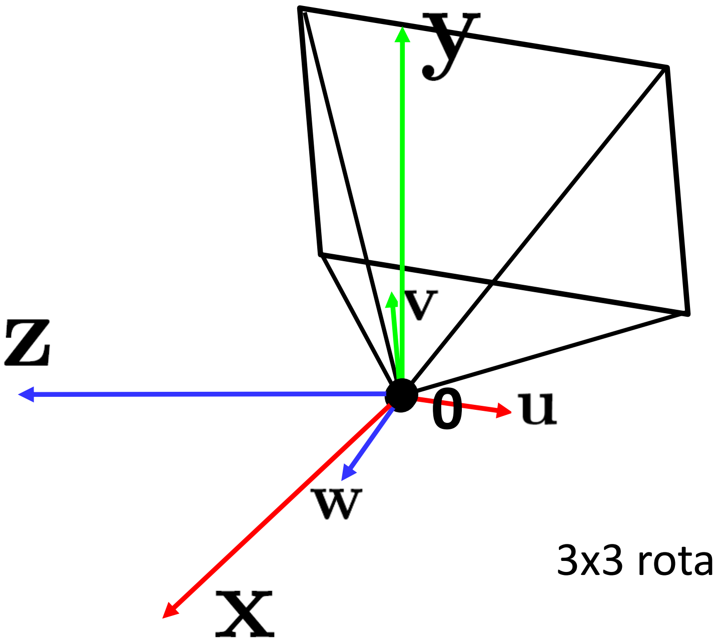
Step 1: Translate by -**c**

How do we represent translation as a matrix multiplication?

$$\mathbf{T} = \begin{bmatrix} \mathbf{I}_{3\times 3} & -\mathbf{c} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}$$

Extrinsics

- How do we get the camera to “canonical form”?
  - (Center of projection at the origin, x-axis points right, y-axis points up, z-axis points backwards)



Step 1: Translate by -**c**

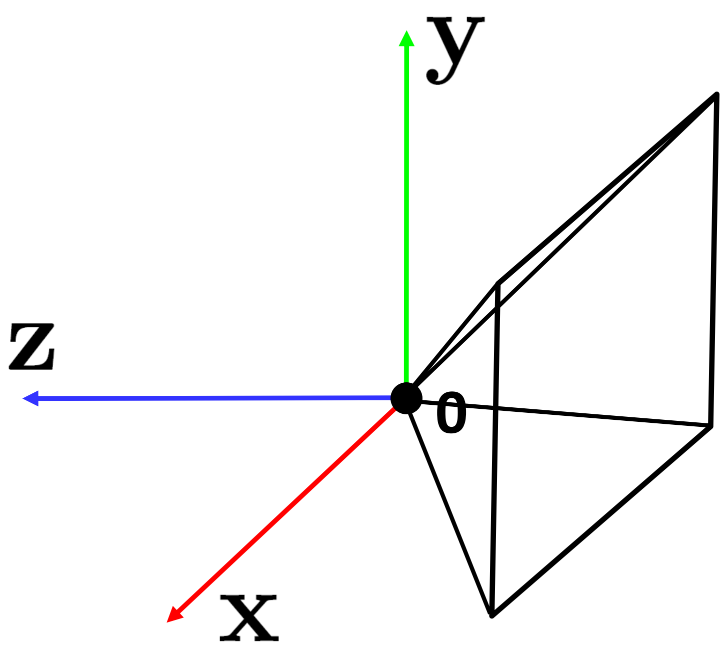
Step 2: Rotate by **R**

$$\mathbf{R} = \begin{bmatrix} \mathbf{u}^T \\ \mathbf{v}^T \\ \mathbf{w}^T \end{bmatrix}$$

3x3 rotation matrix

Extrinsics

- How do we get the camera to “canonical form”?
  - (Center of projection at the origin, x-axis points right, y-axis points up, z-axis points backwards)



Step 1: Translate by -**c**

Step 2: Rotate by **R**

$$\mathbf{R} = \begin{bmatrix} \mathbf{u}^T \\ \mathbf{v}^T \\ \mathbf{w}^T \end{bmatrix}$$

Perspective projection

$$\underbrace{\begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

(intrinsics)

(converts from 3D rays in camera coordinate system to pixel coordinates)

in general, $\mathbf{K} = \begin{bmatrix} -f & s & c_x \\ 0 & -\alpha f & c_y \\ 0 & 0 & 1 \end{bmatrix}$

f is the focal length of the camera

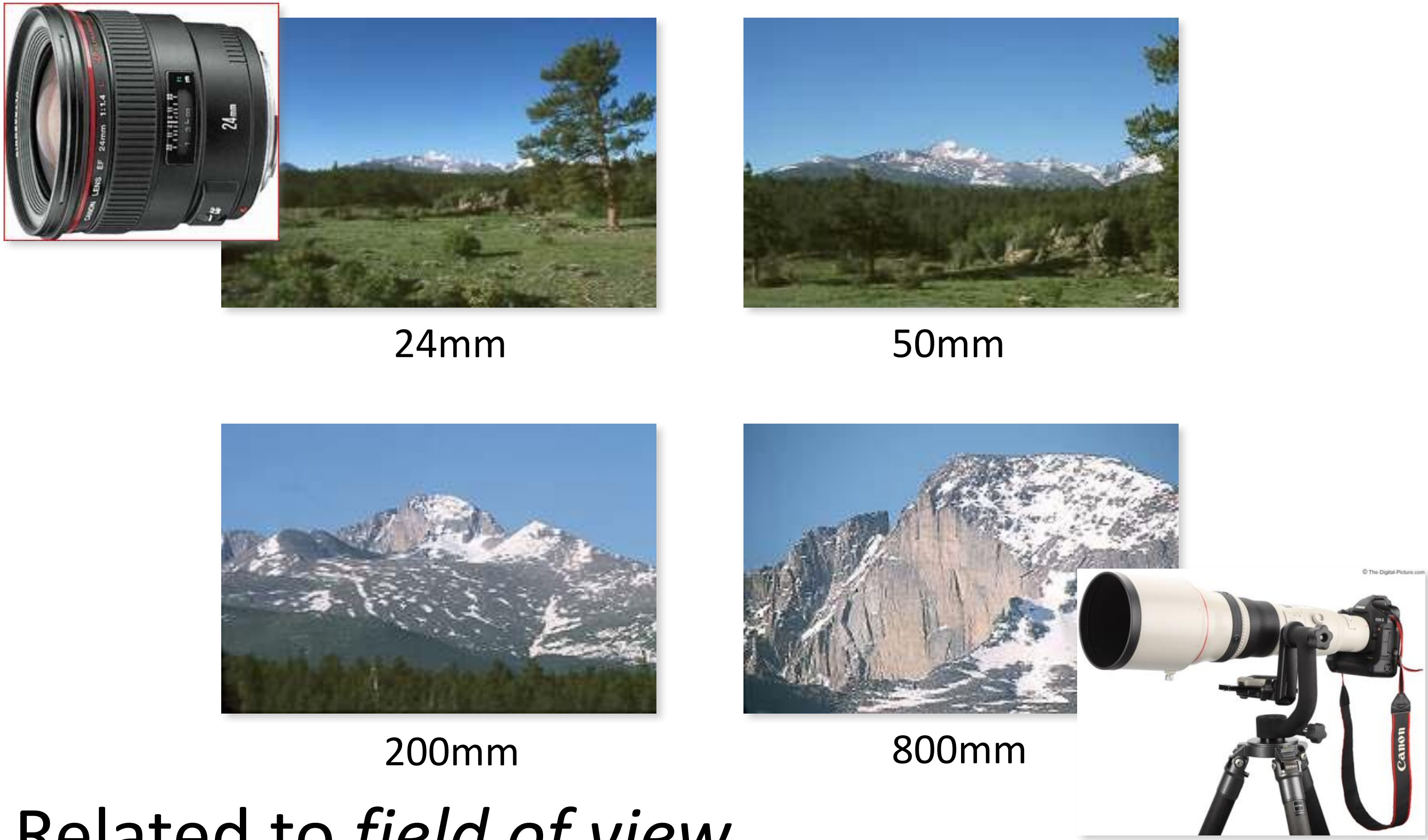
α : **aspect ratio** (1 unless pixels are not square)

s : **skew** (0 unless pixels are shaped like rhombi/parallelograms)

(c_x, c_y) : **principal point** ((0,0) unless optical axis doesn’t intersect projection plane at origin)

Focal length

- Can think of as “zoom”



24mm

50mm

200mm

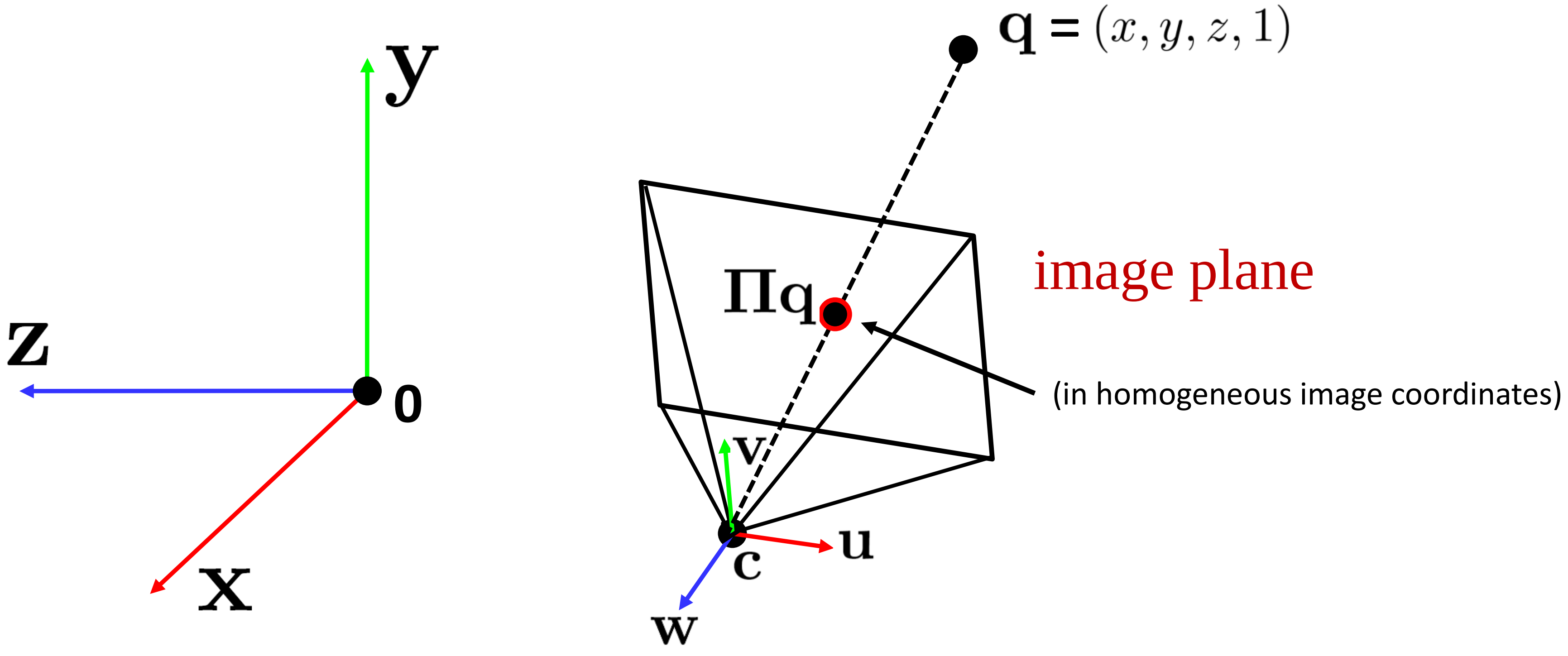
800mm

- Related to *field of view*

Projection matrix

$$\boldsymbol{\Pi} = \underset{\text{intrinsics}}{\mathbf{K}} \underbrace{\underset{\text{projection}}{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\end{bmatrix}} \underset{\text{rotation}}{\begin{bmatrix}\mathbf{R} & \begin{matrix}0\\0\\0\end{matrix}\\ 0\quad 0\quad 0 & 1\end{bmatrix}} \underset{\text{translation}}{\begin{bmatrix}\mathbf{I}_{3\times 3} & -\mathbf{c}\\ 0\quad 0\quad 0 & 1\end{bmatrix}}}_{}$$

Projection matrix

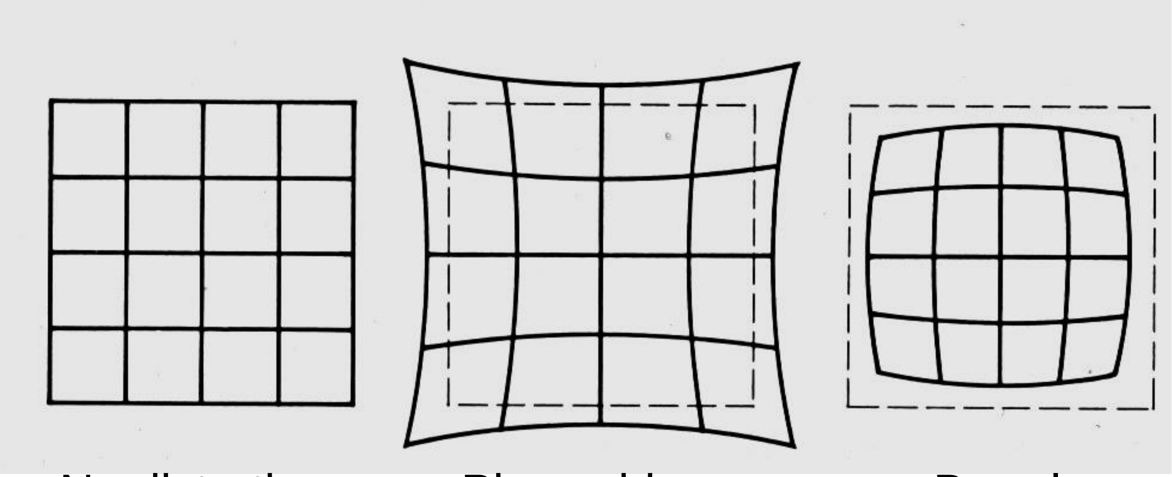


arbitrary 3D point

image plane

(in homogeneous image coordinates)

Distortion

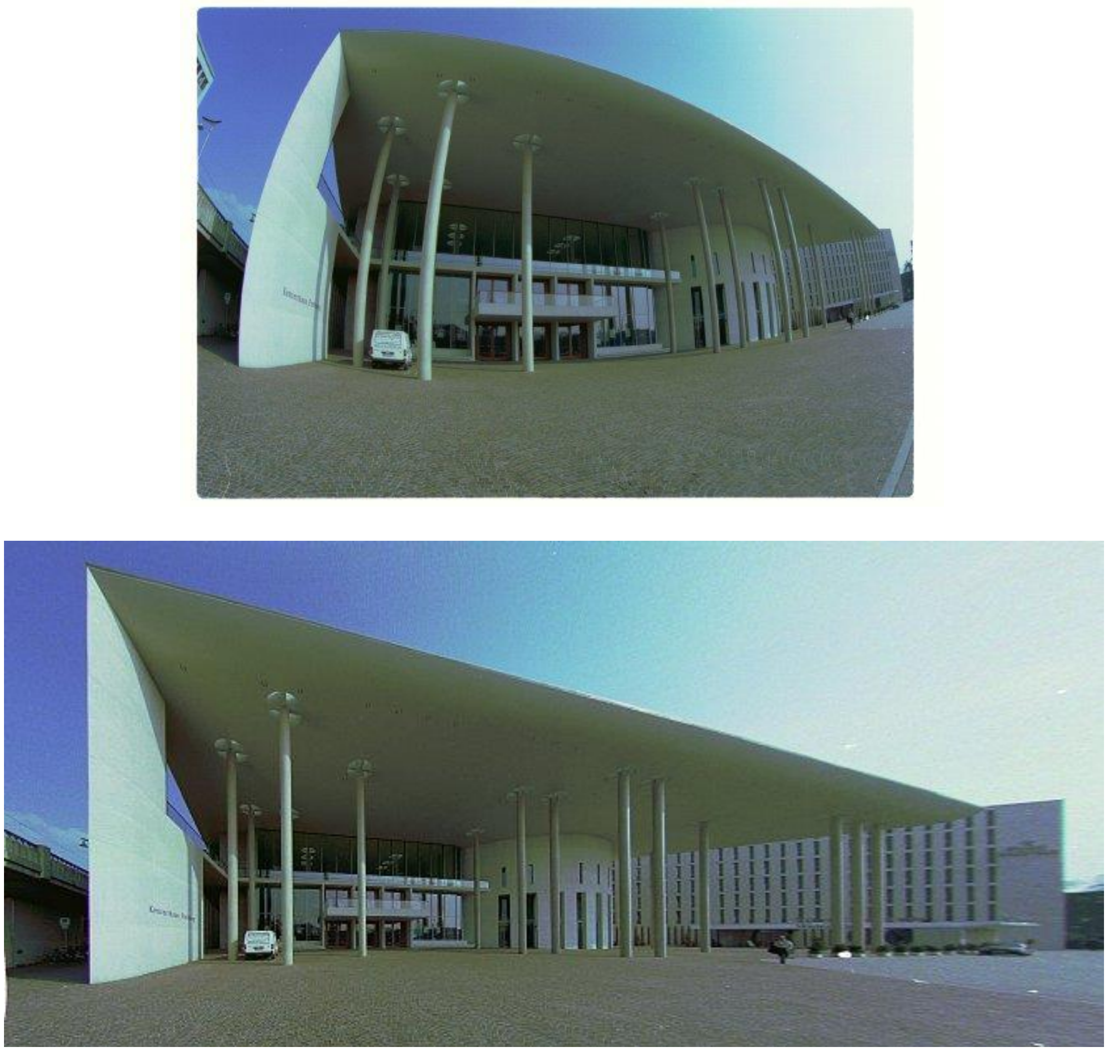


No distortion Pin cushion Barrel

Radial distortion of the image

- Caused by imperfect lenses
- Deviations are most noticeable for rays that pass through the edge of the lens

Correcting radial distortion

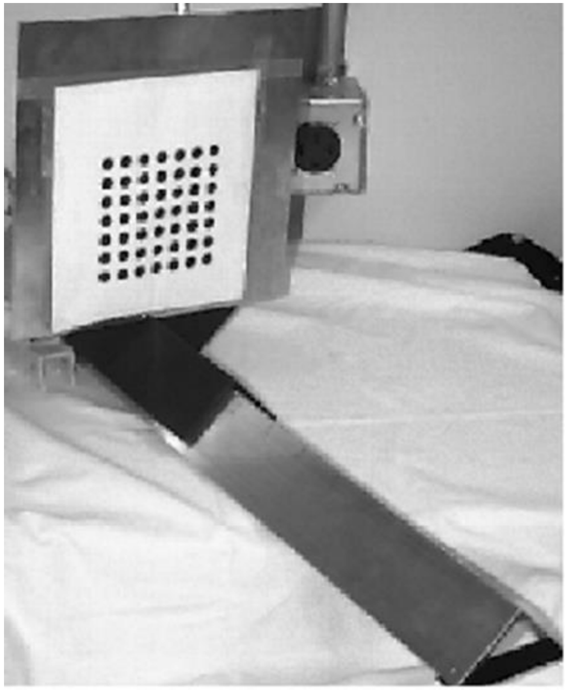


from Helmut Dersch

Where does all this lead?

- We need it to understand stereo
- And 3D reconstruction
- It also leads into camera calibration, which is usually done in factory settings to solve for the camera parameters before performing an industrial task.
- The extrinsic parameters must be determined.
- Some of the intrinsic are given, some are solved for, some are improved.

Camera Calibration

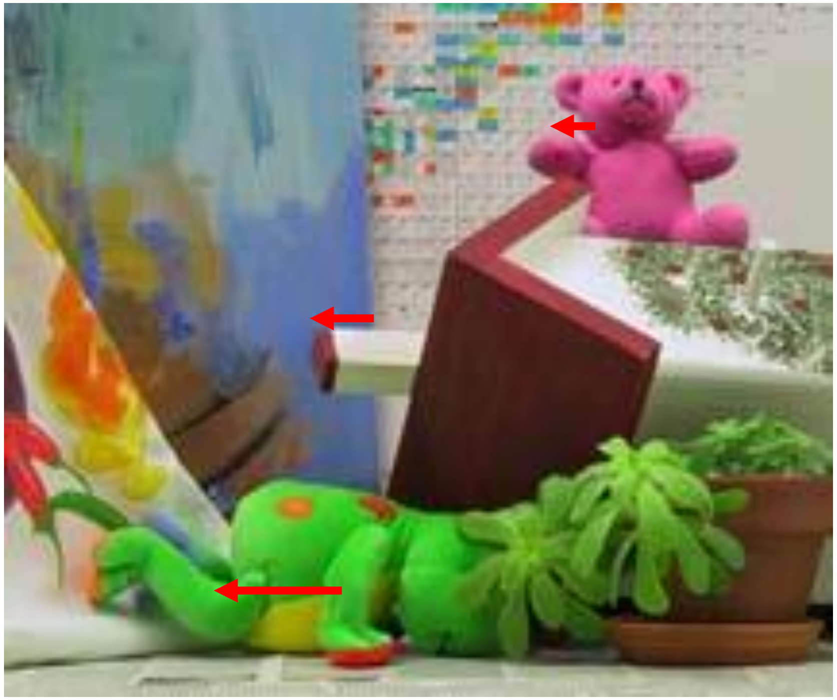


The idea is to snap images at different depths and get a lot of 2D-3D point correspondences.

x1, y1, z1, u1, v1
x2, y2, z1, u2, v2
.
.
xn, yn, zn, un, vn

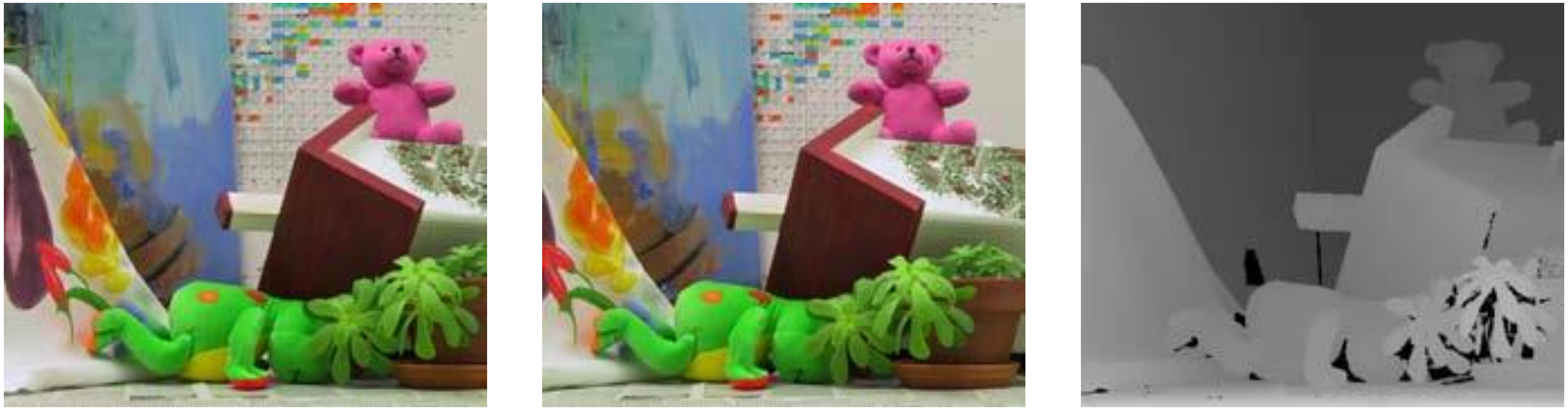
Then solve a system of equations to get camera parameters.

Stereo



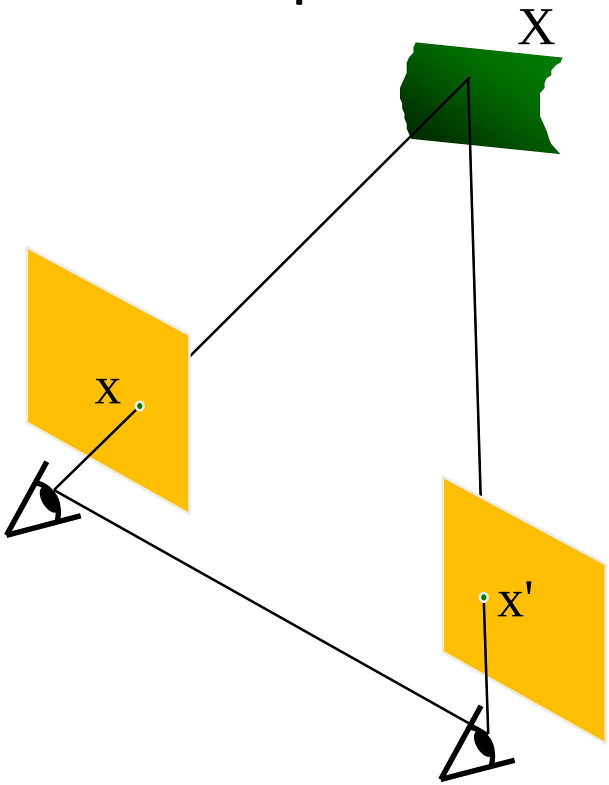
Amount of horizontal movement is …

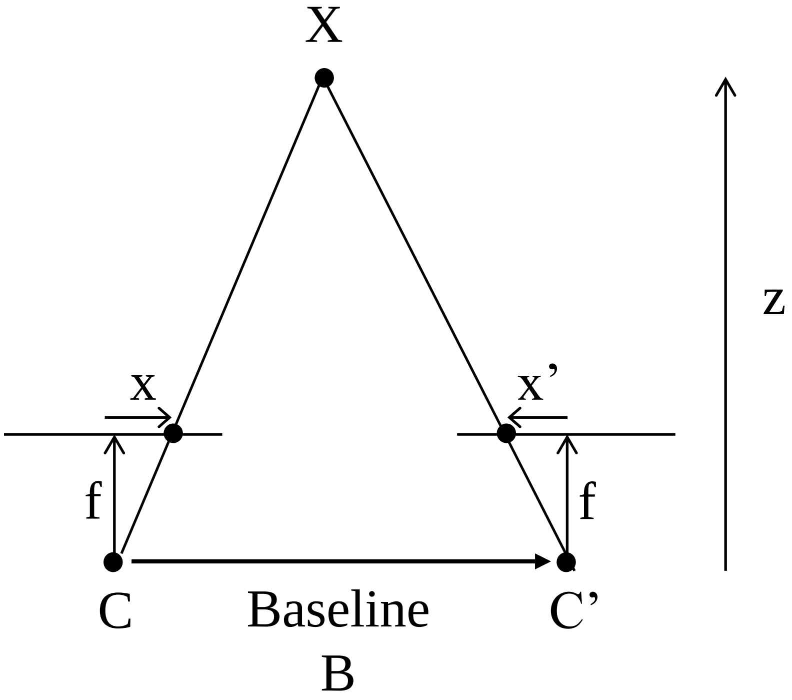
…inversely proportional to the distance from the camera



Depth from Stereo

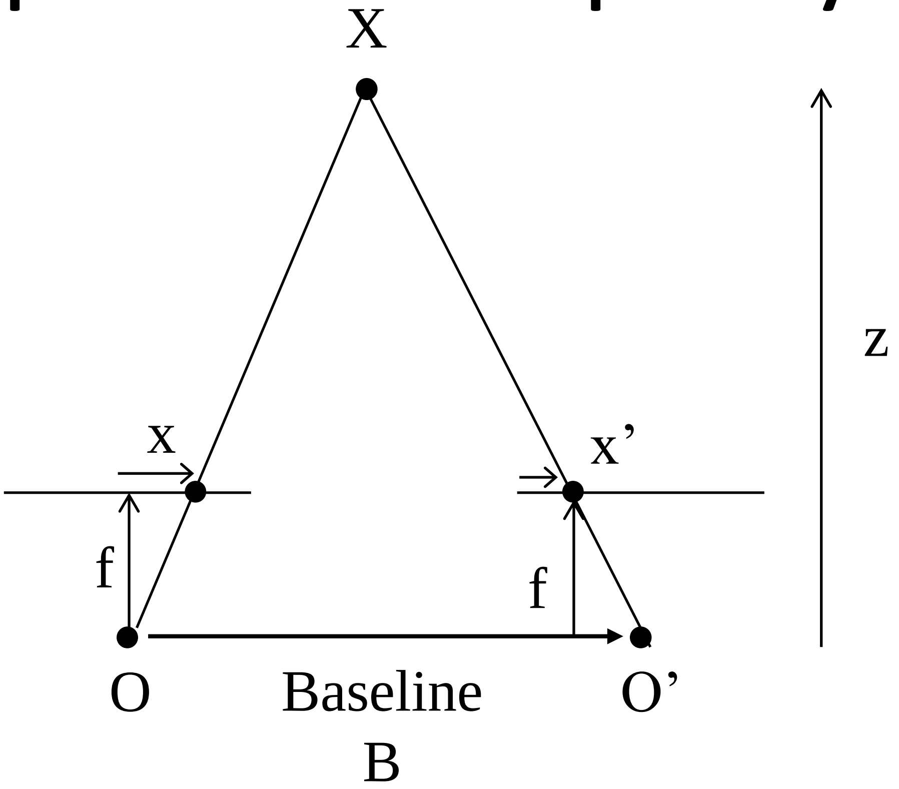
- Goal: recover depth by finding image coordinate x’ that corresponds to x





Depth from disparity

$$\boxed{\frac{x-x'}{O-O'}=\frac{f}{z}}$$

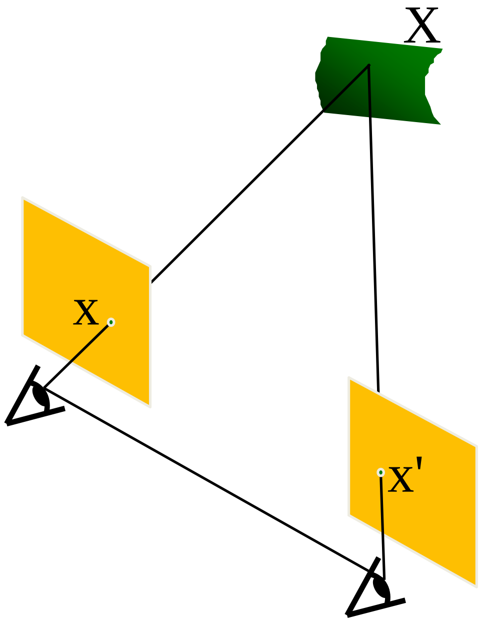


$$disparity = x - x' = \frac{B \cdot f}{z}$$

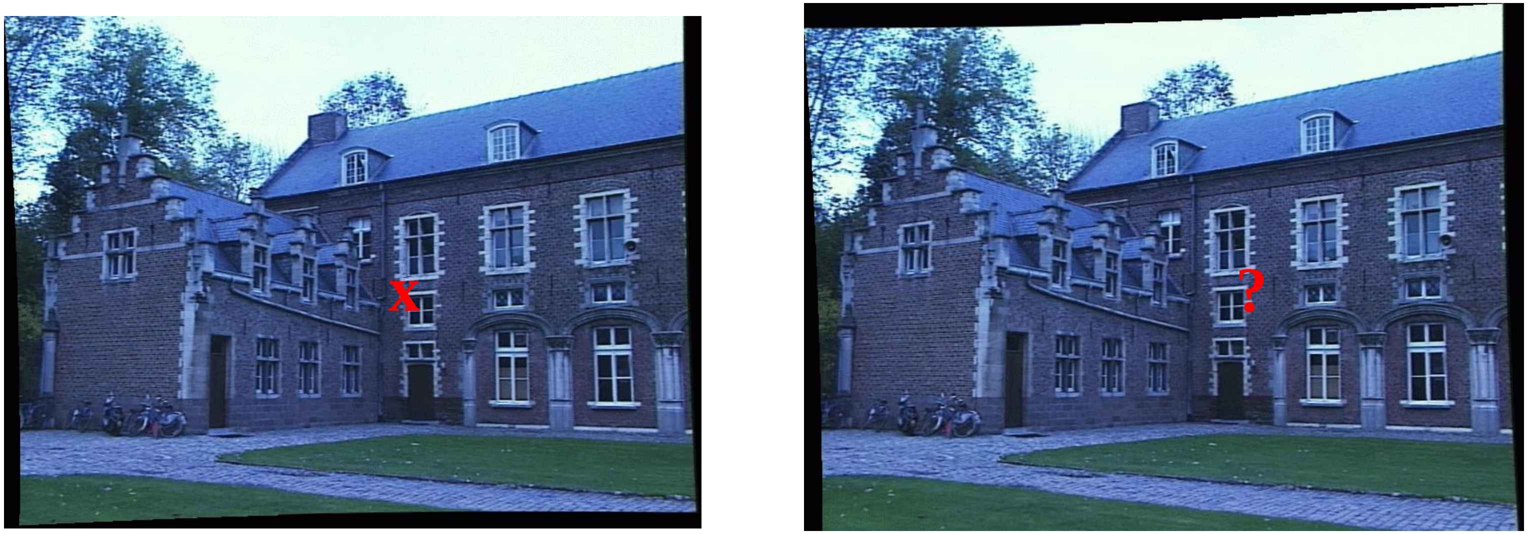
Disparity is inversely proportional to depth.

Depth from Stereo

- Goal: recover depth by finding image coordinate x’ that corresponds to x
- Sub-Problems
  1. Calibration: How do we recover the relation of the cameras (if not already known)?
  2. Correspondence: How do we search for the matching point x’?



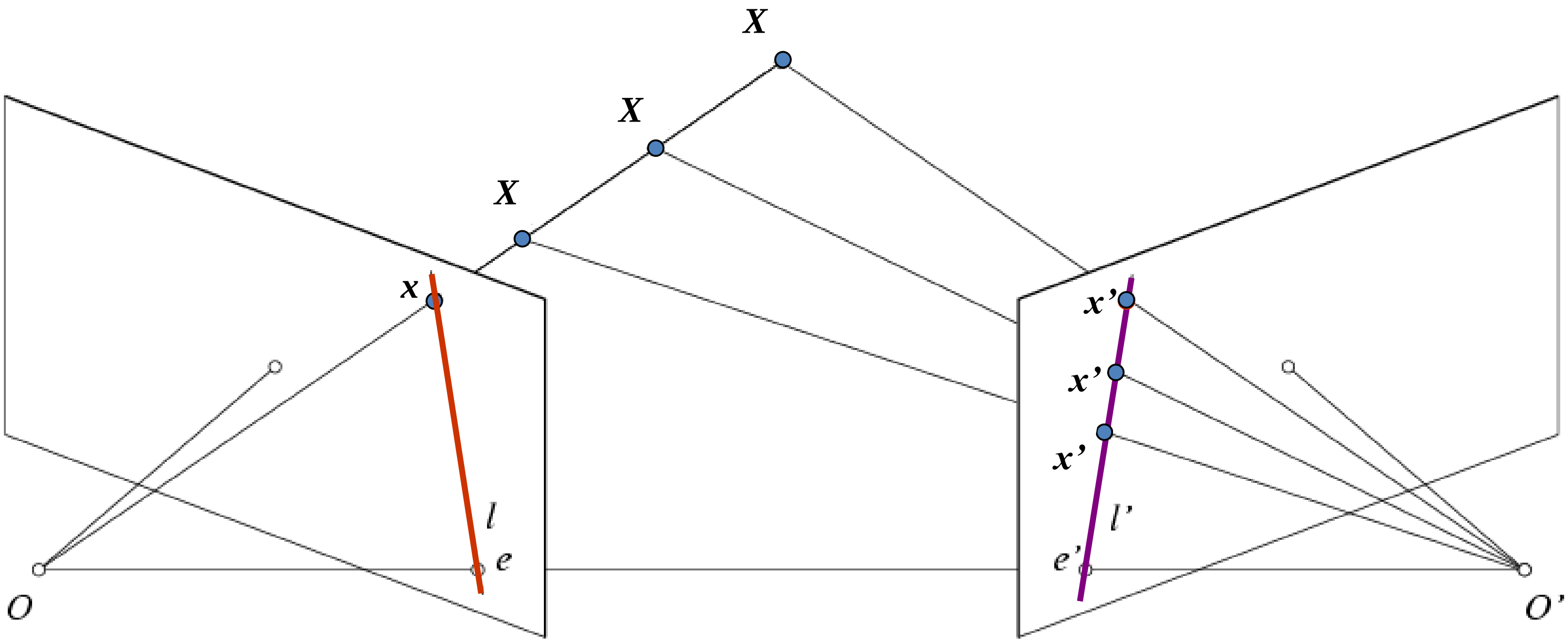
Correspondence Problem



- We have two images taken from cameras with different intrinsic and extrinsic parameters

- How do we match a point in the first image to a point in the second? How can we constrain our search?

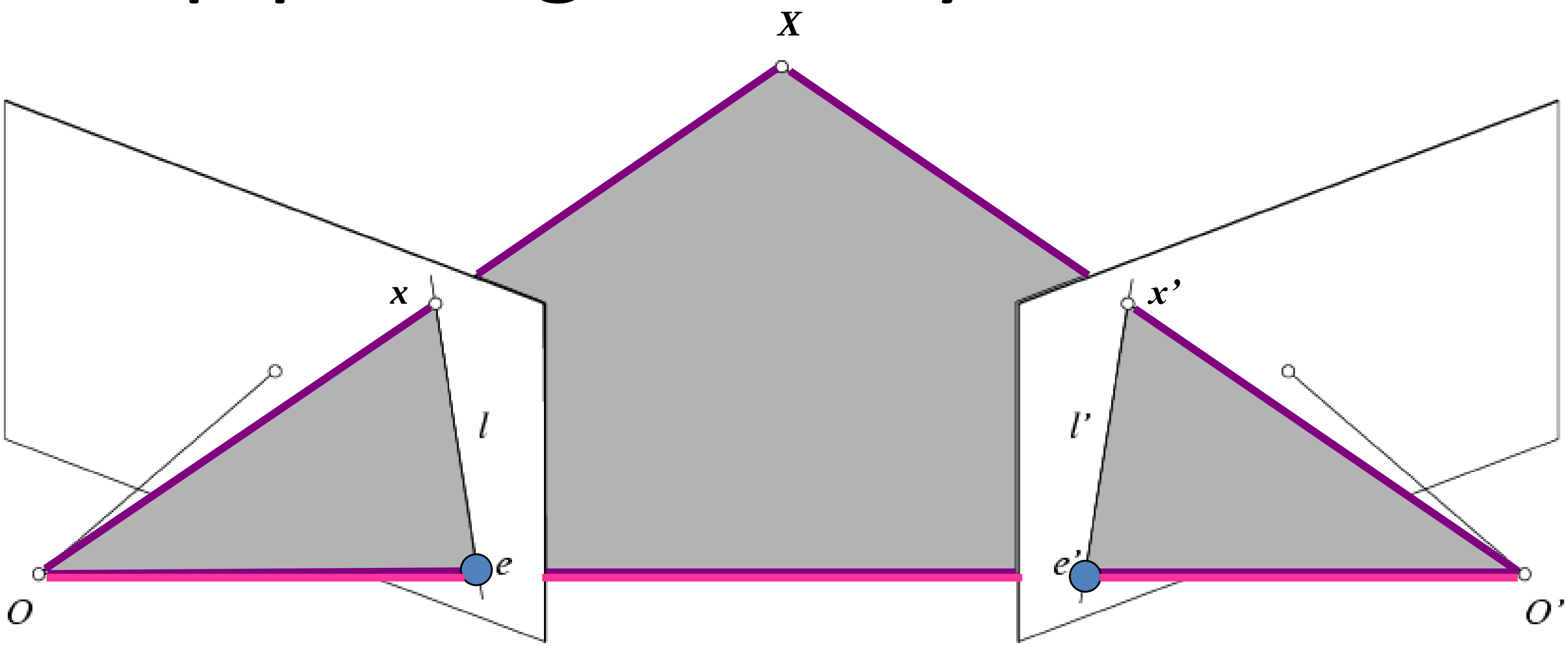
Key idea: Epipolar constraint



Potential matches for *x* have to lie on the corresponding line *l'*.

Potential matches for *x'* have to lie on the corresponding line *l*.

Epipolar geometry: notation



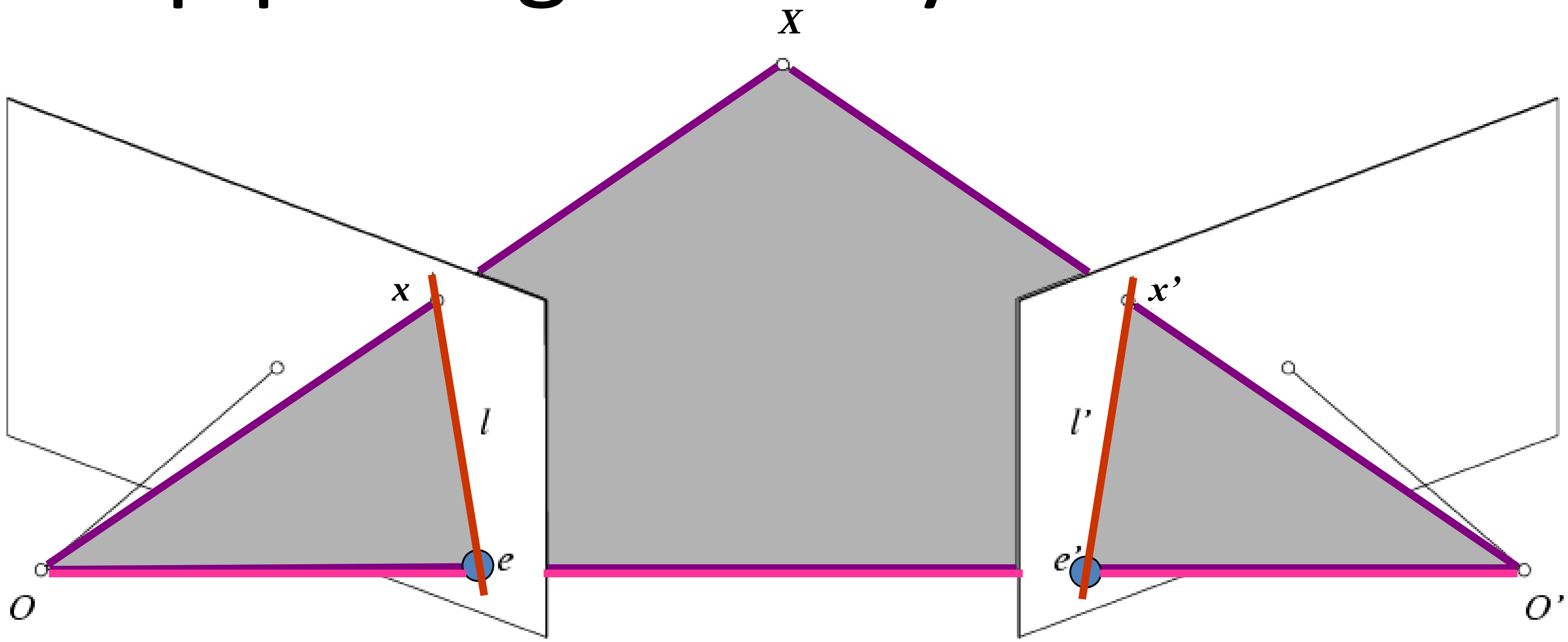
- **Baseline** – line connecting the two camera centers

- **Epipoles**

= intersections of baseline with image planes
= projections of the other camera center

- **Epipolar Plane** – plane containing baseline (1D family)

Epipolar geometry: notation



- **Baseline** – line connecting the two camera centers

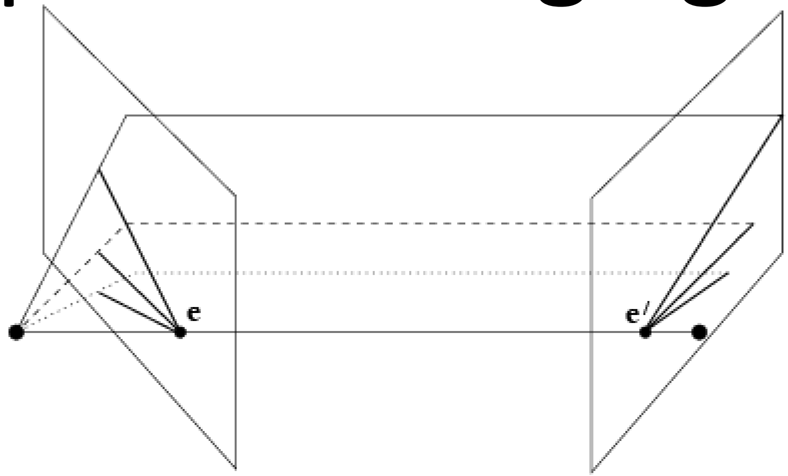
- **Epipoles**

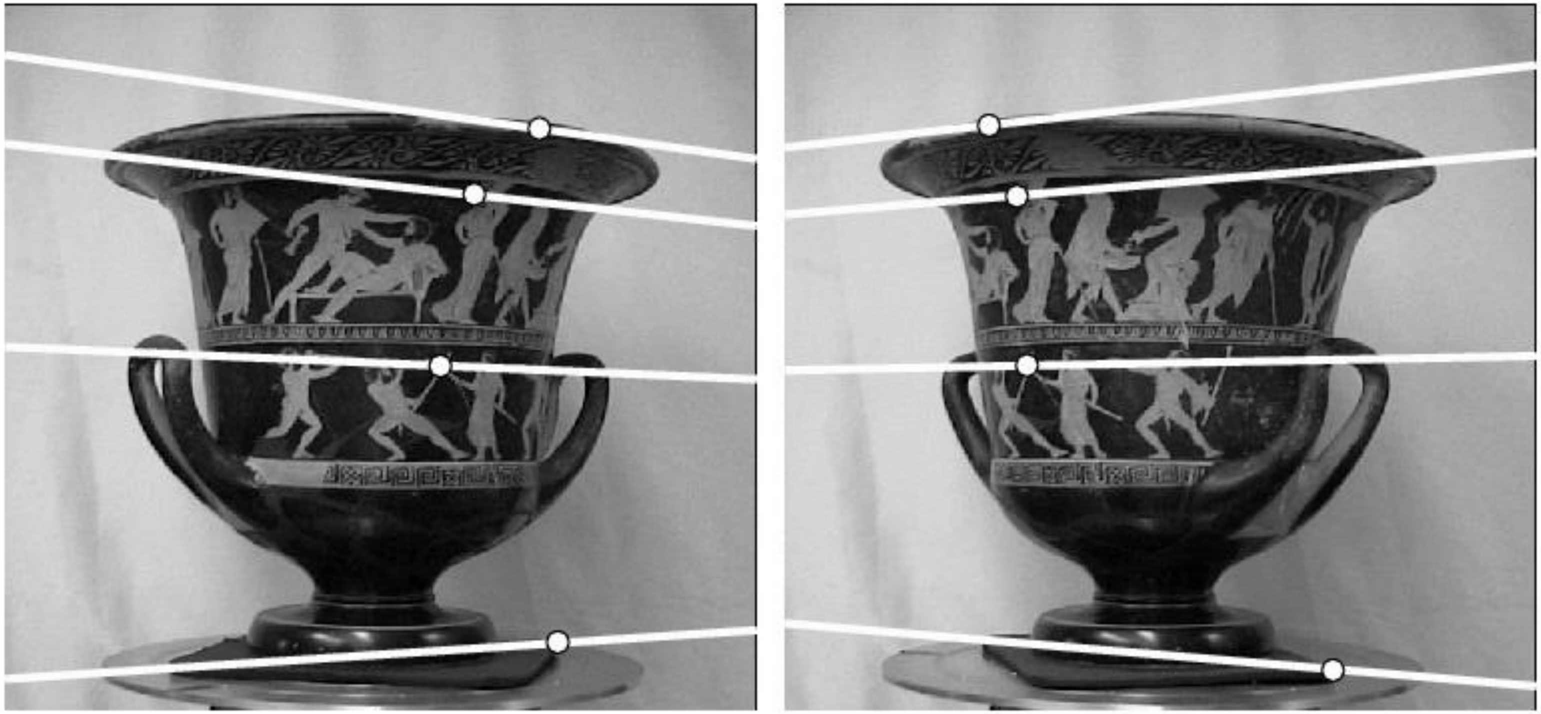
= intersections of baseline with image planes

= projections of the other camera center

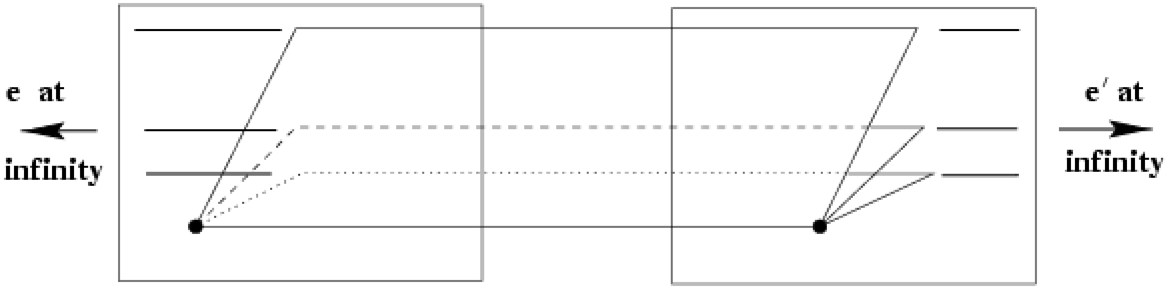
- **Epipolar Plane** – plane containing baseline (1D family)
- **Epipolar Lines -** intersections of epipolar plane with image planes (always come in corresponding pairs)

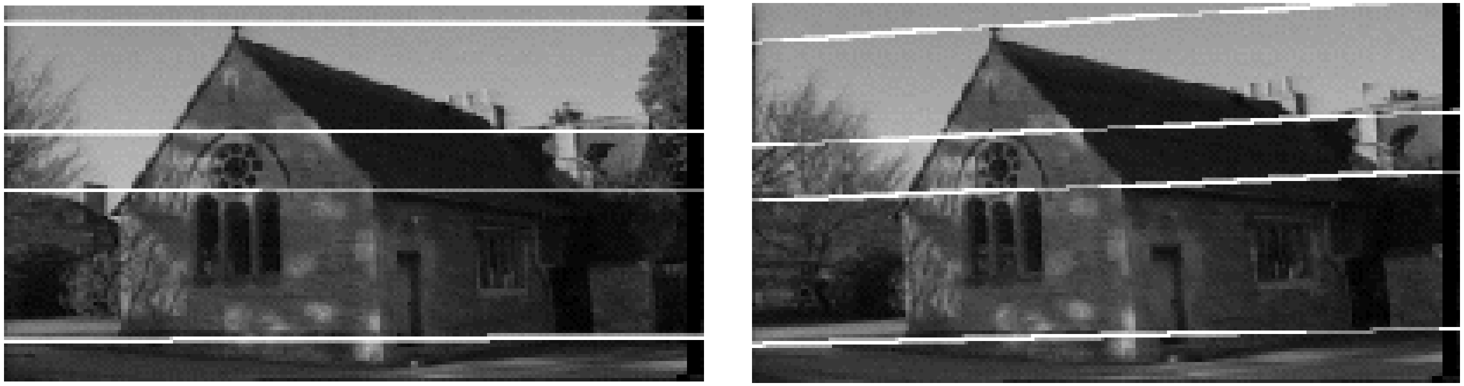
Example: Converging cameras



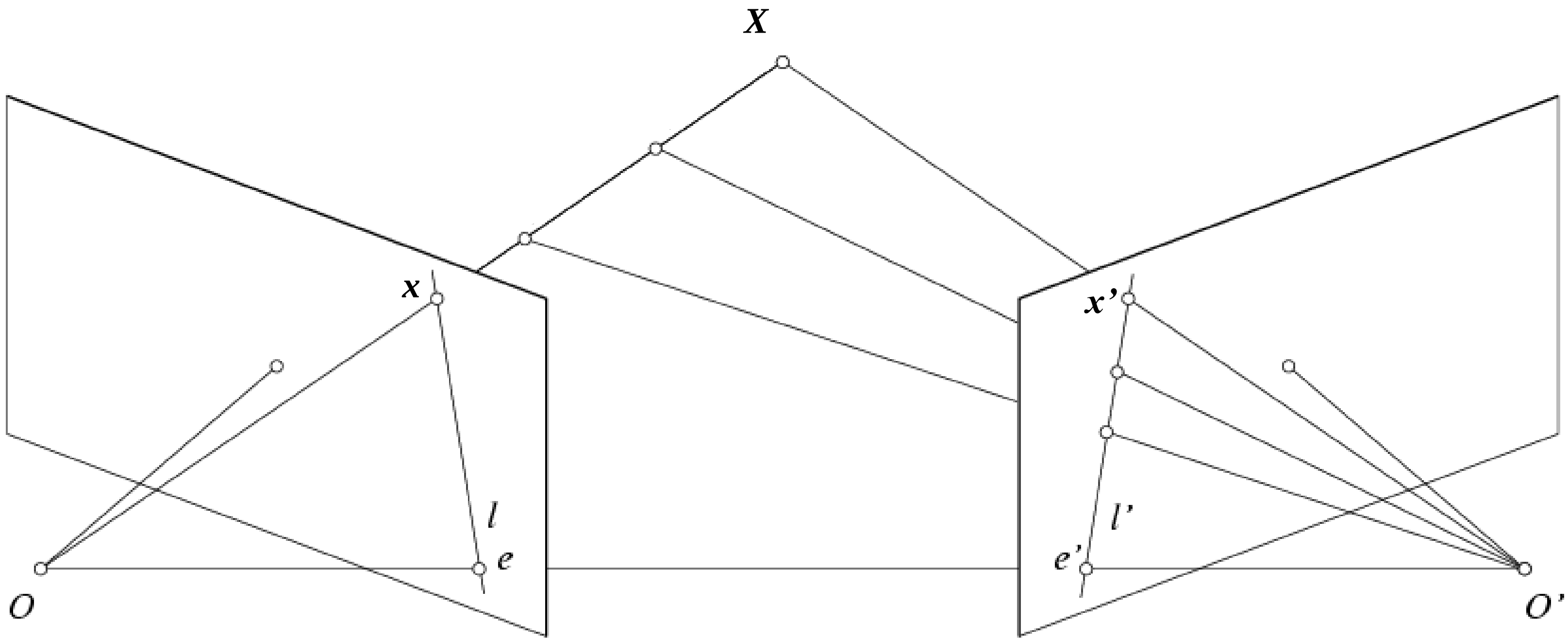


Example: Motion parallel to image plane



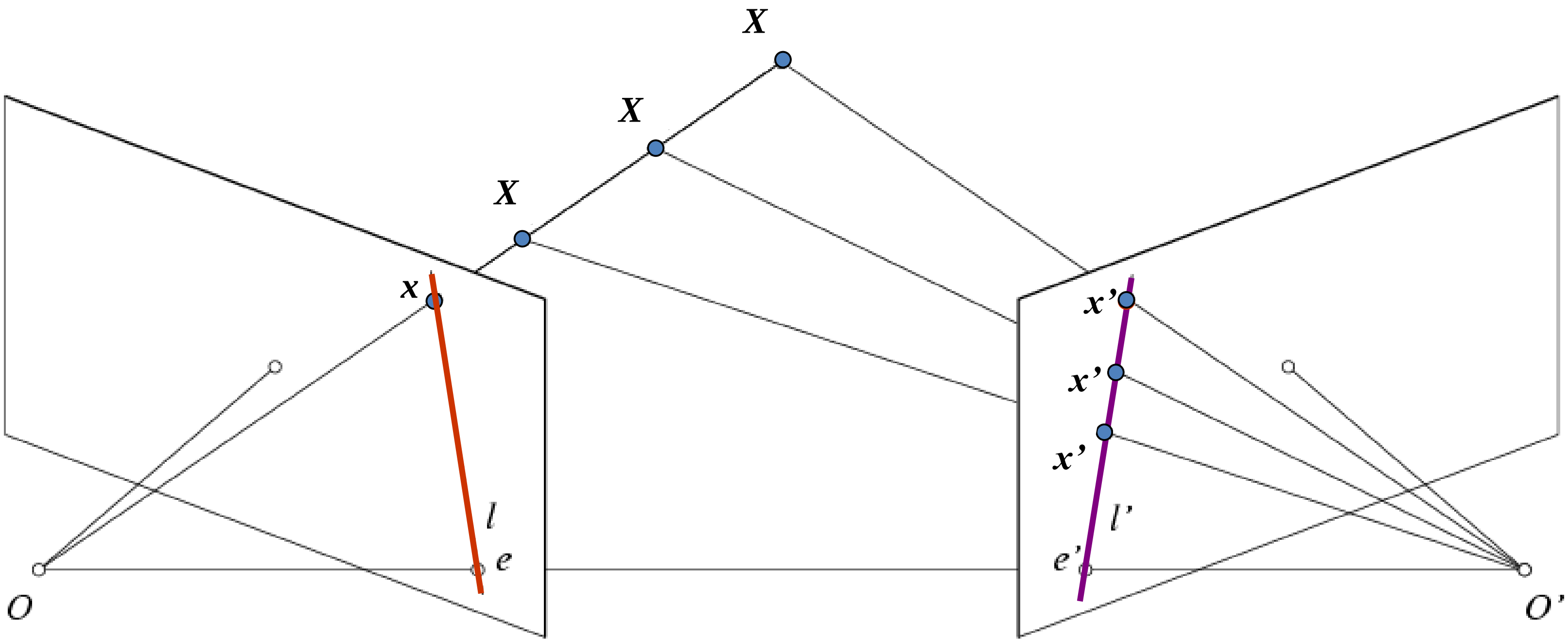


Epipolar constraint



- If we observe a point ***x*** in one image, where can the corresponding point ***x’*** be in the other image?

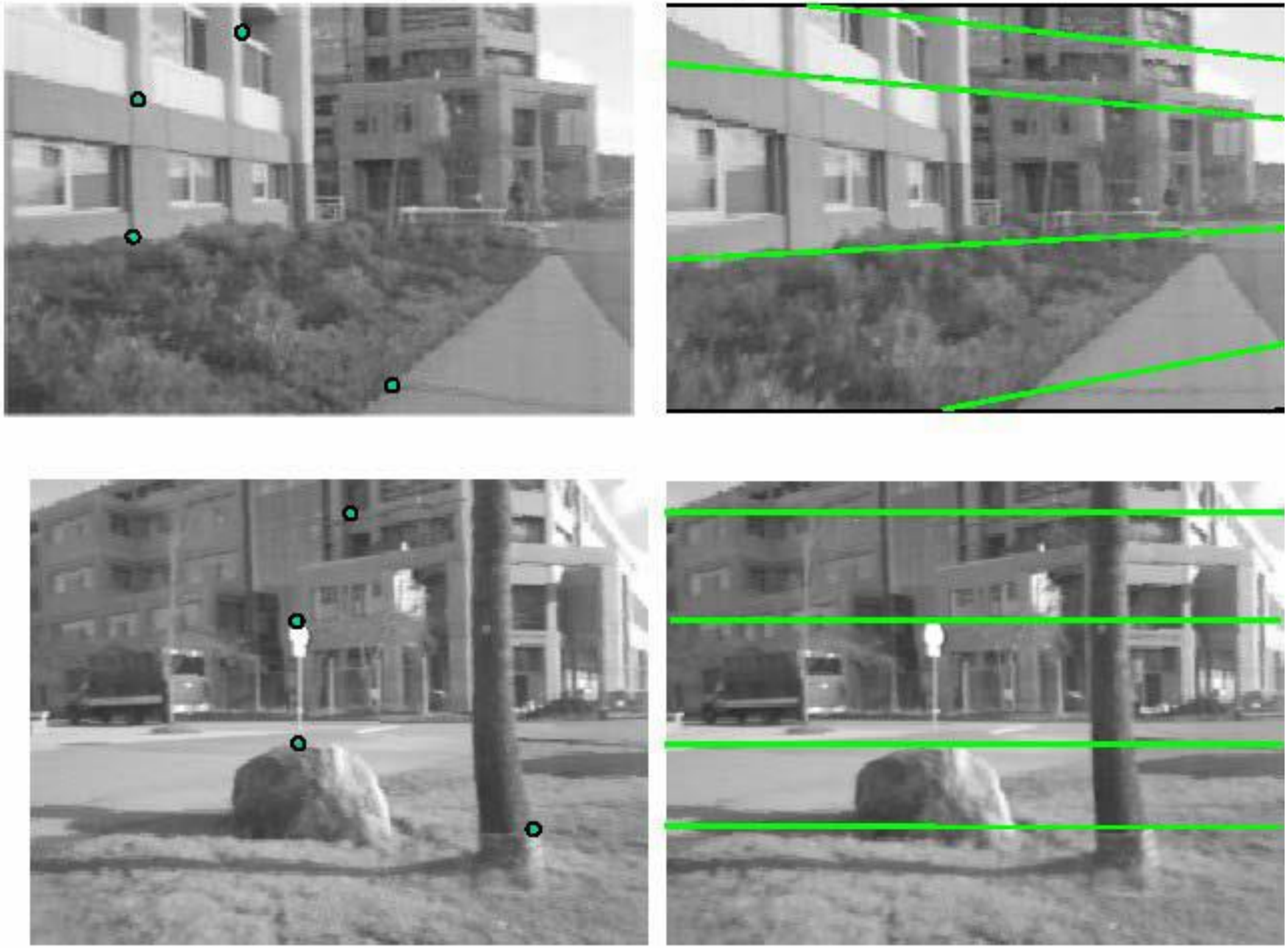
Epipolar constraint



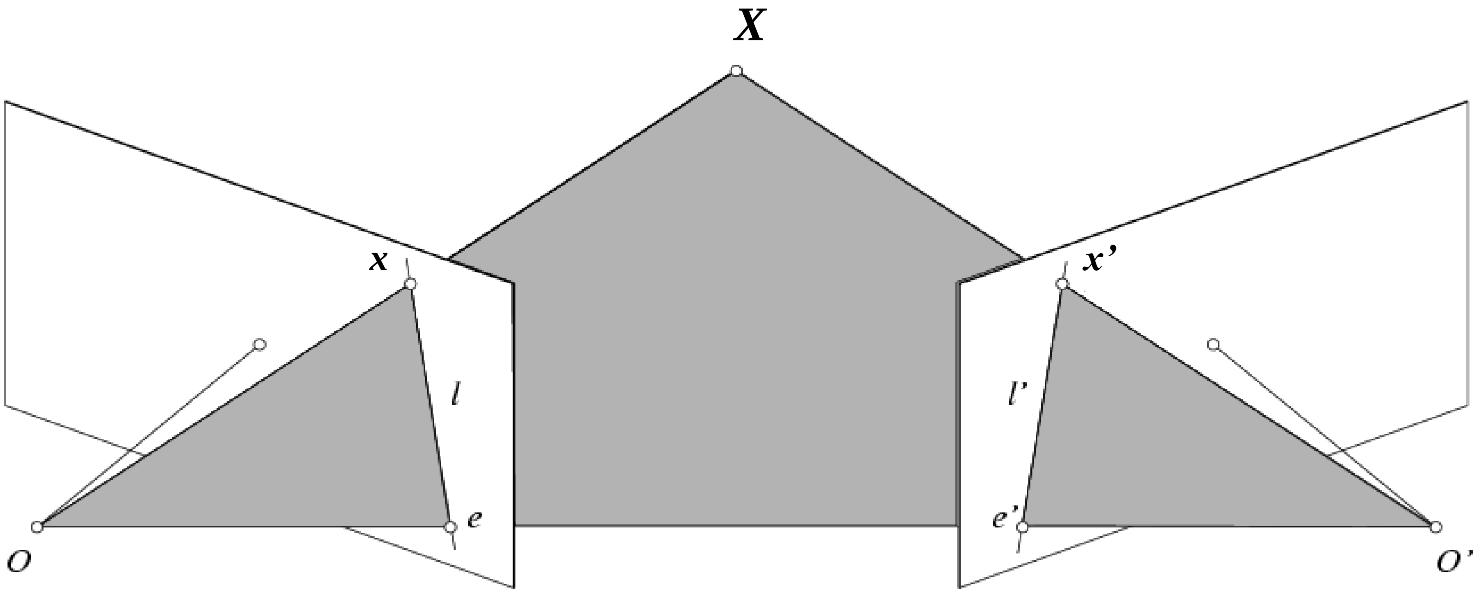
- Potential matches for ***x*** have to lie on the corresponding epipolar line ***l'***.

- Potential matches for ***x'*** have to lie on the corresponding epipolar line ***l***.

Epipolar constraint example



Epipolar constraint: Calibrated case



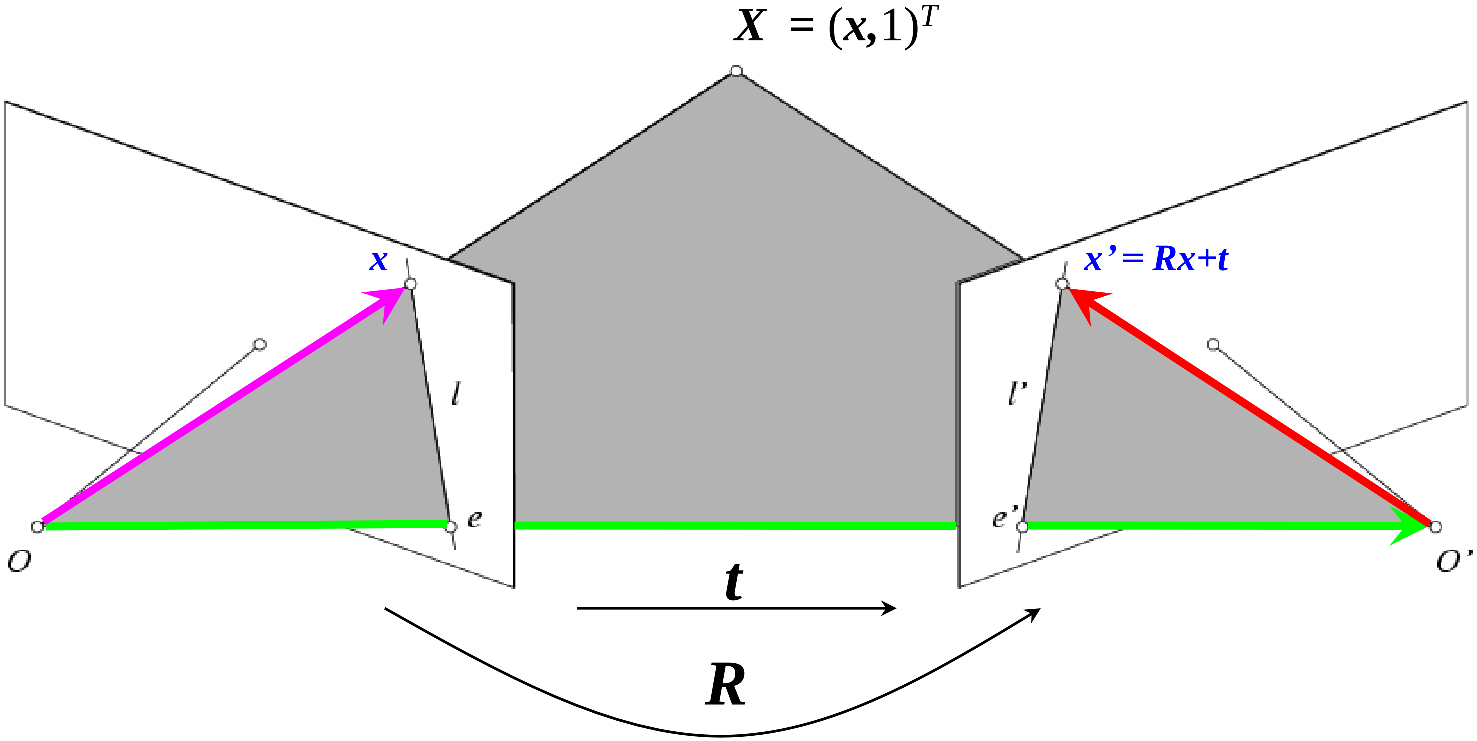
- Assume that the intrinsic and extrinsic parameters of the cameras are known
- We can multiply the projection matrix of each camera (and the image points) by the inverse of the calibration matrix to get *normalized* image coordinates
- We can also set the global coordinate system to the coordinate system of the first camera. Then the projection matrices of the two cameras can be written as $[\mathbf{I} \mid \mathbf{0}]$ and $[\mathbf{R} \mid \mathbf{t}]$

Simplified Matrices for the 2 Cameras

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \left(\, \mathbf{I} \, \middle| \, \mathbf{0} \, \right)$$

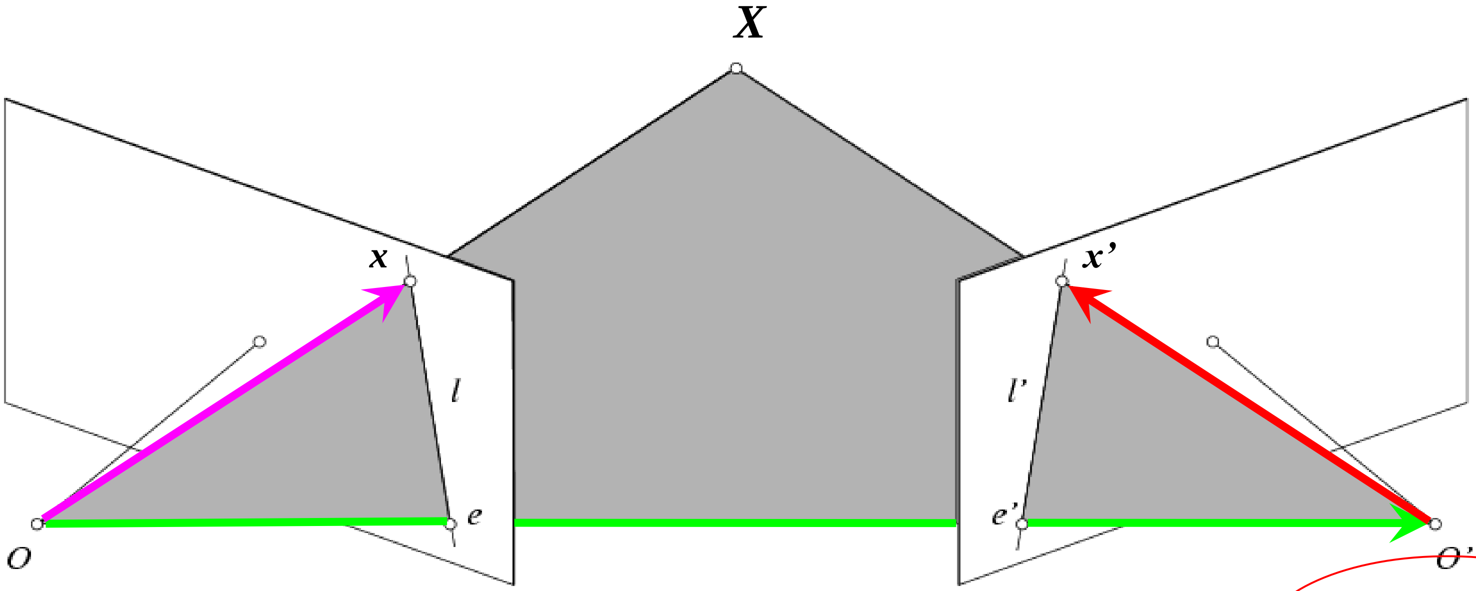
$$\left(\begin{array}{c|c} \mathbf{R} & \mathbf{t} \\ \hline \mathbf{0} & \mathbf{1} \end{array} \right) = (\mathrm{R} \mid \mathrm{T})$$

Epipolar constraint: Calibrated case



The vectors ***Rx***, ***t***, and ***x'*** are coplanar

Epipolar constraint: Calibrated case



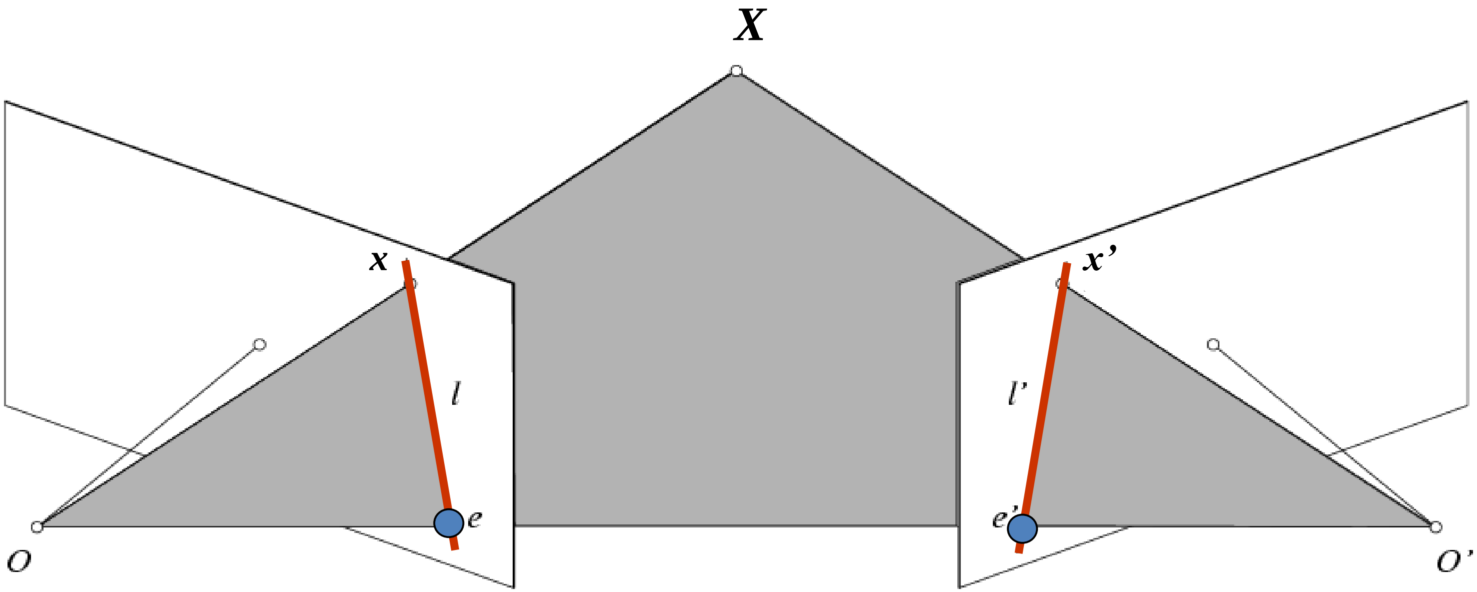
$$\boldsymbol{x}' \cdot [\boldsymbol{t} \times (\boldsymbol{R}\,\boldsymbol{x})] = 0 \quad \Rightarrow \quad \boldsymbol{x}'^{T} \boldsymbol{E}\, \boldsymbol{x} = 0 \quad \text{with} \quad \boldsymbol{E} = [\boldsymbol{t}_{\times}]\boldsymbol{R}$$

Essential Matrix E

(Longuet-Higgins, 1981)

The vectors ***Rx***, ***t***, and ***x'*** are coplanar

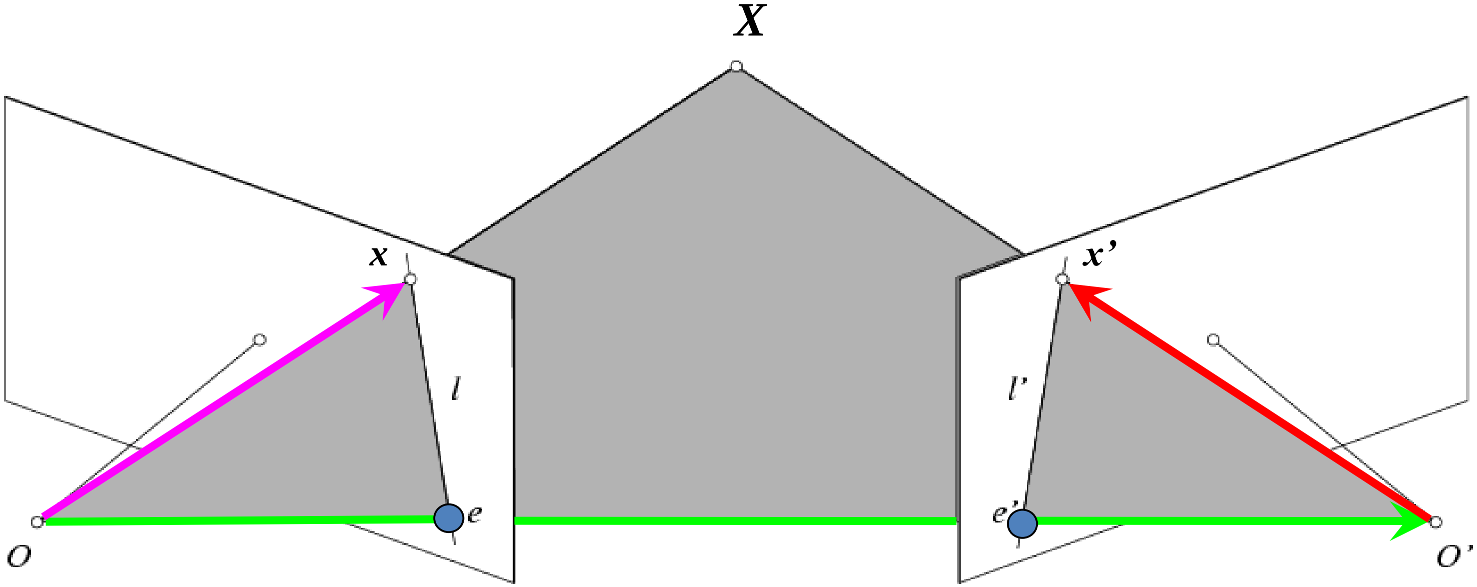
Epipolar constraint: Calibrated case



$$\mathbf{x}' \cdot [\boldsymbol{t} \times (\boldsymbol{R}\mathbf{x})] = 0 \quad \Rightarrow \quad \mathbf{x}'^{T} \boldsymbol{E}\, \mathbf{x} = 0 \quad \text{with} \quad \boldsymbol{E} = [\boldsymbol{t}_{\times}]\boldsymbol{R}$$

- ***E x*** is the epipolar line associated with ***x*** (***l'*** = ***E x***)
- ***E***^{T}***x'*** is the epipolar line associated with ***x'*** (***l*** = ***E***^{T}***x'***)
- ***E e*** = 0   and   ***E***^{T}***e'*** *= 0*
- ***E*** is singular (rank two)
- ***E*** has five degrees of freedom

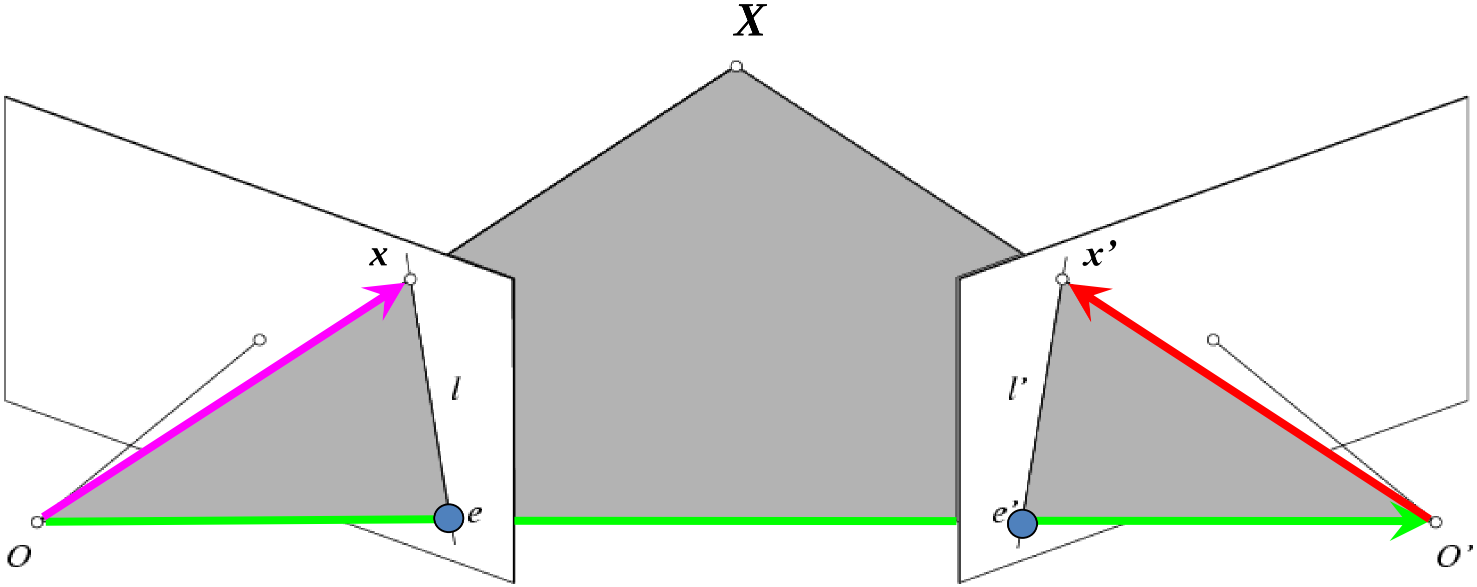
Epipolar constraint: Uncalibrated case



- The calibration matrices ***K*** and ***K'*** of the two cameras are unknown
- We can write the epipolar constraint in terms of *unknown* normalized coordinates:

$$\hat{\mathbf{x}}'^{T} \boldsymbol{E} \hat{\mathbf{x}} = 0 \qquad \hat{\mathbf{x}} = \boldsymbol{K}^{-1}\mathbf{x}, \quad \hat{\mathbf{x}}' = \boldsymbol{K}'^{-1}\hat{\mathbf{x}}'$$

Epipolar constraint: Uncalibrated case



$$\hat{\mathbf{x}}'^{T} \boldsymbol{E}\, \hat{\mathbf{x}} = 0 \quad \Rightarrow \quad \mathbf{x}'^{T} \boldsymbol{F}\, \mathbf{x} = 0 \quad \text{with} \quad \boldsymbol{F} = \boldsymbol{K}'^{-T} \boldsymbol{E}\, \boldsymbol{K}^{-1}$$

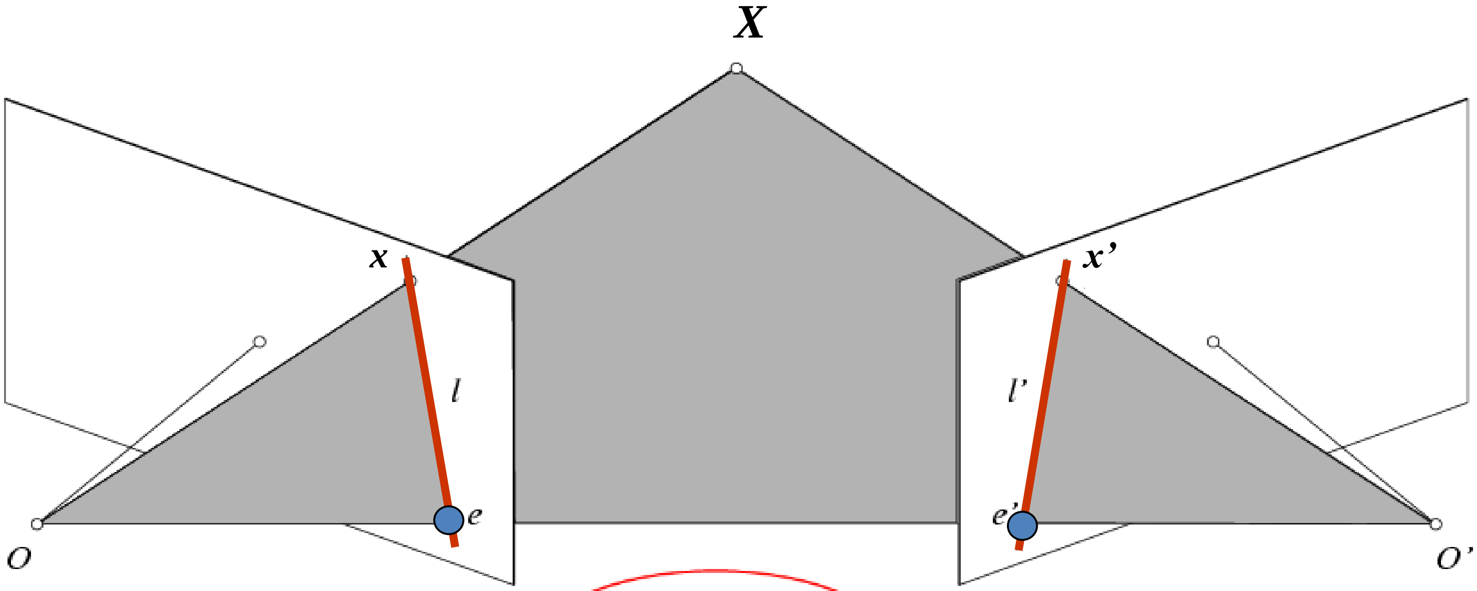
$$\hat{\mathbf{x}} = \boldsymbol{K}^{-1} \mathbf{x}$$

$$\hat{\mathbf{x}}' = \boldsymbol{K}'^{-1} \mathbf{x}'$$

Fundamental Matrix

(Faugeras and Luong, 1992)

Epipolar constraint: Uncalibrated case



$$\hat{\mathbf{x}}'^{T} \boldsymbol{E}\, \hat{\mathbf{x}} = 0 \quad \Rightarrow \quad \mathbf{x}'^{T} \boldsymbol{F}\, \mathbf{x} = 0 \quad \text{with} \quad \boldsymbol{F} = \boldsymbol{K}'^{-T} \boldsymbol{E}\, \boldsymbol{K}^{-1}$$

- $\boldsymbol{F}\,\boldsymbol{x}$ is the epipolar line associated with $\boldsymbol{x}$ ($\boldsymbol{l}' = \boldsymbol{F}\,\boldsymbol{x}$)
- $\boldsymbol{F}^{T}\boldsymbol{x}'$ is the epipolar line associated with $\boldsymbol{x}'$ ($\boldsymbol{l}' = \boldsymbol{F}^{T}\boldsymbol{x}'$)
- $\boldsymbol{F}\,\boldsymbol{e} = 0$ and $\boldsymbol{F}^{T}\boldsymbol{e}' = 0$

The eight-point algorithm

$$\boldsymbol{x} = (u, v, 1)^T, \quad \boldsymbol{x}' = (u', v', 1)$$

$$\begin{bmatrix} u' & v' & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = 0 \Rightarrow \underset{\text{A}}{\begin{bmatrix} u'u & u'v & u' & v'u & v'v & v' & u & v & 1 \end{bmatrix}} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

Smallest eigenvalue of A^TA

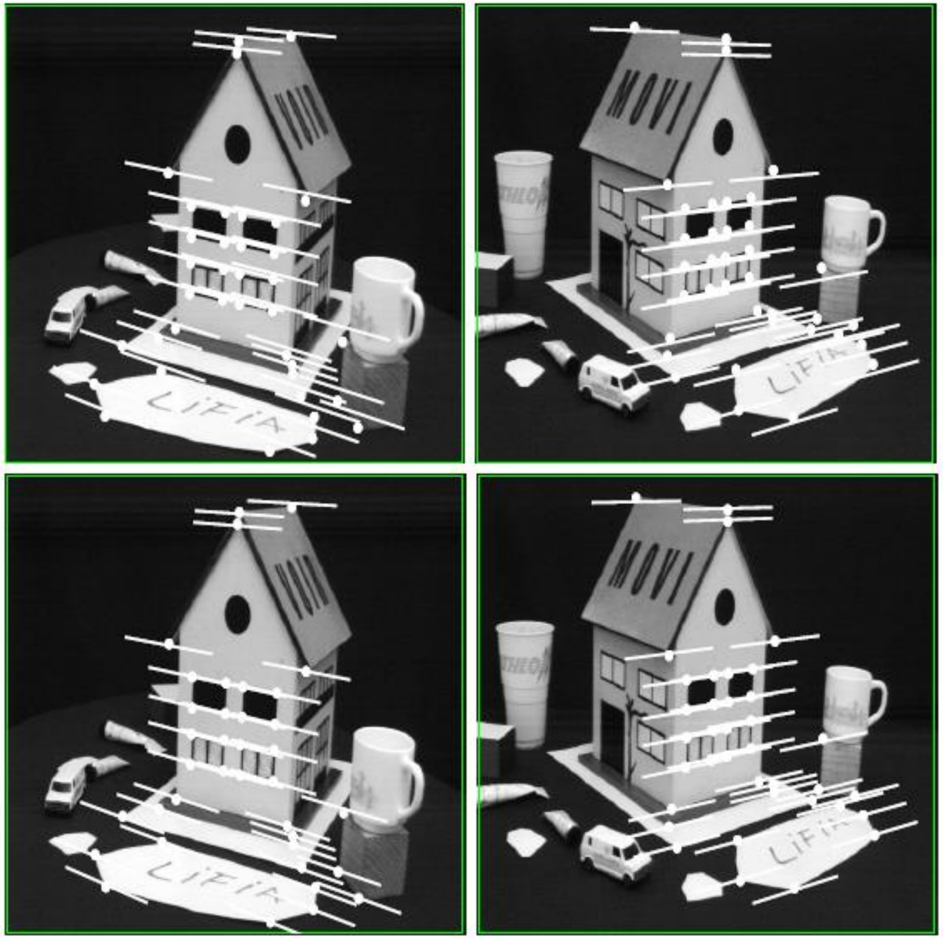
Minimize:

$$\sum_{i=1}^{N} (\boldsymbol{x}_i'^T \boldsymbol{F} \, \boldsymbol{x}_i)^2$$

under the constraint

$||\boldsymbol{F}||^2 = 1$

Comparison of estimation



	8-point	Normalized 8-point	Nonlinear least squares
Av. Dist. 1	2.33 pixels	0.92 pixel	0.86 pixel
Av. Dist. 2	2.18 pixels	0.85 pixel	0.80 pixel

Moving on to stereo…

Fuse a calibrated binocular stereo pair to produce a depth image

image 1

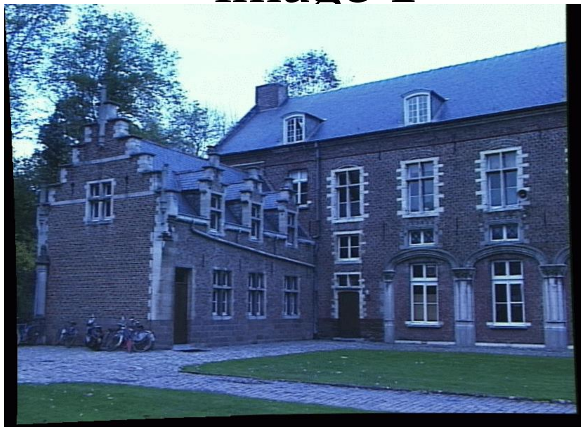
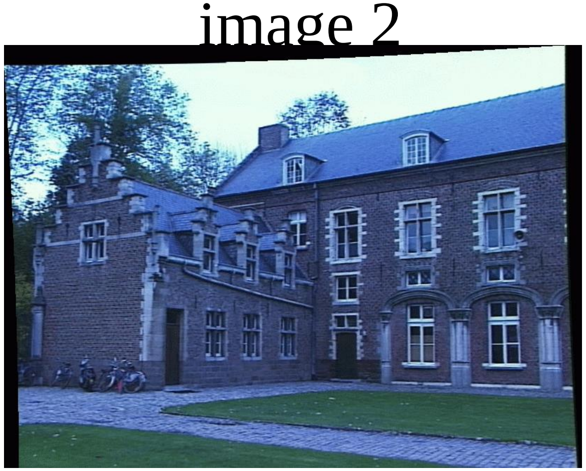
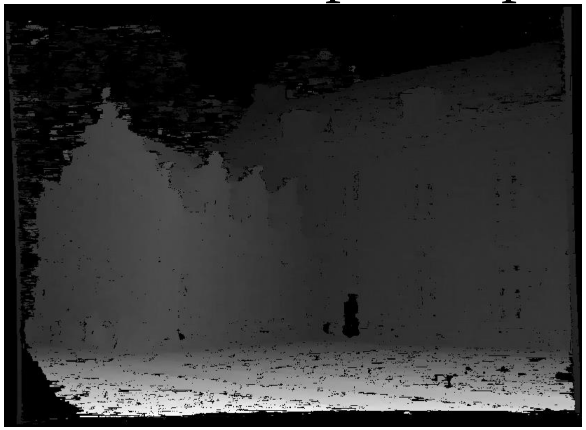


image 2



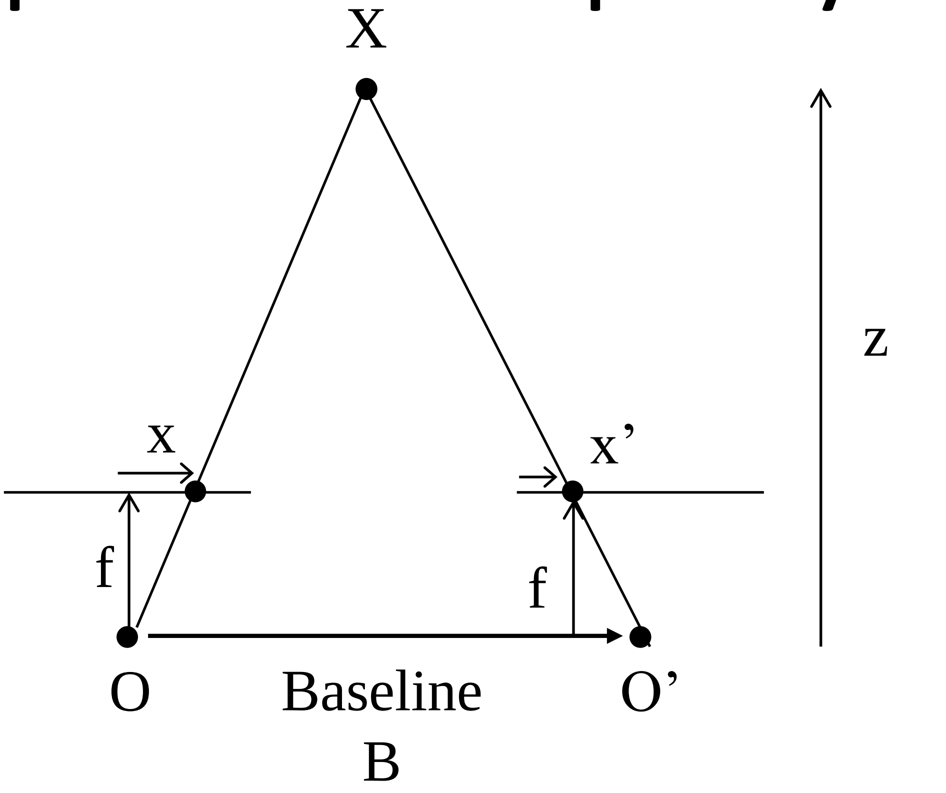
Dense depth map



Many of these slides adapted from Steve Seitz and Lana Lazebnik

Depth from disparity

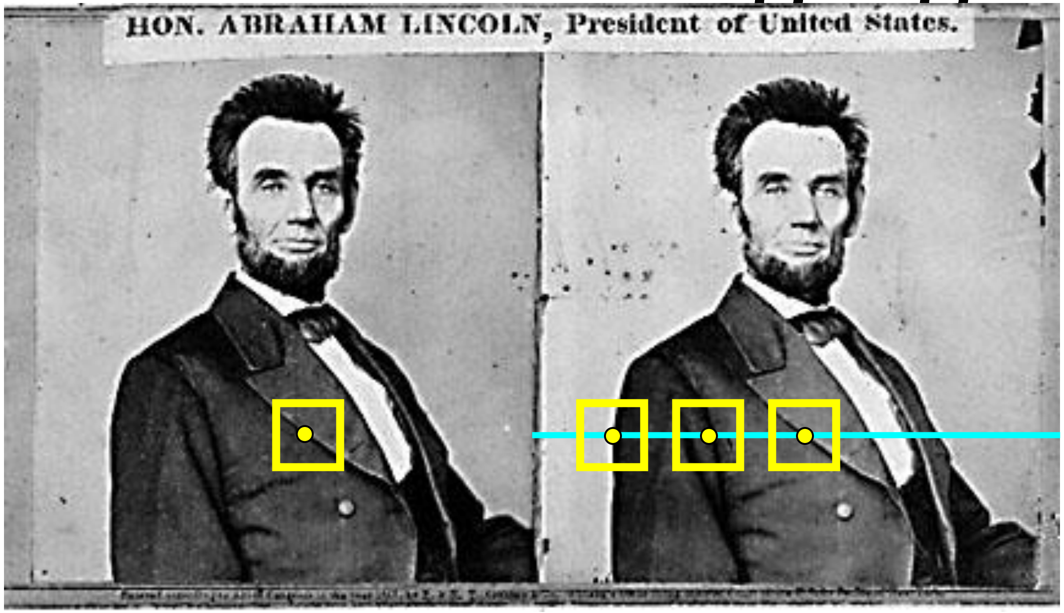
$$\frac{x - x'}{O - O'} = \frac{f}{z}$$



$$\boxed{disparity = x - x' = \frac{B \cdot f}{z}}$$

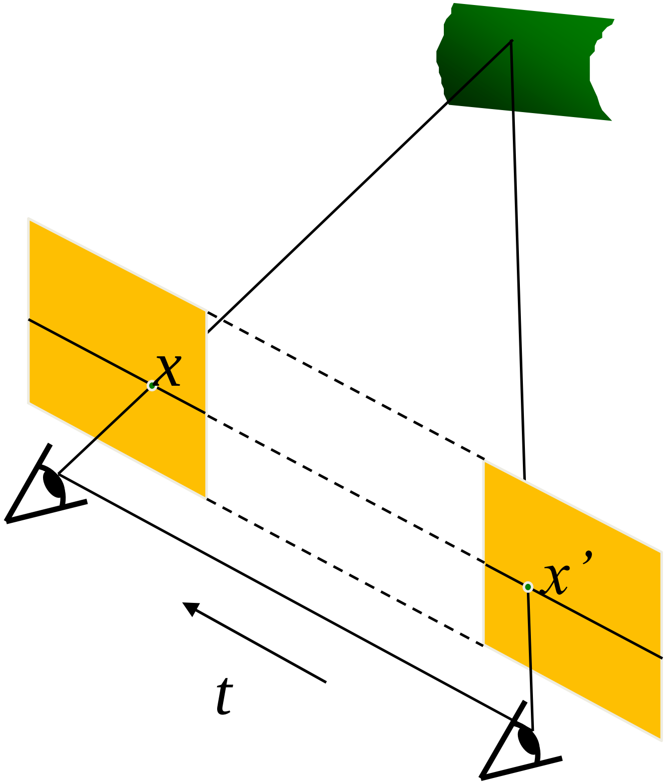
Disparity is inversely proportional to depth.

Basic stereo matching algorithm



- If necessary, rectify the two stereo images to transform epipolar lines into scanlines
- For each pixel x in the first image
  - Find corresponding epipolar scanline in the right image
  - Search the scanline and pick the best match x’
  - Compute disparity x-x’ and set depth(x) = fB/(x-x’)

Simplest Case: Parallel images



Epipolar constraint:

$$x^T E x' = 0, \quad E = t \times R$$

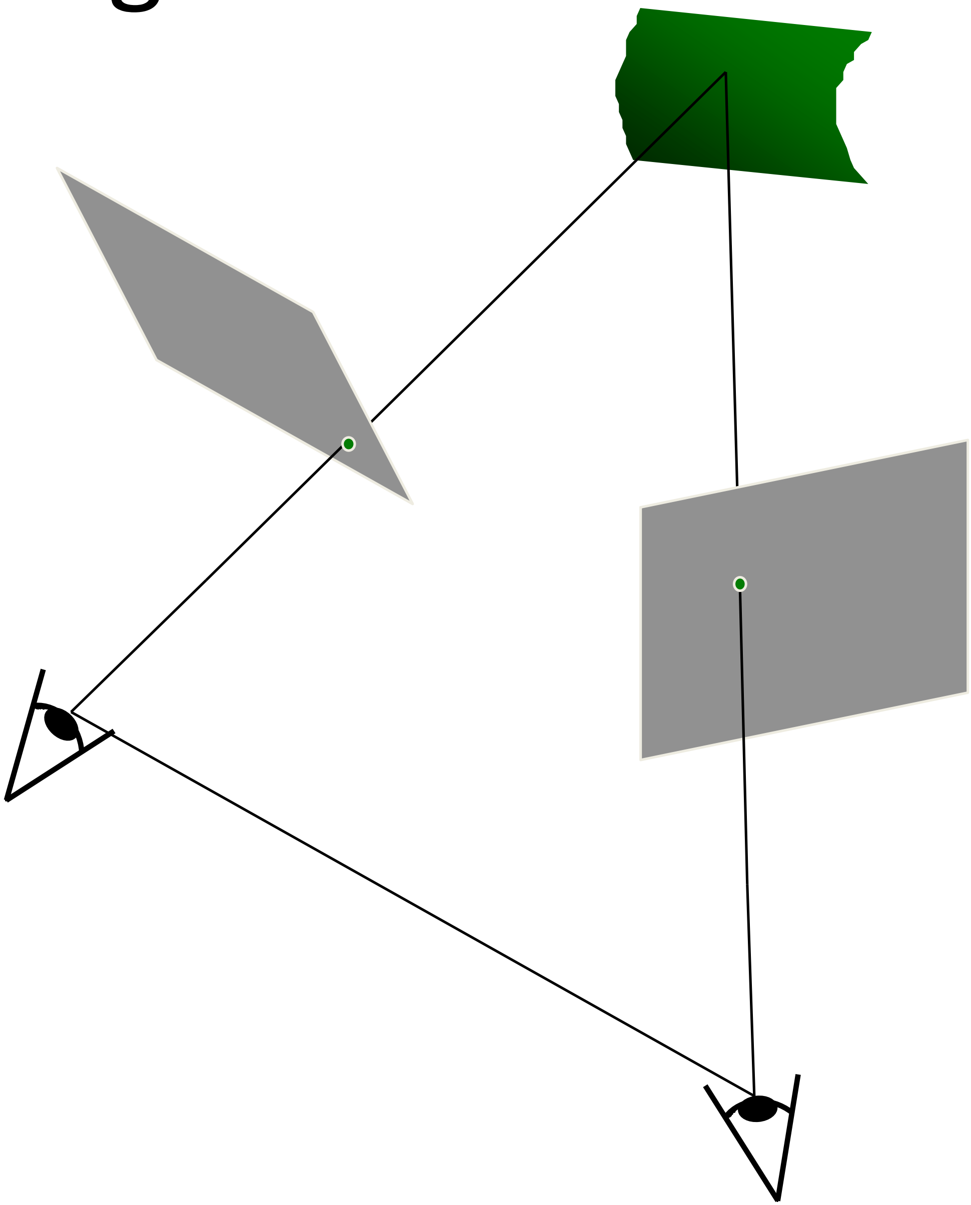
$$R = I \qquad t = (T, 0, 0)$$

$$E = t \times R = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T \\ 0 & T & 0 \end{bmatrix}$$

$$\begin{pmatrix} u & v & 1 \end{pmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T \\ 0 & T & 0 \end{bmatrix} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0$$

The y-coordinates of corresponding points are the same

Stereo image rectification

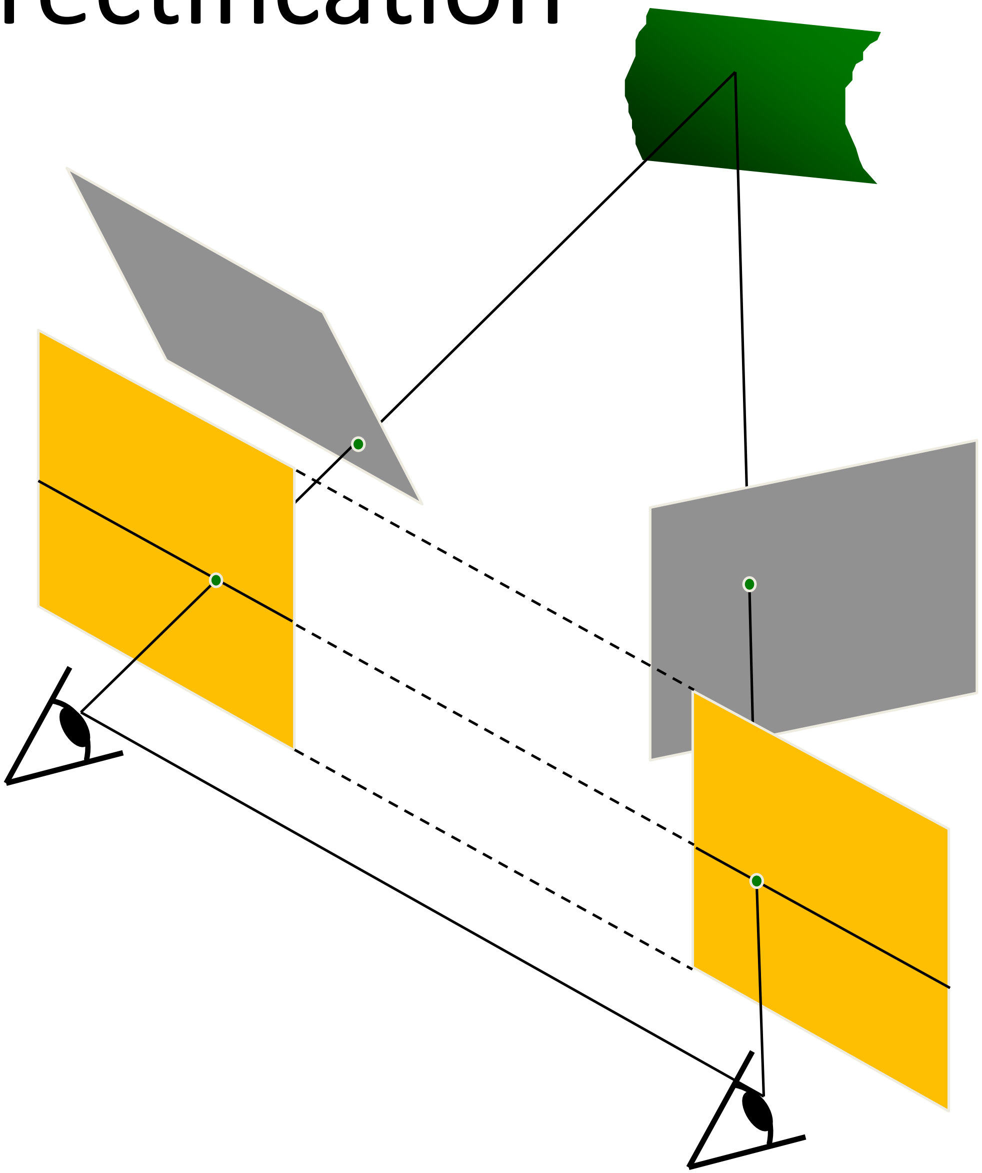


Stereo image rectification

- Reproject image planes onto a common plane parallel to the line between camera centers

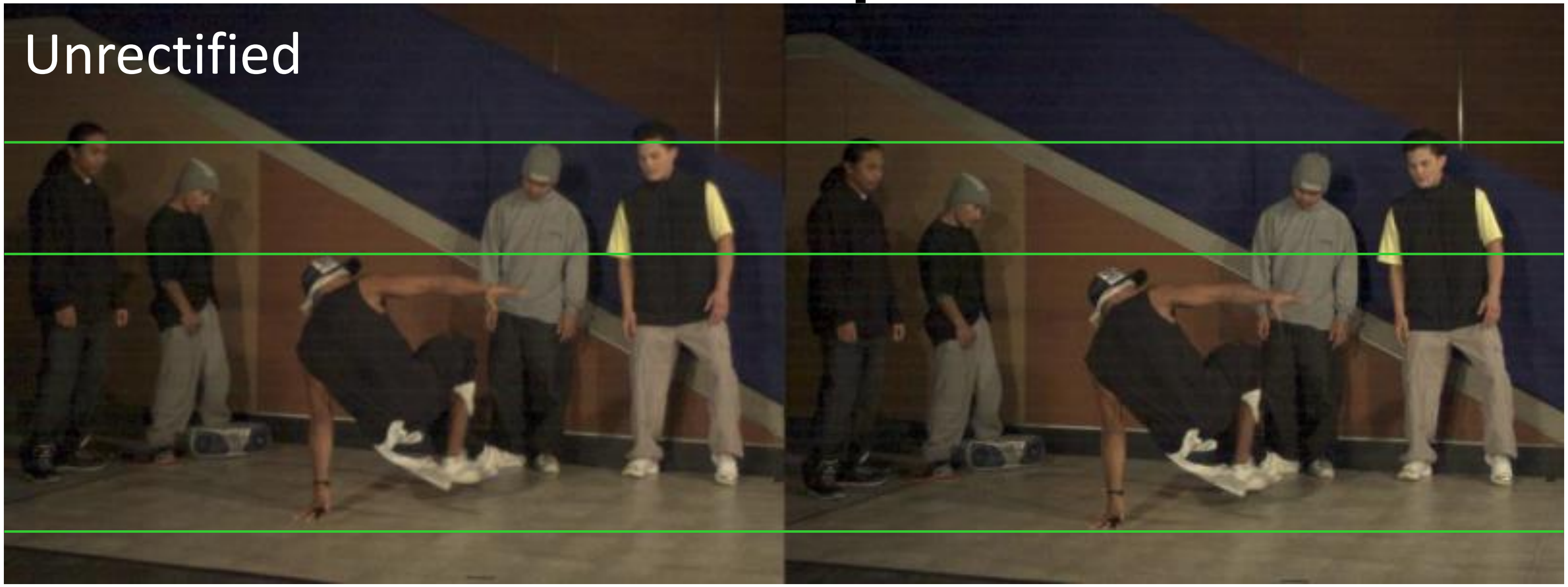
- Pixel motion is horizontal after this transformation

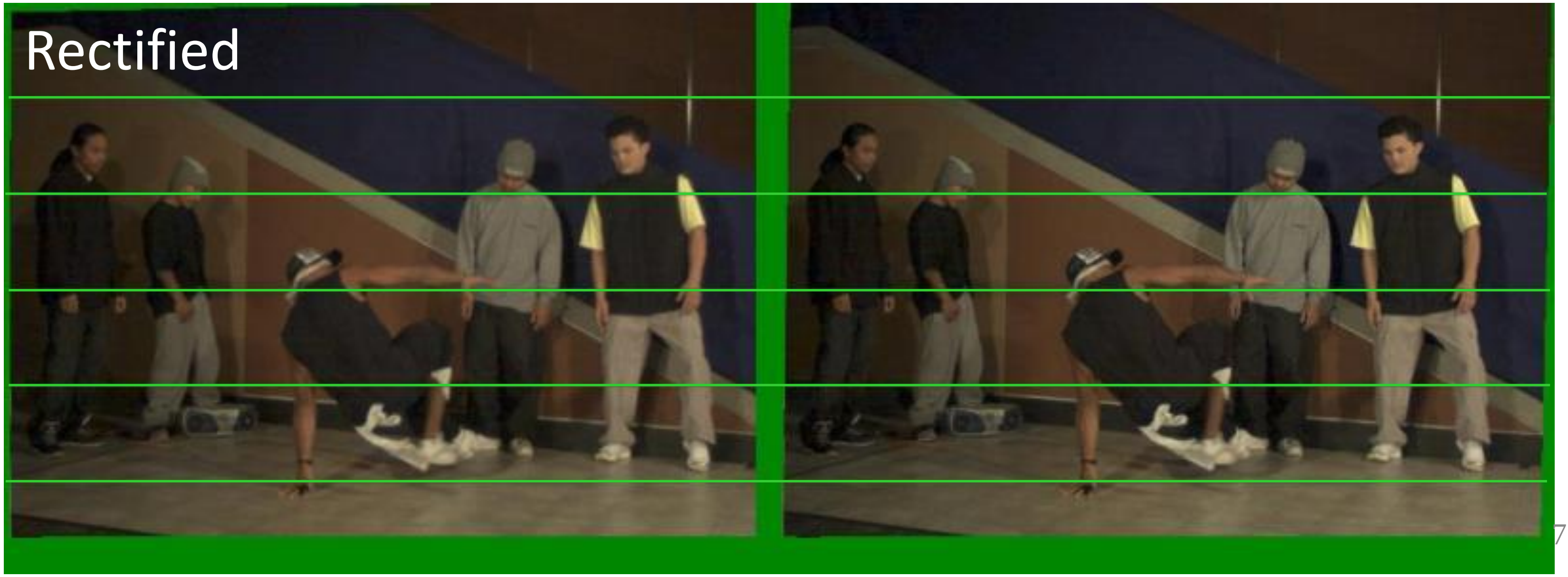
- Two homographies (3x3 transform), one for each input image reprojection

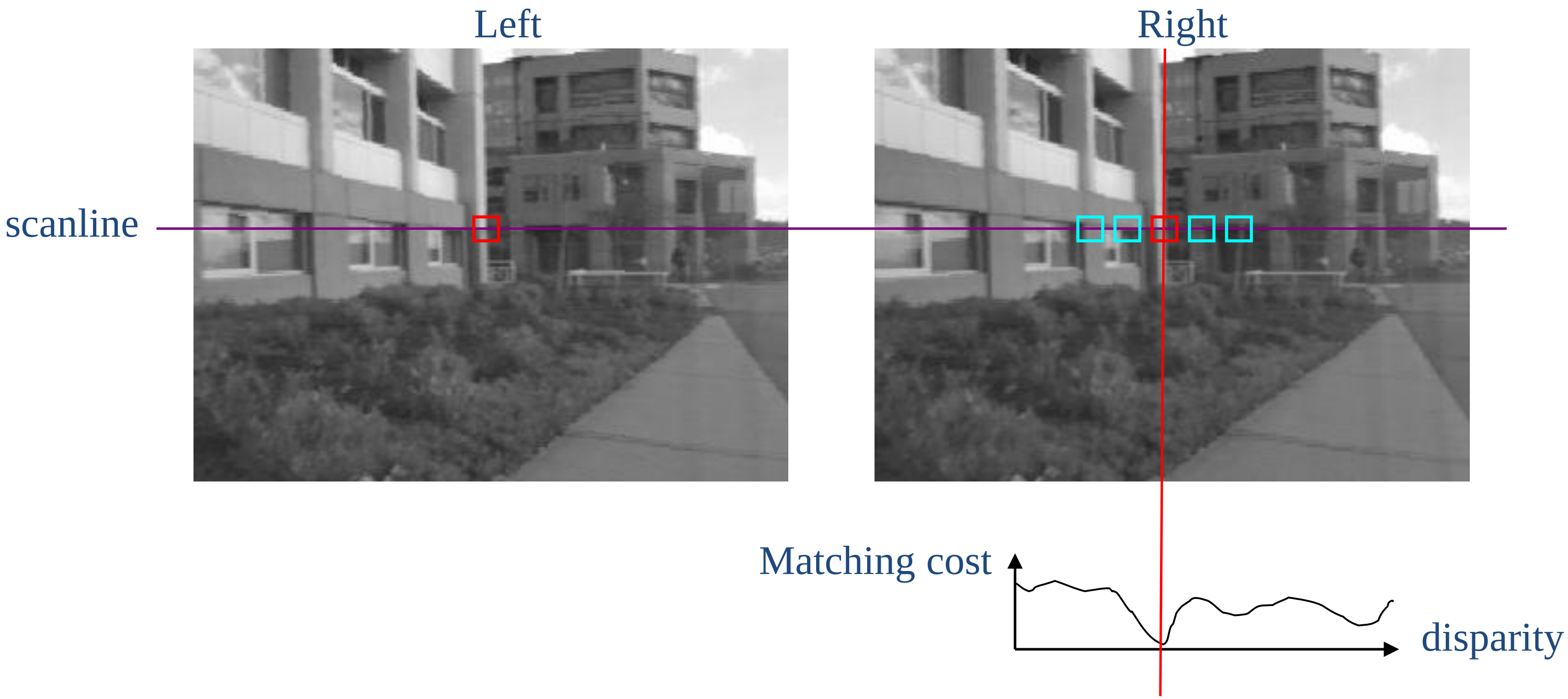


➢ C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision. IEEE Conf. Computer Vision and Pattern Recognition, 1999.

Example

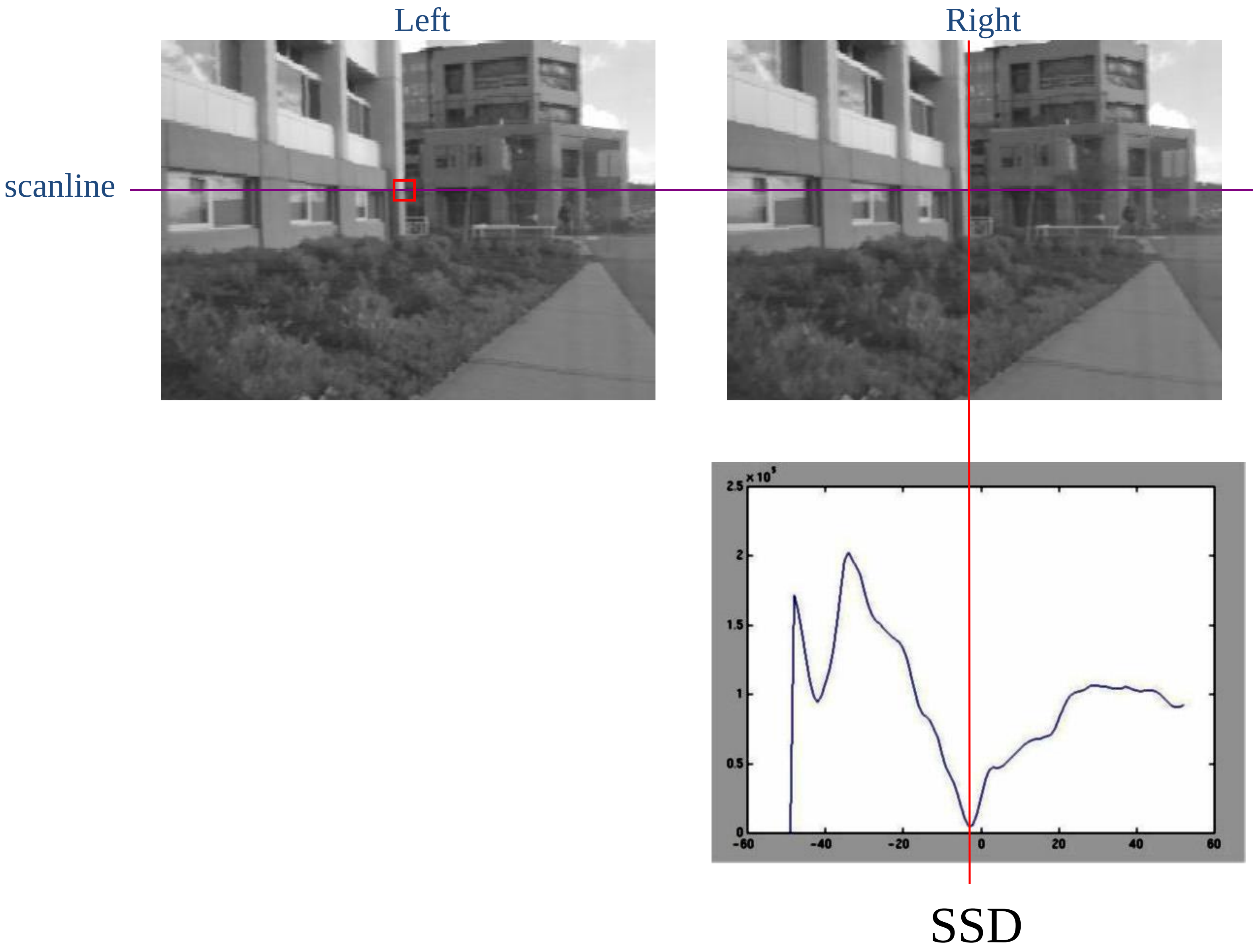






- Slide a window along the right scanline and compare contents of that window with the reference window in the left image
- Matching cost: SSD, SAD, or normalized correlation

Correspondence search



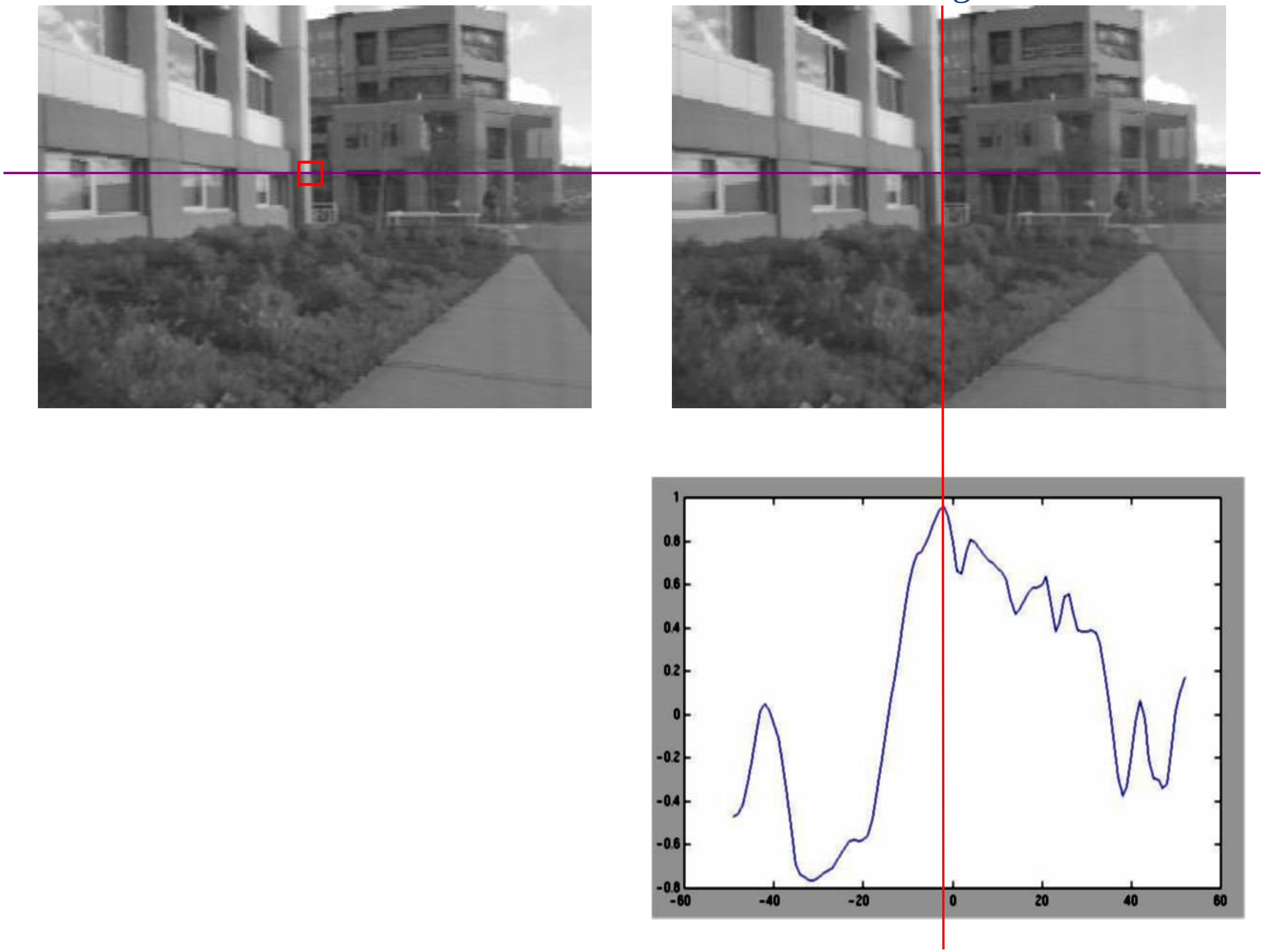
Left

Right

scanline

SSD

Correspondence search



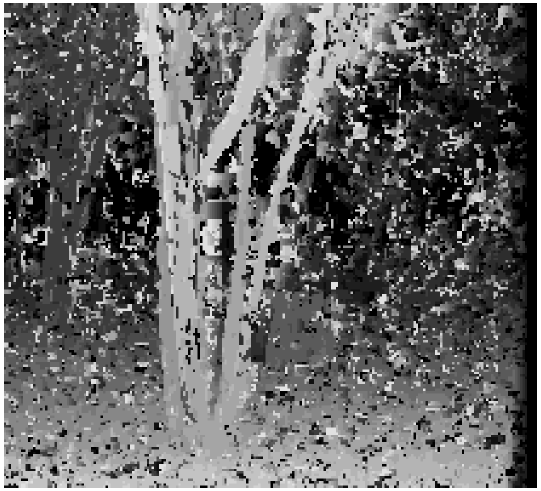
Left

Right

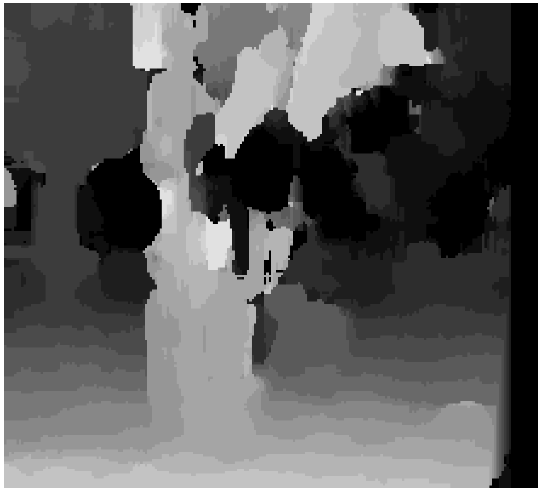
scanline

Norm. corr

Effect of window size



W = 3

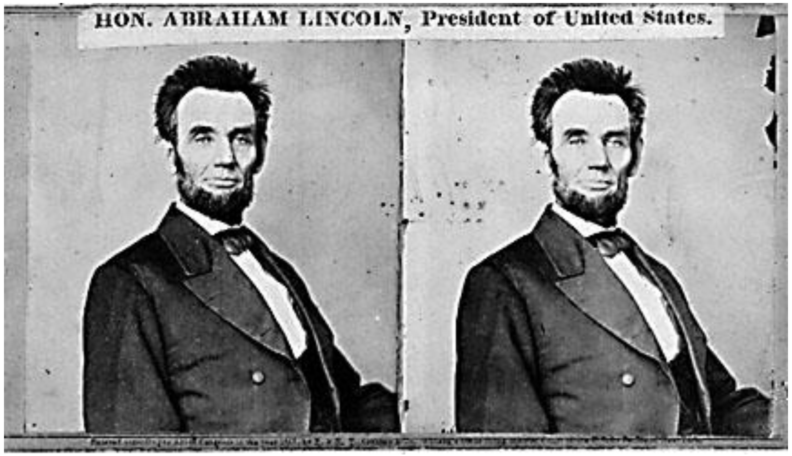


W = 20

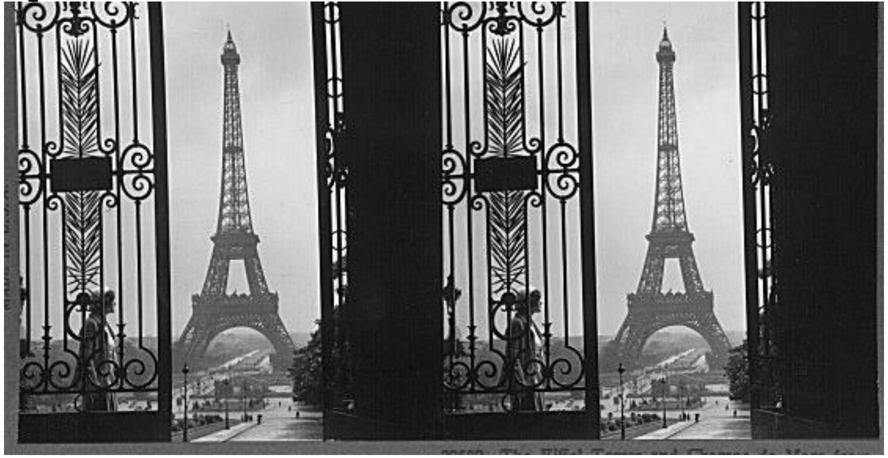
- Smaller window
  - + More detail
  - – More noise

- Larger window
  - + Smoother disparity maps
  - – Less detail
  - – Fails near boundaries

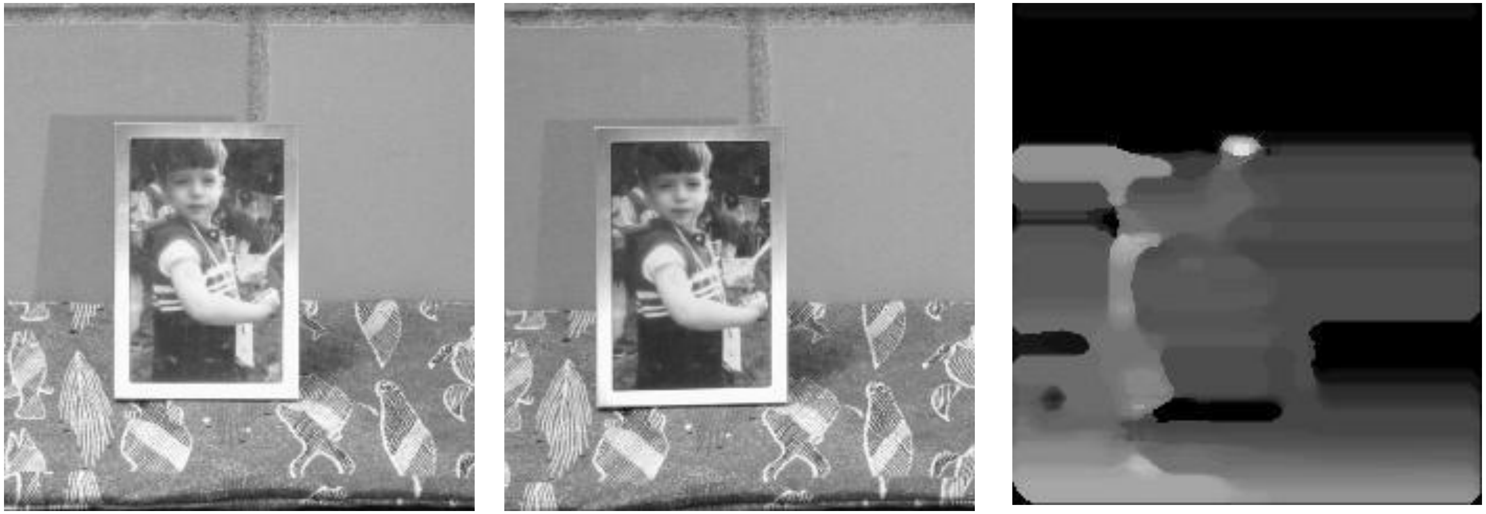
Failures of correspondence search



Textureless surfaces



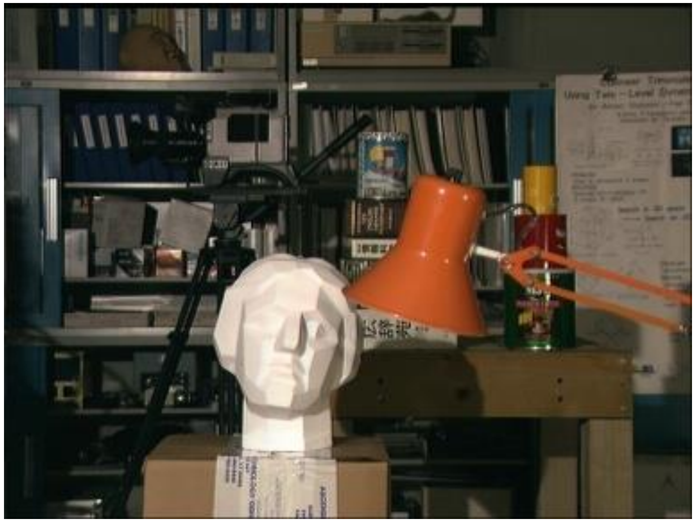
Occlusions, repetition



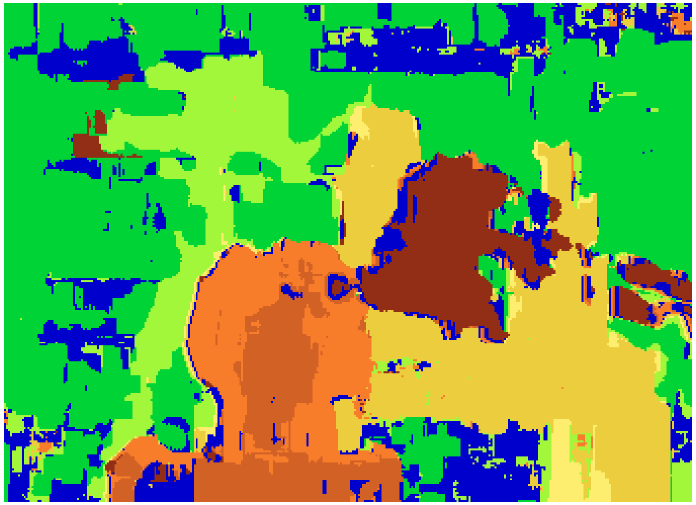
Non-Lambertian surfaces, specularities

Results with window search

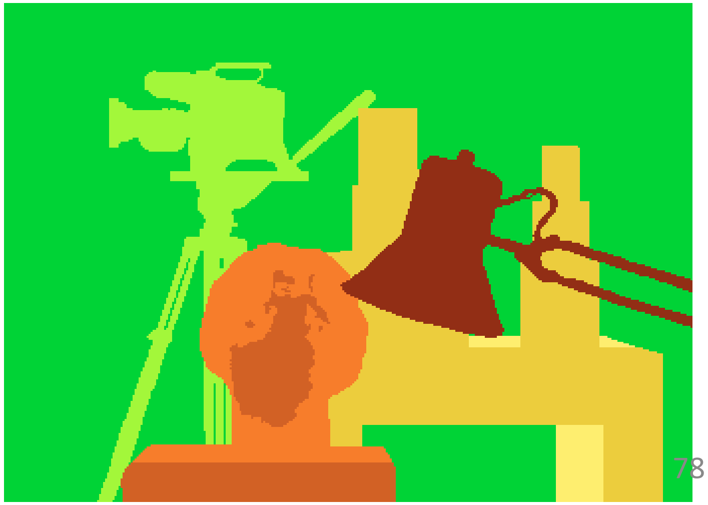
Data



Window-based matching



Ground truth



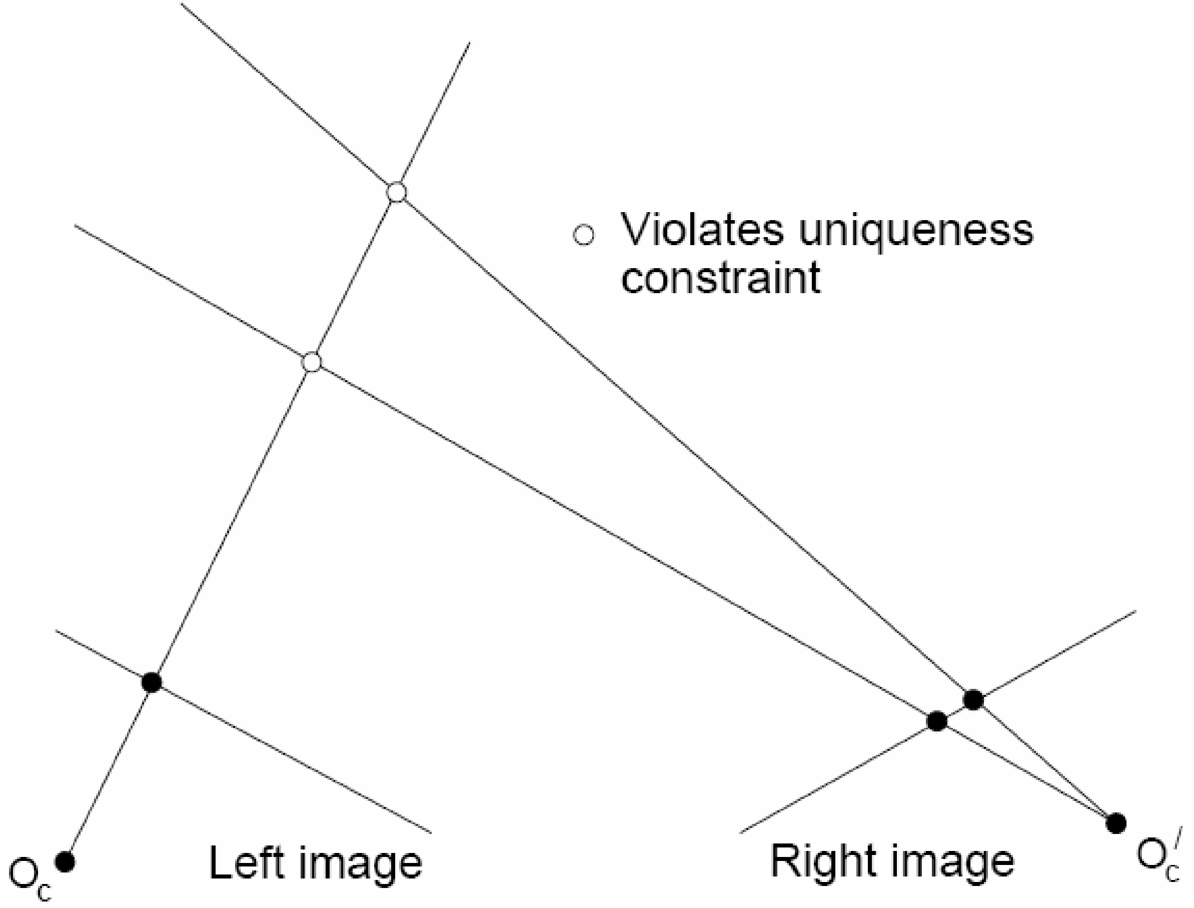
How can we improve window-based matching?

- So far, matches are independent for each point

- What constraints or priors can we add?

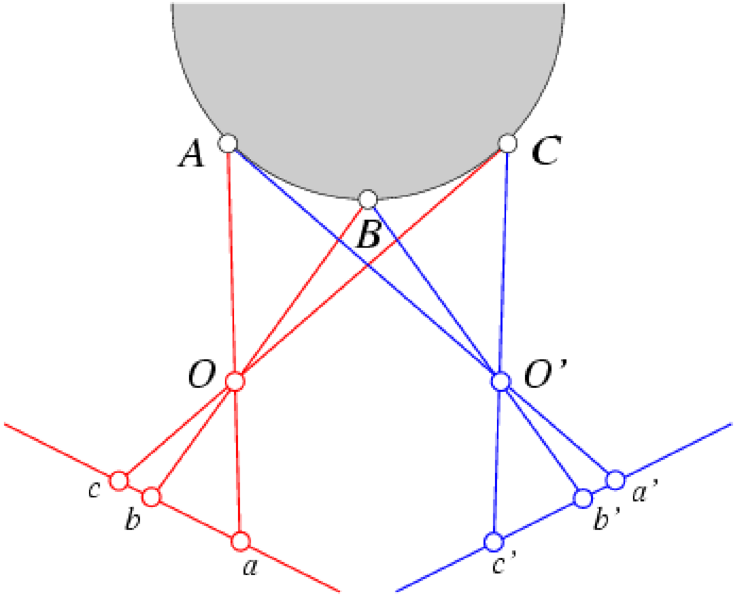
Stereo constraints/priors

- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image



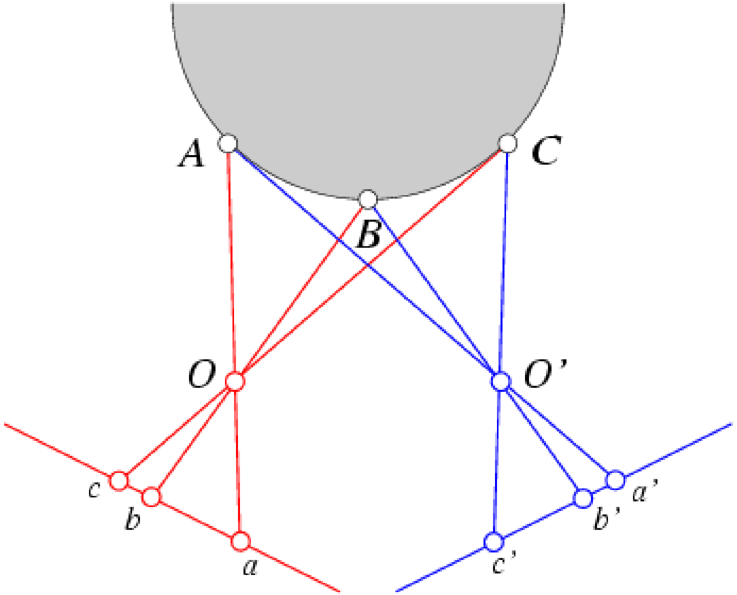
Stereo constraints/priors

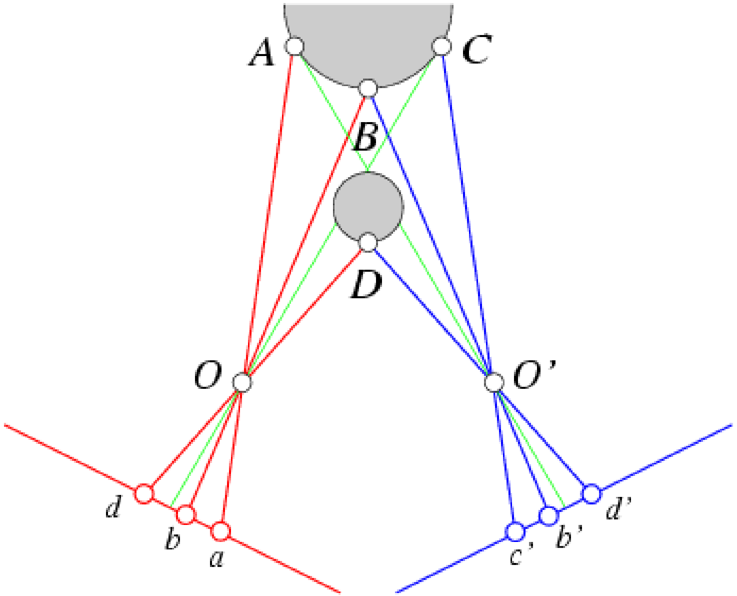
- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image
- Ordering
  - Corresponding points should be in the same order in both views



Stereo constraints/priors

- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image
- Ordering
  - Corresponding points should be in the same order in both views



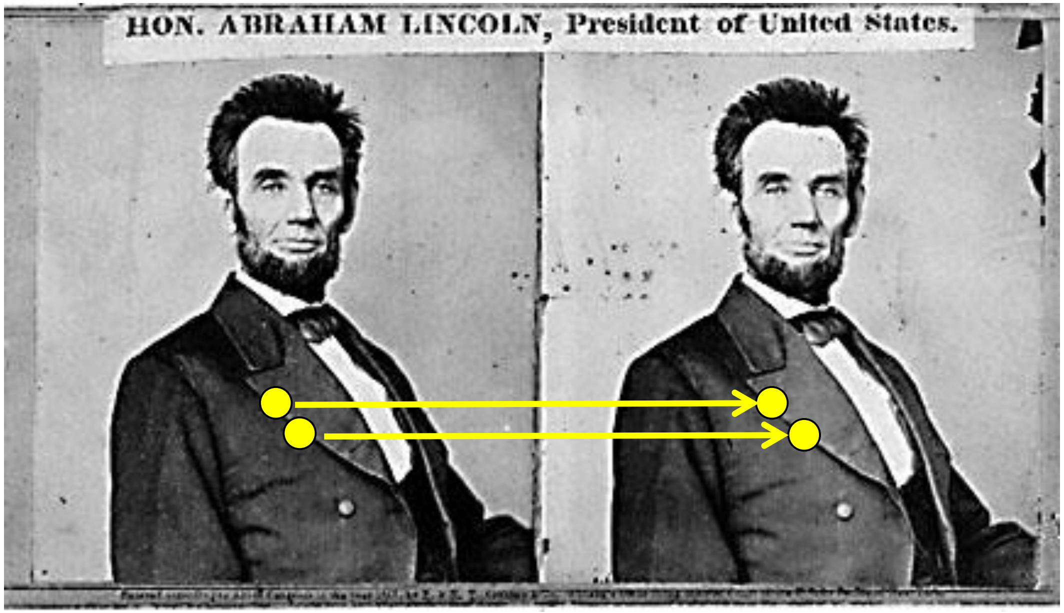


Ordering constraint doesn't hold

Priors and constraints

- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image
- Ordering
  - Corresponding points should be in the same order in both views
- Smoothness
  - We expect disparity values to change slowly (for the most part)

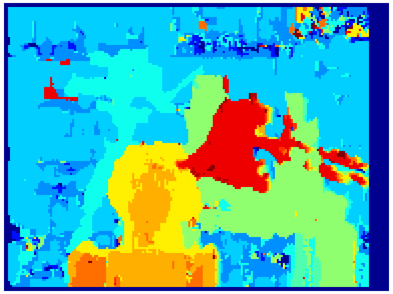
Stereo as energy minimization

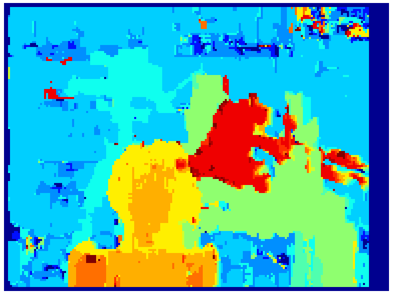


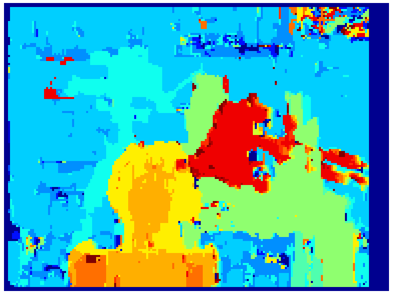
- What defines a good stereo correspondence?
  1. Match quality
     - Want each pixel to find a good match in the other image
  2. Smoothness

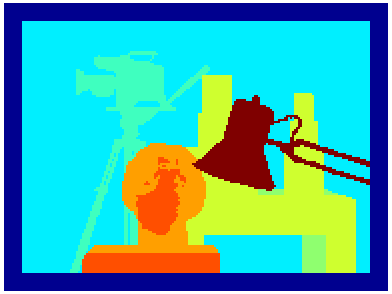
Matching windows:

Similarity Measure	Formula
Sum of Absolute Differences (SAD)	$\sum_{(i,j)\in W} \lvert I_1(i,j) - I_2(x+i,y+j)\rvert$
Sum of Squared Differences (SSD)	$\sum_{(i,j)\in W} \left(I_1(i,j) - I_2(x+i,y+j)\right)^2$
Zero-mean SAD	$\sum_{(i,j)\in W} \lvert I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i,y+j) + \bar{I}_2(x+i,y+j)\rvert$
Locally scaled SAD	$\sum_{(i,j)\in W} \lvert I_1(i,j) - \frac{\bar{I}_1(i,j)}{\bar{I}_2(x+i,y+j)} I_2(x+i,y+j)\rvert$
Normalized Cross Correlation (NCC)	$\frac{\sum_{(i,j)\in W} I_1(i,j).I_2(x+i,y+j)}{\sqrt[2]{\sum_{(i,j)\in W} I_1^2(i,j).\sum_{(i,j)\in W} I_2^2(x+i,y+j)}}$


SAD


SSD


NCC


Ground truth

http://siddhantahuja.wordpress.com/category/stereo-vision/

Real-time stereo

Nomad robot searches for meteorites in Antartica

http://www.frc.ri.cmu.edu/projects/meteorobot/index.html

- Used for robot navigation (and other tasks)
  - Several software-based real-time stereo

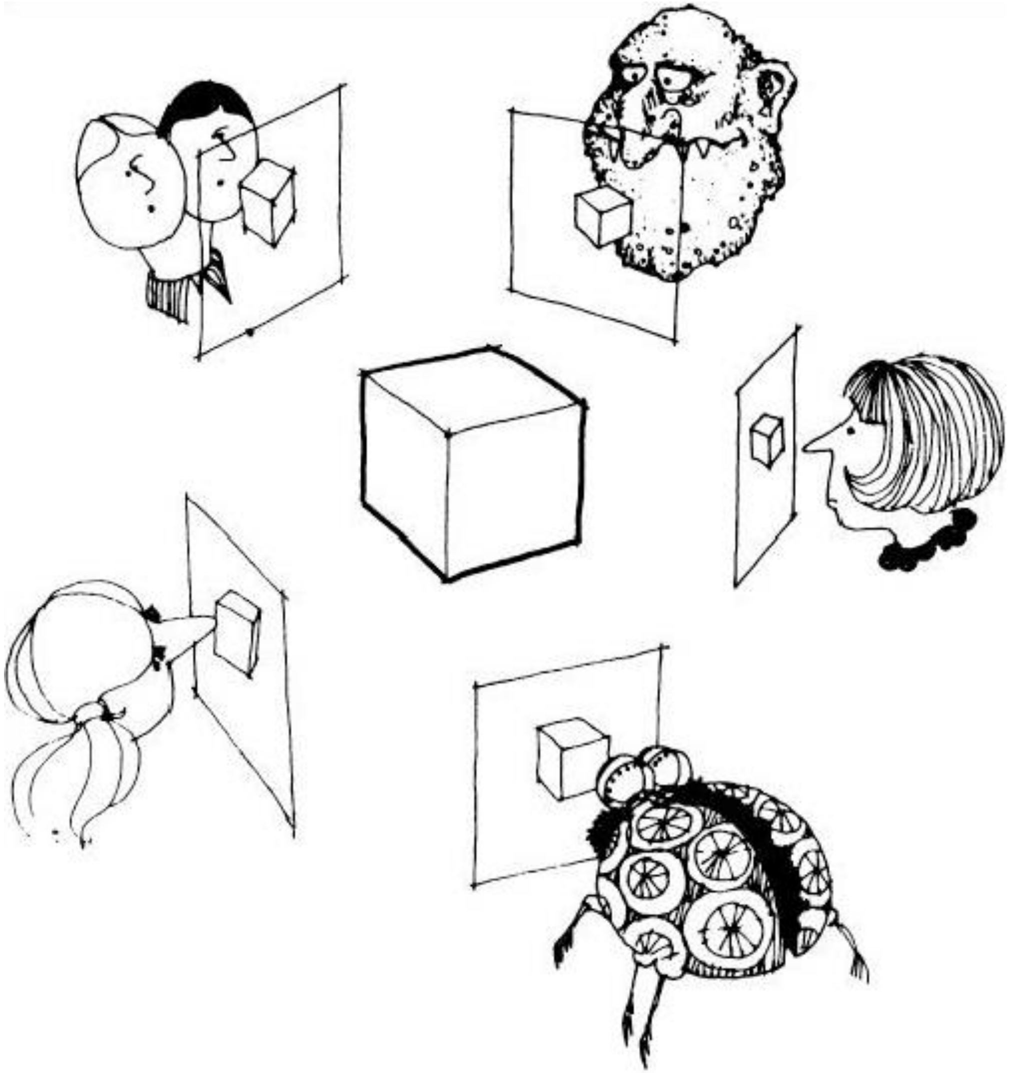
Stereo reconstruction pipeline

- Steps
  - – Calibrate cameras
  - – Rectify images
  - – Compute disparity
  - – Estimate depth

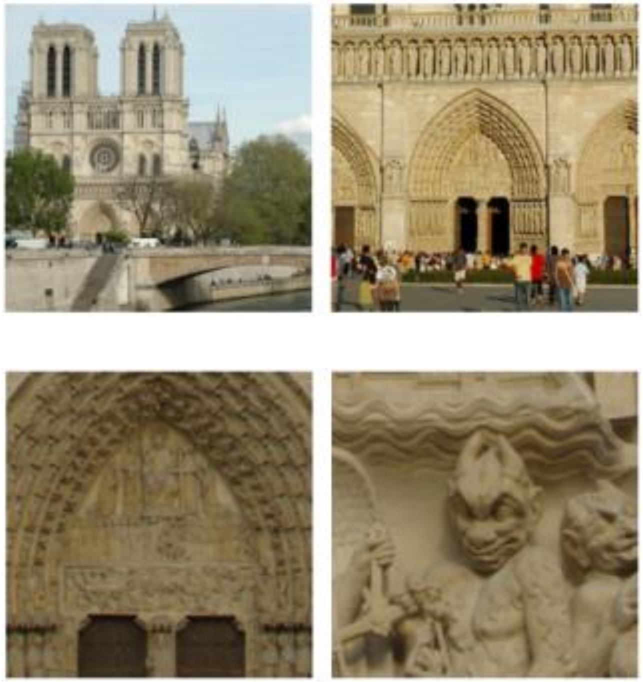
What will cause errors?

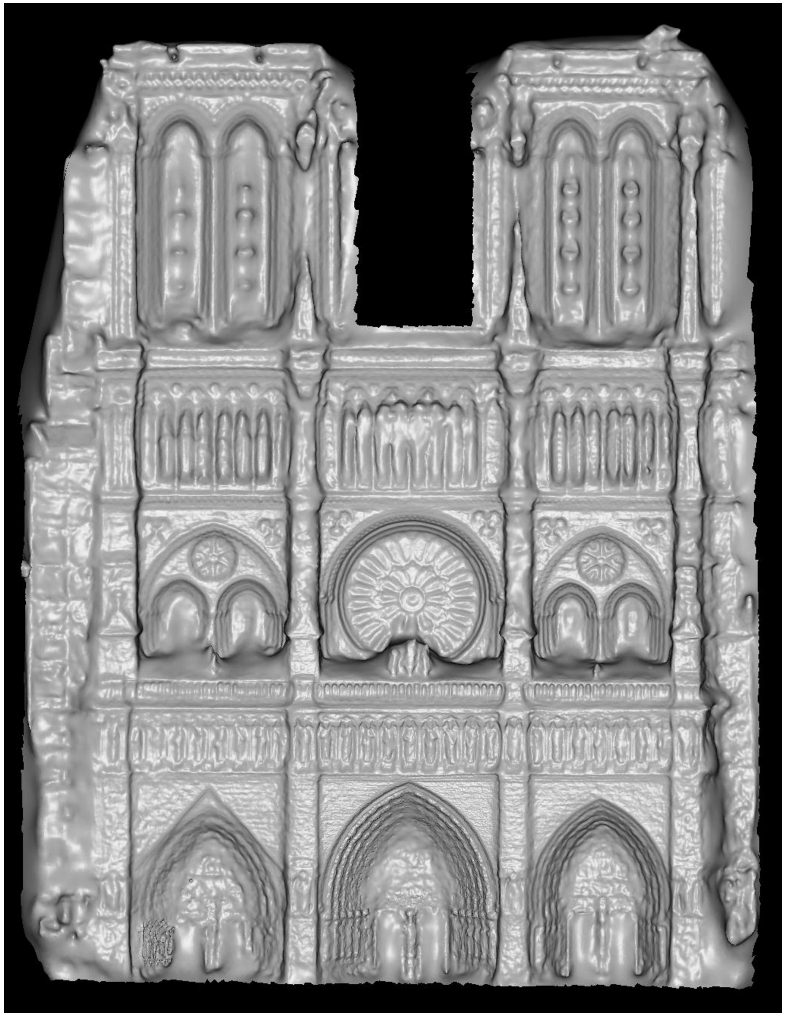
- Camera calibration errors
- Poor image resolution
- Occlusions
- Violations of brightness constancy (specular reflections)
- Large motions
- Low-contrast image regions

Multi-view stereo ?



Using more than two images





Multi-View Stereo for Community Photo Collections
M. Goesele, N. Snavely, B. Curless, H. Hoppe, S. Seitz
Proceedings of ICCV 2007,