

# Stereo

CSE 455

Linda Shapiro

**WHY DO WE HAVE  
TWO EYES?**



# Why do we perceive depth?



# What do humans use as depth cues?

---

## Motion

### **Convergence**

When watching an object close to us, our eyes point slightly inward. This difference in the direction of the eyes is called convergence. This depth cue is effective only on short distances (less than 10 meters).

### **Binocular Parallax**

As our eyes see the world from slightly different locations, the images sensed by the eyes are slightly different. This difference in the sensed images is called binocular parallax. Human visual system is very sensitive to these differences, and binocular parallax is the most important depth cue for medium viewing distances. The sense of depth can be achieved using binocular parallax even if all other depth cues are removed.

### **Monocular Movement Parallax**

If we close one of our eyes, we can perceive depth by moving our head. This happens because human visual system can extract depth information in two similar images sensed after each other, in the same way it can combine two images from different eyes.

## Focus

### **Accommodation**

Accommodation is the tension of the muscle that changes the focal length of the lens of eye. Thus it brings into focus objects at different distances. This depth cue is quite weak, and it is effective only at short viewing distances (less than 2 meters) and with other cues.

# What do humans use as depth cues?

---

## Image cues

### **Retinal Image Size**

When the real size of the object is known, our brain compares the sensed size of the object to this real size, and thus acquires information about the distance of the object.

### **Linear Perspective**

When looking down a straight level road we see the parallel sides of the road meet in the horizon. This effect is often visible in photos and it is an important depth cue. It is called linear perspective.

### **Texture Gradient**

The closer we are to an object the more detail we can see of its surface texture. So objects with smooth textures are usually interpreted being farther away. This is especially true if the surface texture spans all the distance from near to far.

### **Overlapping**

When objects block each other out of our sight, we know that the object that blocks the other one is closer to us. The object whose outline pattern looks more continuous is felt to lie closer.

### **Aerial Haze**

The mountains in the horizon look always slightly bluish or hazy. The reason for this are small water and dust particles in the air between the eye and the mountains. The farther the mountains, the hazier they look.

### **Shades and Shadows**

When we know the location of a light source and see objects casting shadows on other objects, we learn that the object shadowing the other is closer to the light source. As most illumination comes downward we tend to resolve ambiguities using this information. The three dimensional looking computer user interfaces are a nice example on this. Also, bright objects seem to be closer to the observer than dark ones.



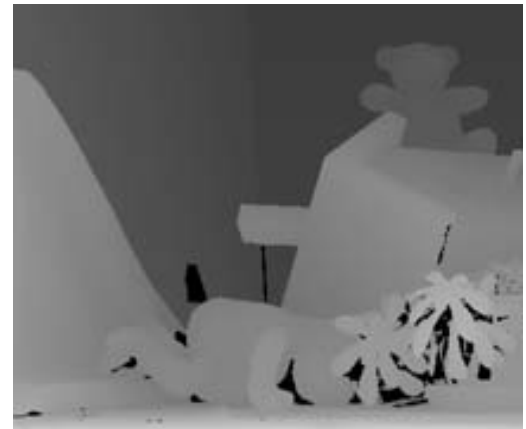




# Amount of horizontal movement is ...

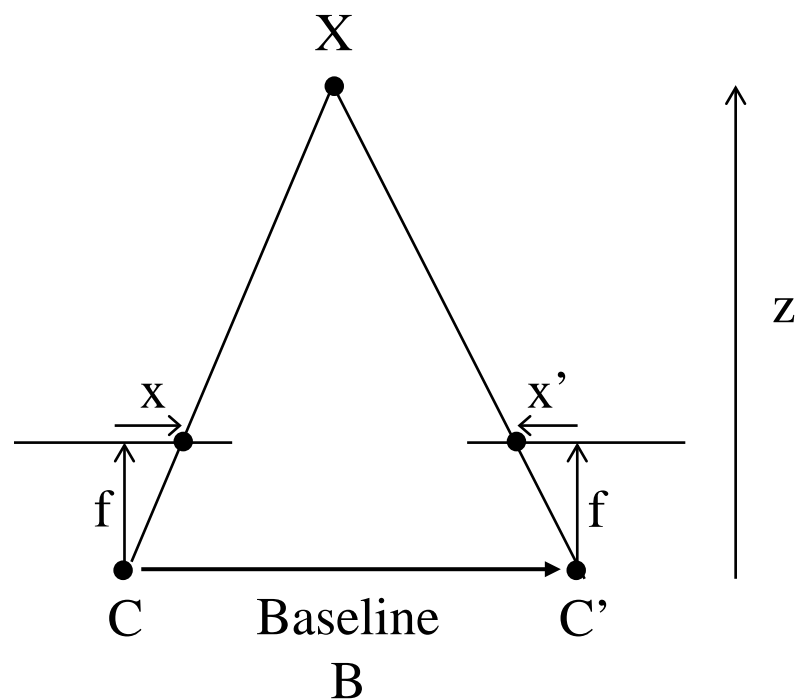
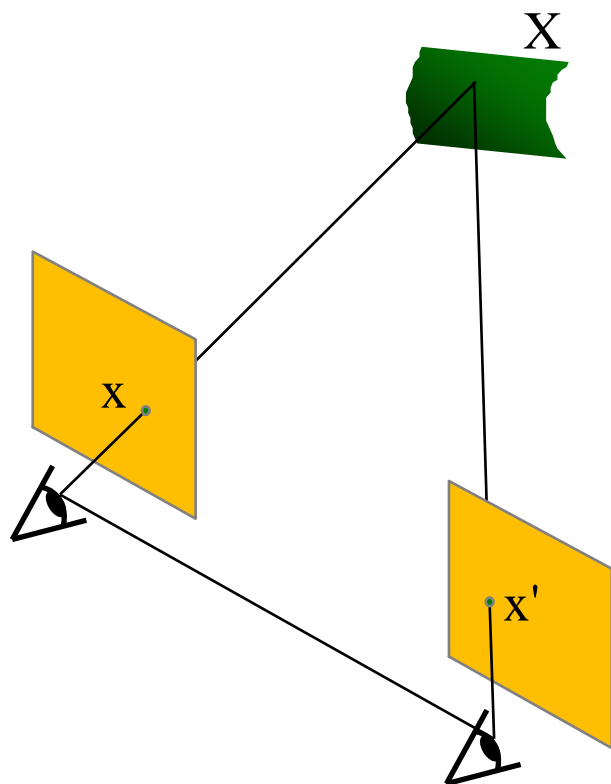
---

...inversely proportional to the distance from the camera



# Depth from Stereo

Goal: recover depth by finding image coordinate  $x'$  that corresponds to  $x$

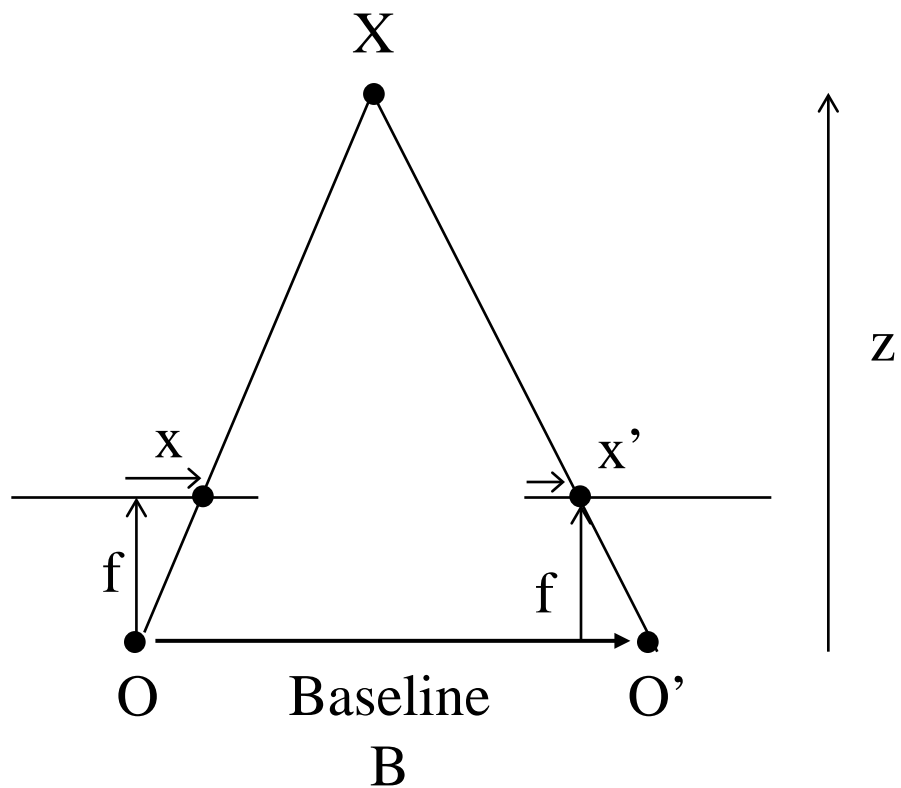




# Depth from disparity

---

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



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

Disparity is inversely proportional to depth.

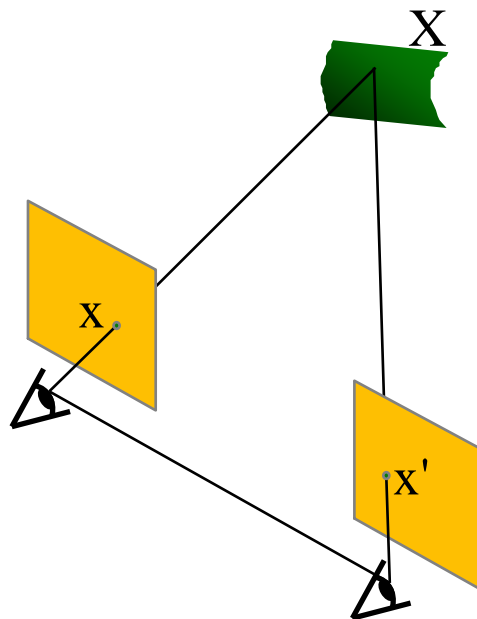
# Depth from Stereo

---

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

---

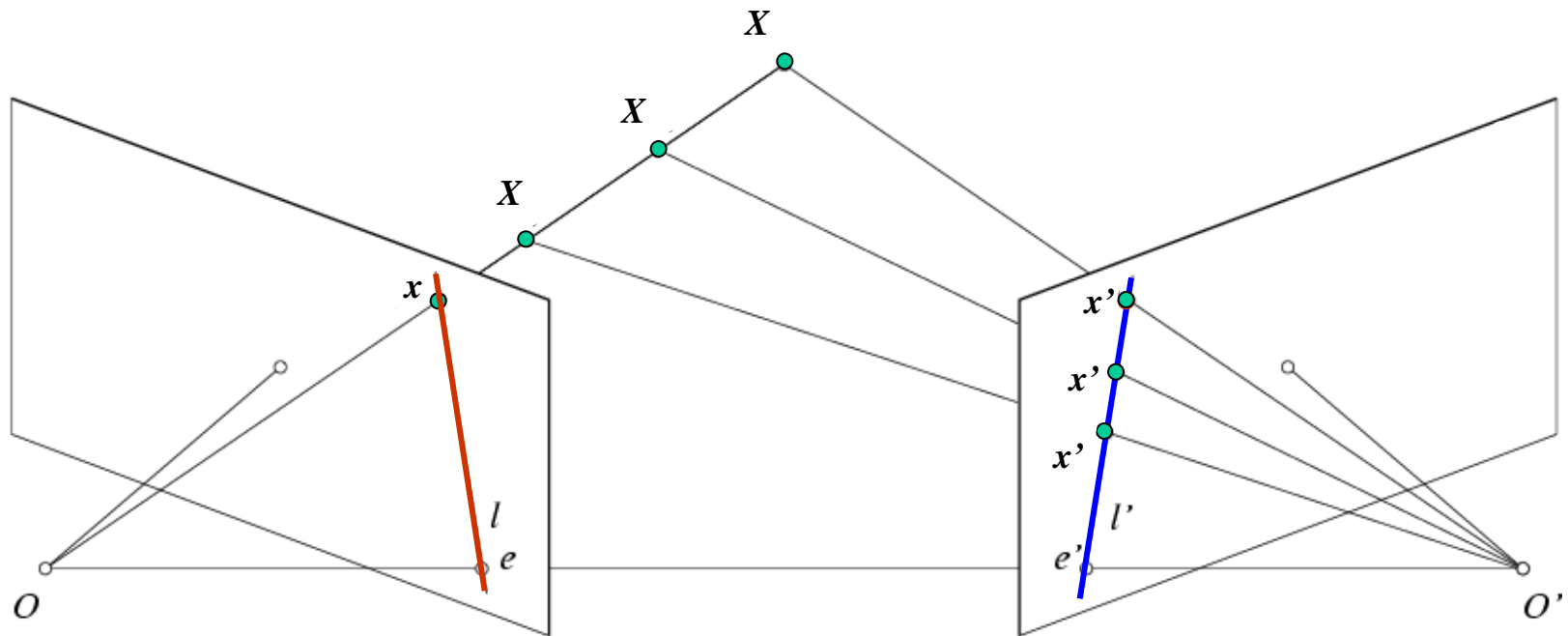


We have two images taken from cameras with different intrinsic and extrinsic parameters

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

---

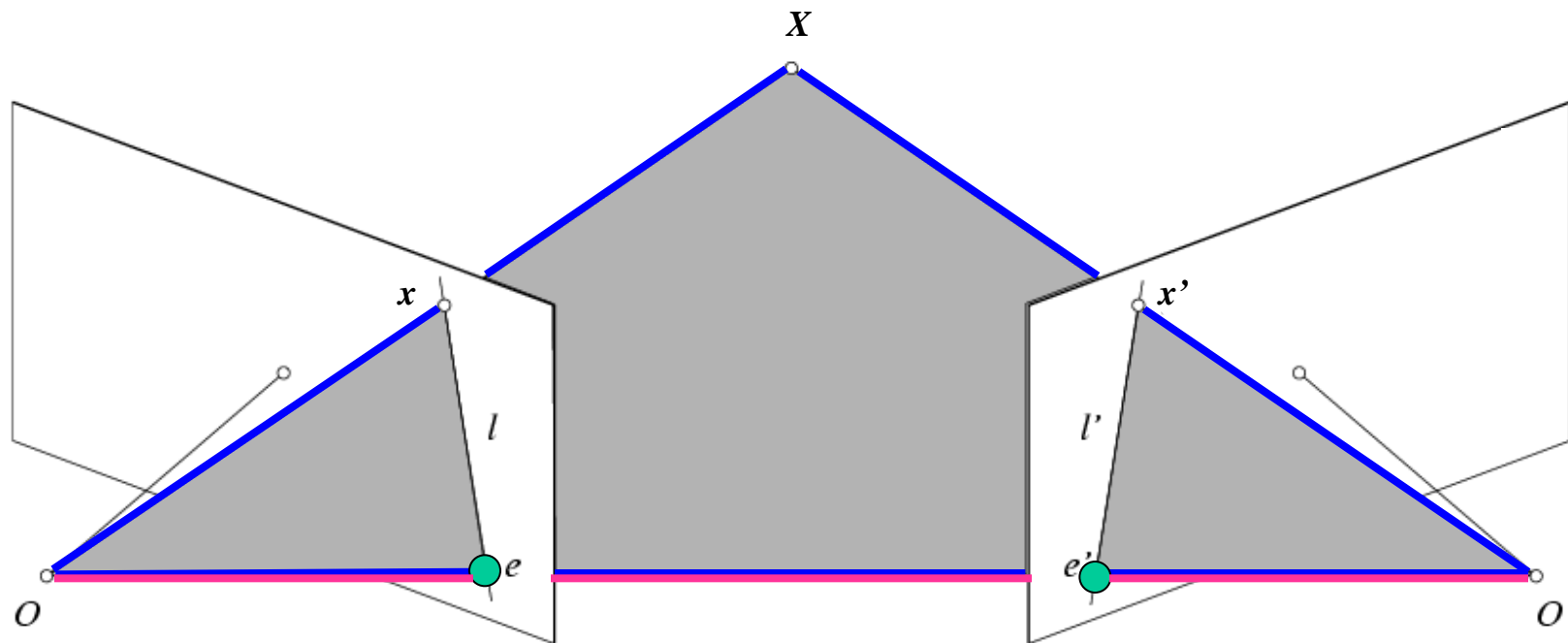


Potential matches for  $x$  have to lie on the corresponding line  $l'$ .

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

# Epipolar geometry: notation

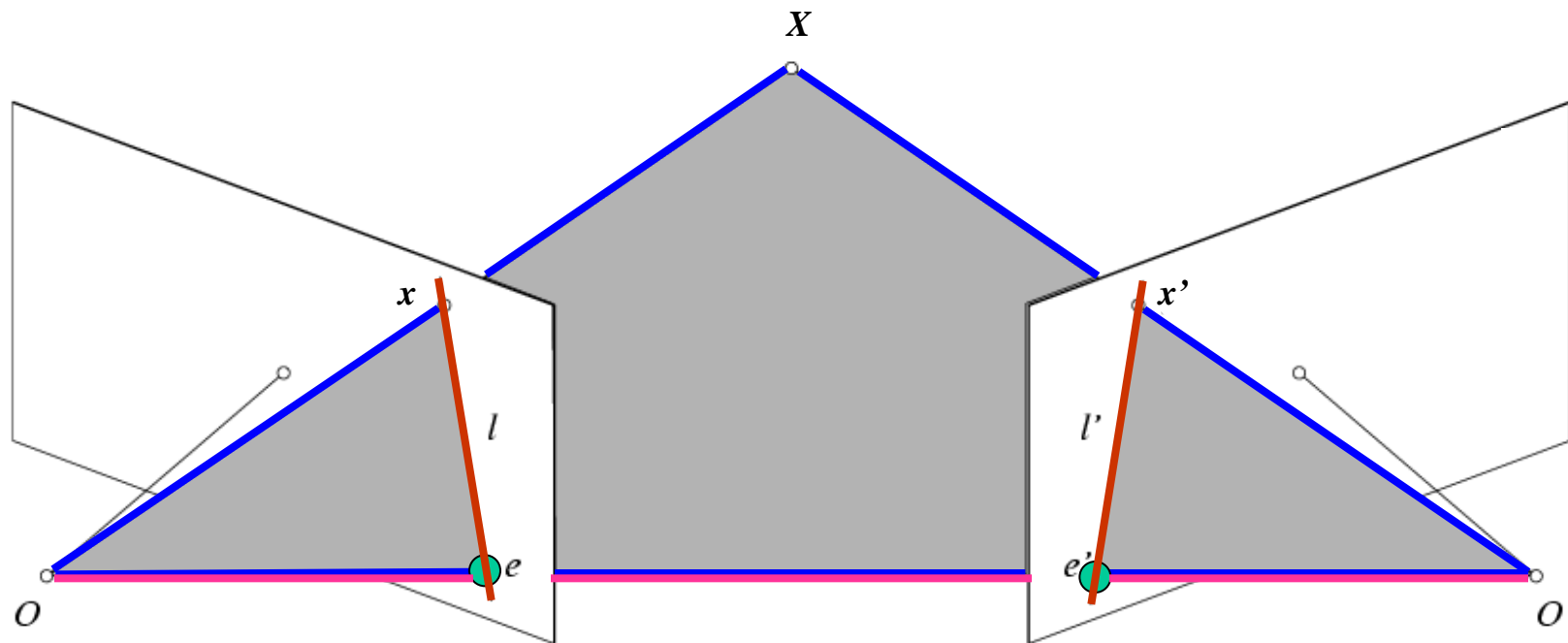
---



- **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

---

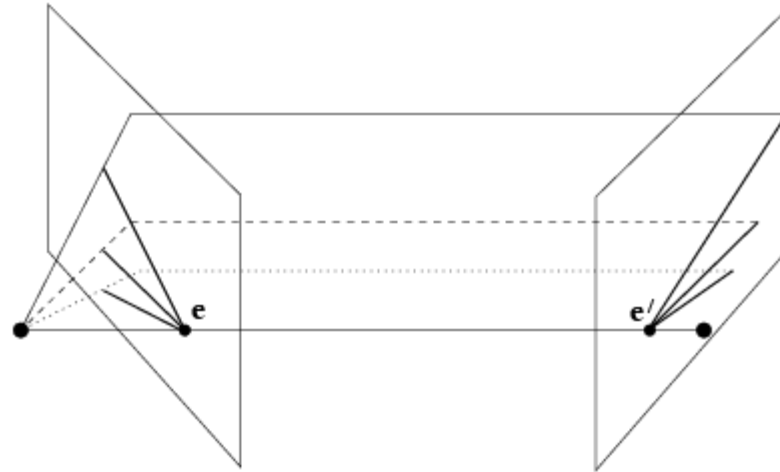


- **Baseline** – line connecting the two camera centers
- **Epipoles**  
= intersections of baseline with image planes  
= projections of the other camera center
- **Epipolar Plane** – plane containing baseline (1D family)
- **Epipolar Lines** - intersections of epipolar plane with image planes (always come in corresponding pairs)



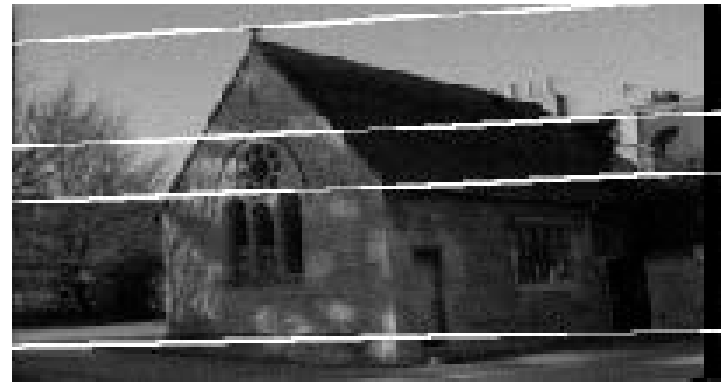
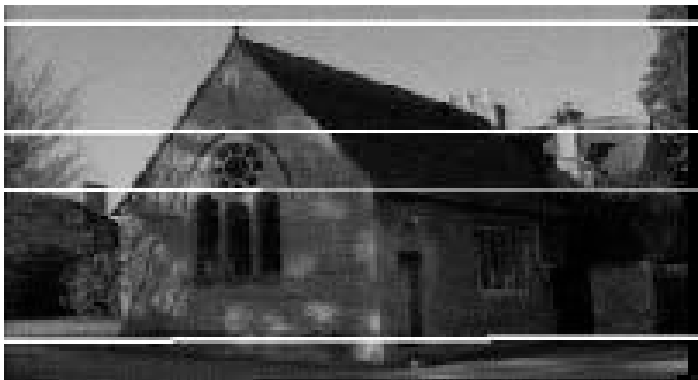
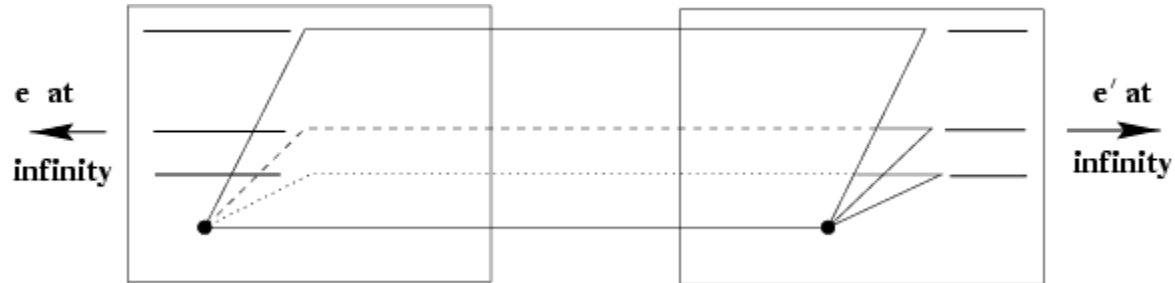
# Example: Converging cameras

---



# Example: Motion parallel to image plane

---



# Example: Forward motion

---

What would the epipolar lines look like if the camera moves directly forward?

# Example: Motion perpendicular to image plane

---



# Example: Motion perpendicular to image plane

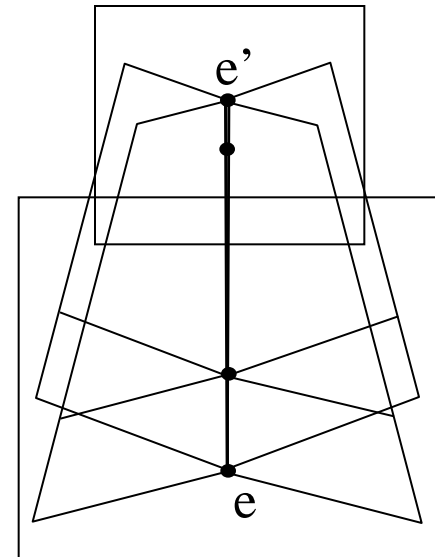
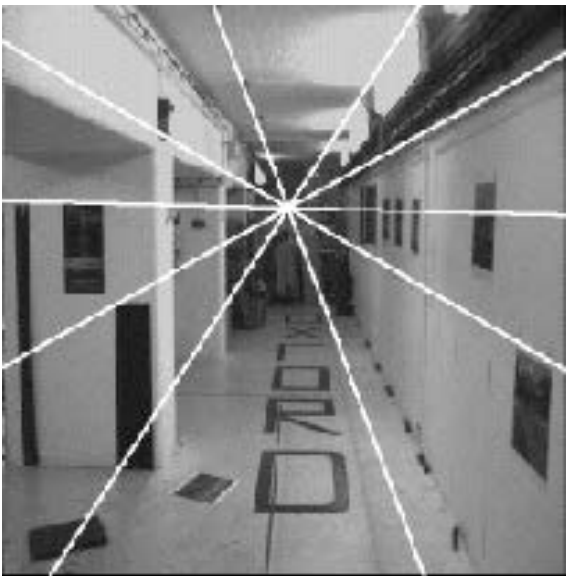
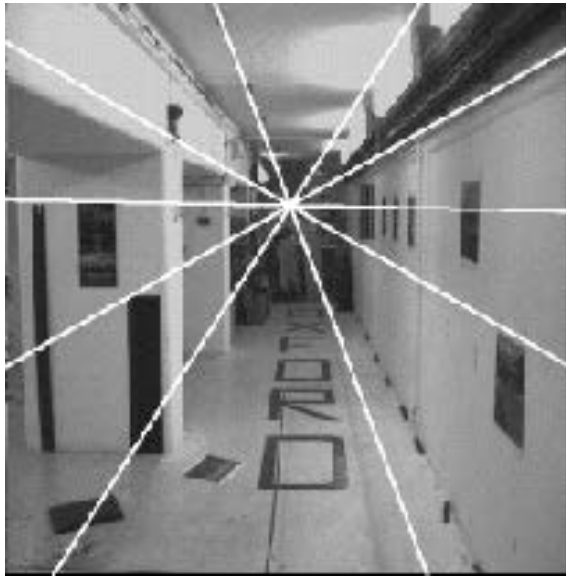
---



- Points move along lines radiating from the epipole: “focus of expansion”
- Epipole is the principal point

# Example: Forward motion

---

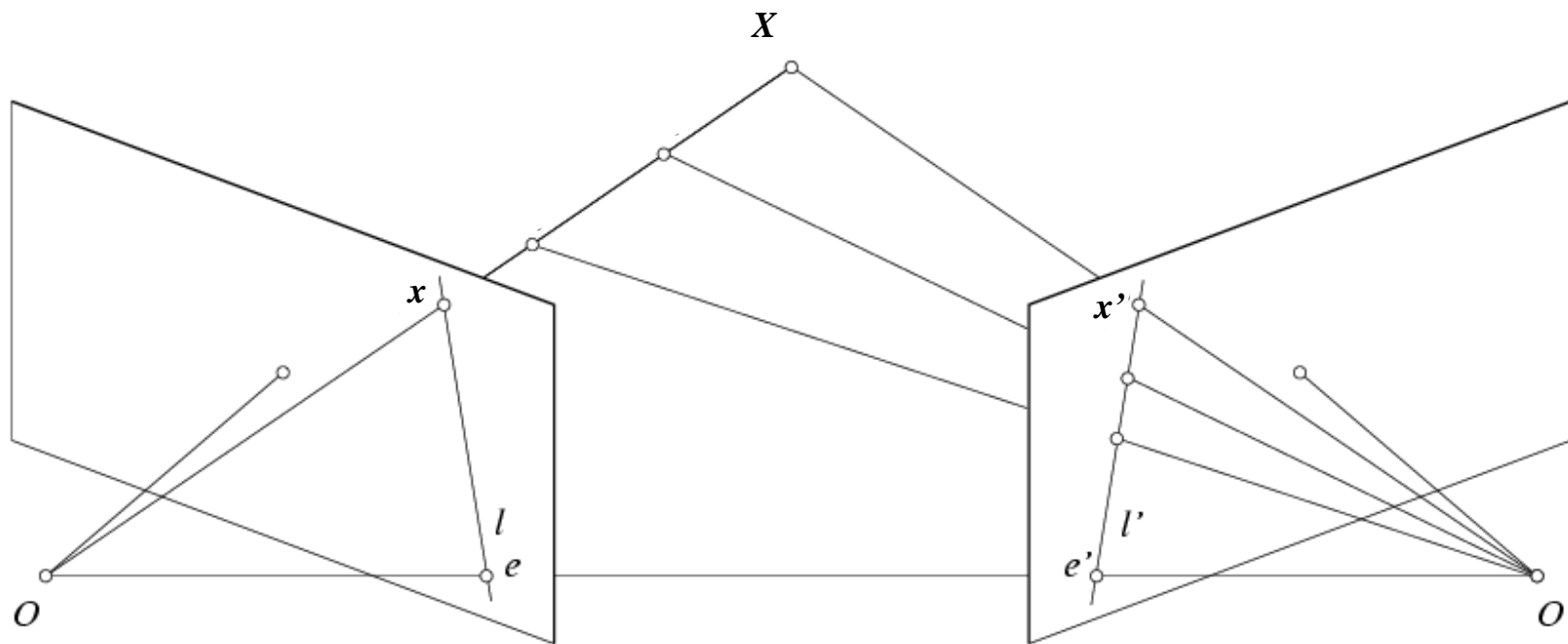


Epipole has same coordinates in both images.  
Points move along lines radiating from “Focus of expansion”



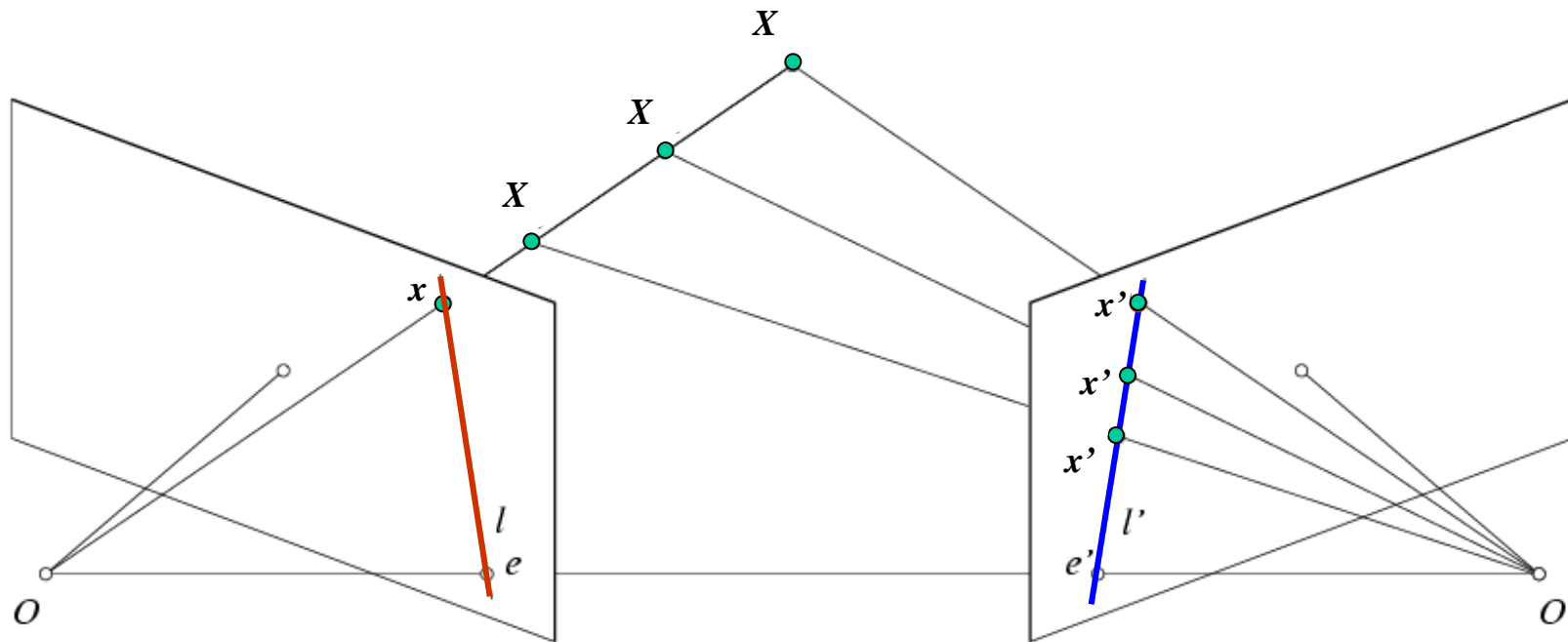
# Epipolar constraint

---



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

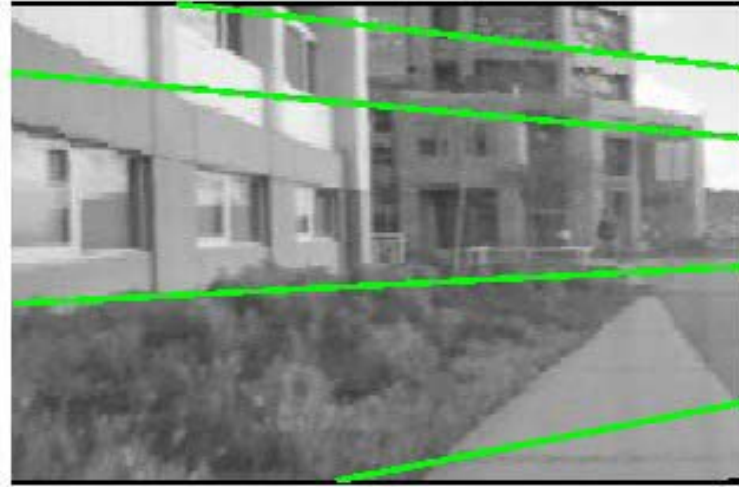
# Epipolar constraint



- 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  $l$ .

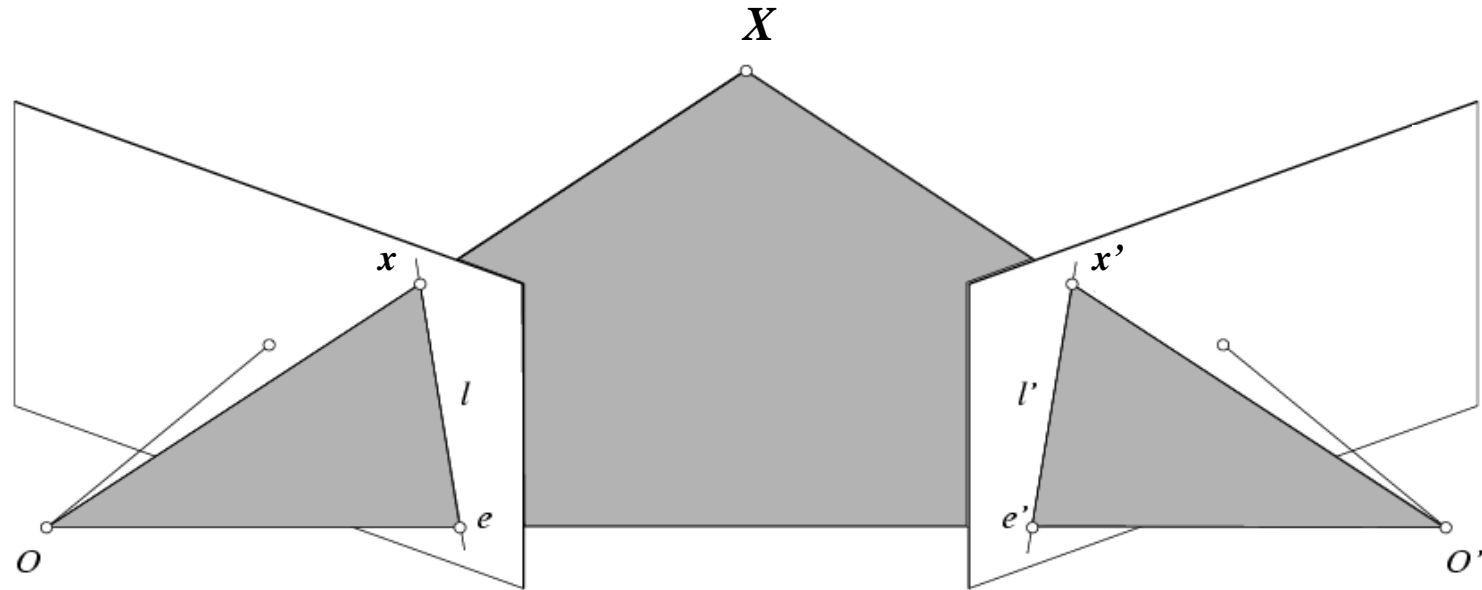
# Epipolar constraint example

---



# 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  $[\mathbf{R} \mid \mathbf{t}]$

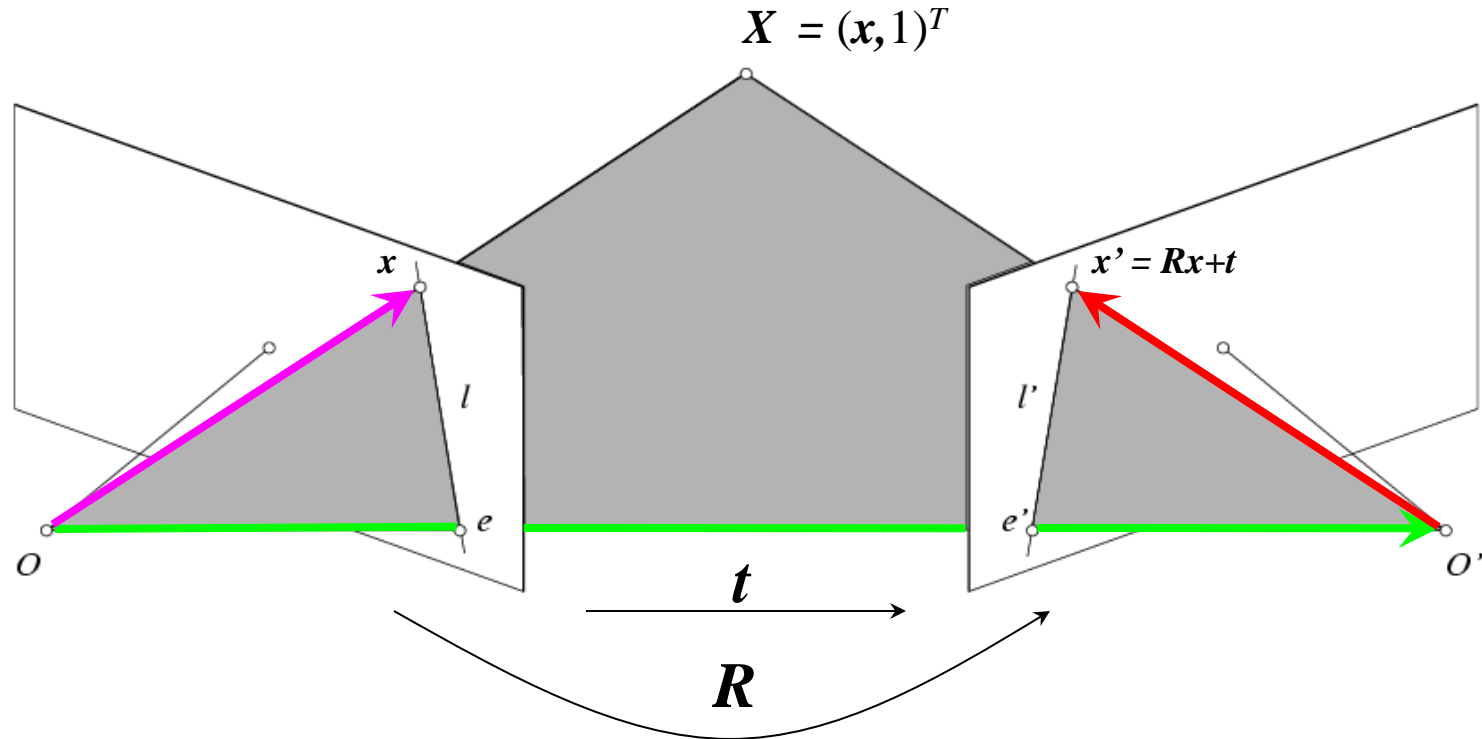
# Simplified Matrices for the 2 Cameras

---

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} = (\mathbf{I} \mid \mathbf{0})$$

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

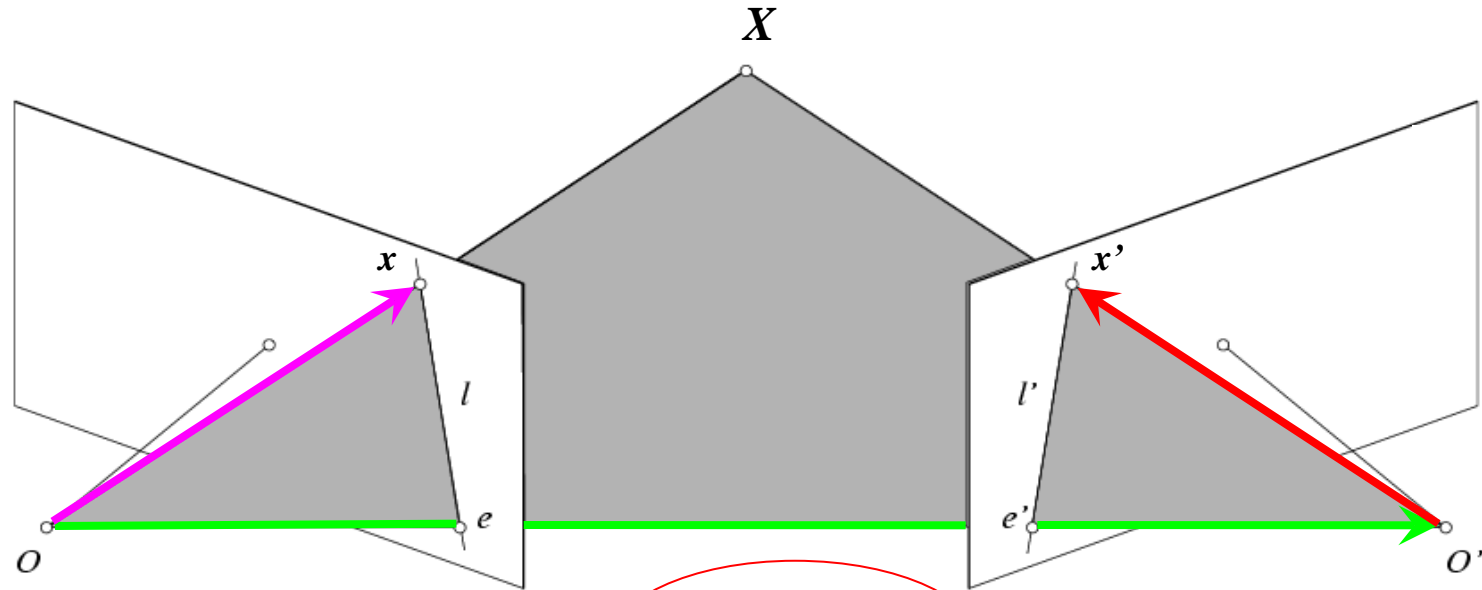
# Epipolar constraint: Calibrated case



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



# Epipolar constraint: Calibrated case

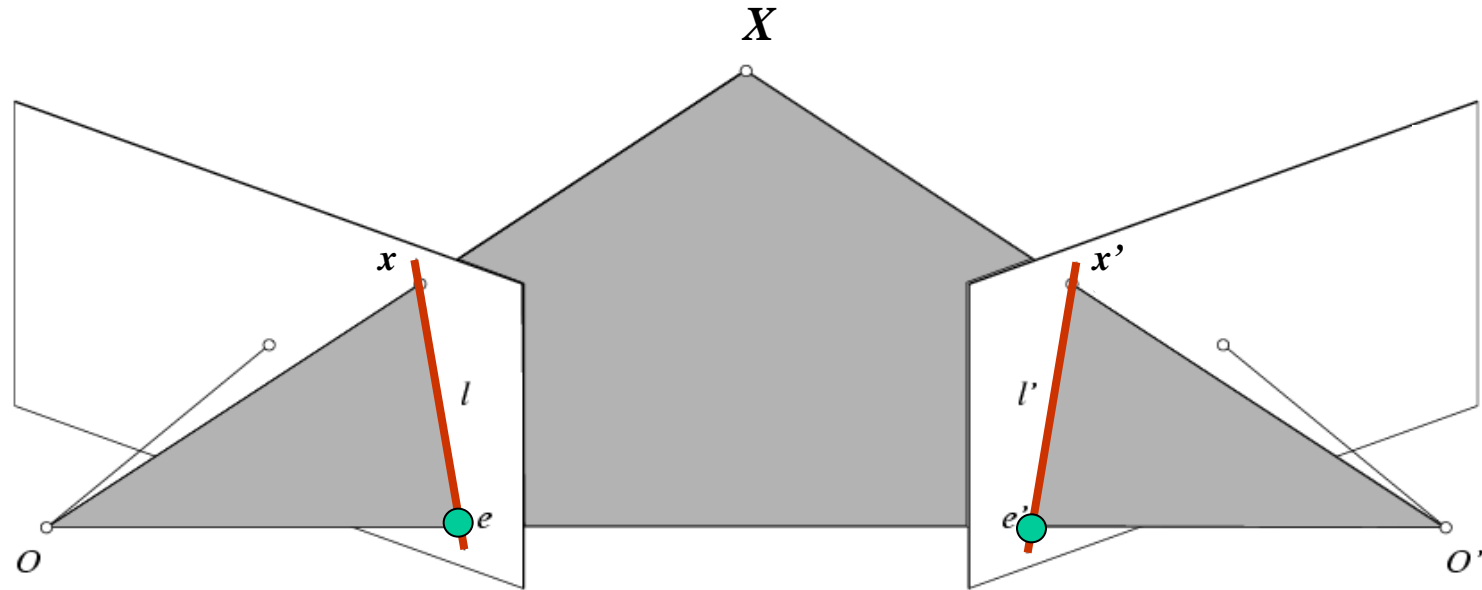


$$\mathbf{x}' \cdot [\mathbf{t} \times (\mathbf{R}\mathbf{x})] = 0 \quad \Rightarrow \quad \mathbf{x}'^T \mathbf{E} \mathbf{x} = 0 \quad \text{with} \quad \mathbf{E} = [\mathbf{t}_\times] \mathbf{R}$$

**Essential Matrix E**  
(Longuet-Higgins, 1981)

The vectors  $\mathbf{R}\mathbf{x}$ ,  $\mathbf{t}$ , and  $\mathbf{x}'$  are coplanar

# Epipolar constraint: Calibrated case

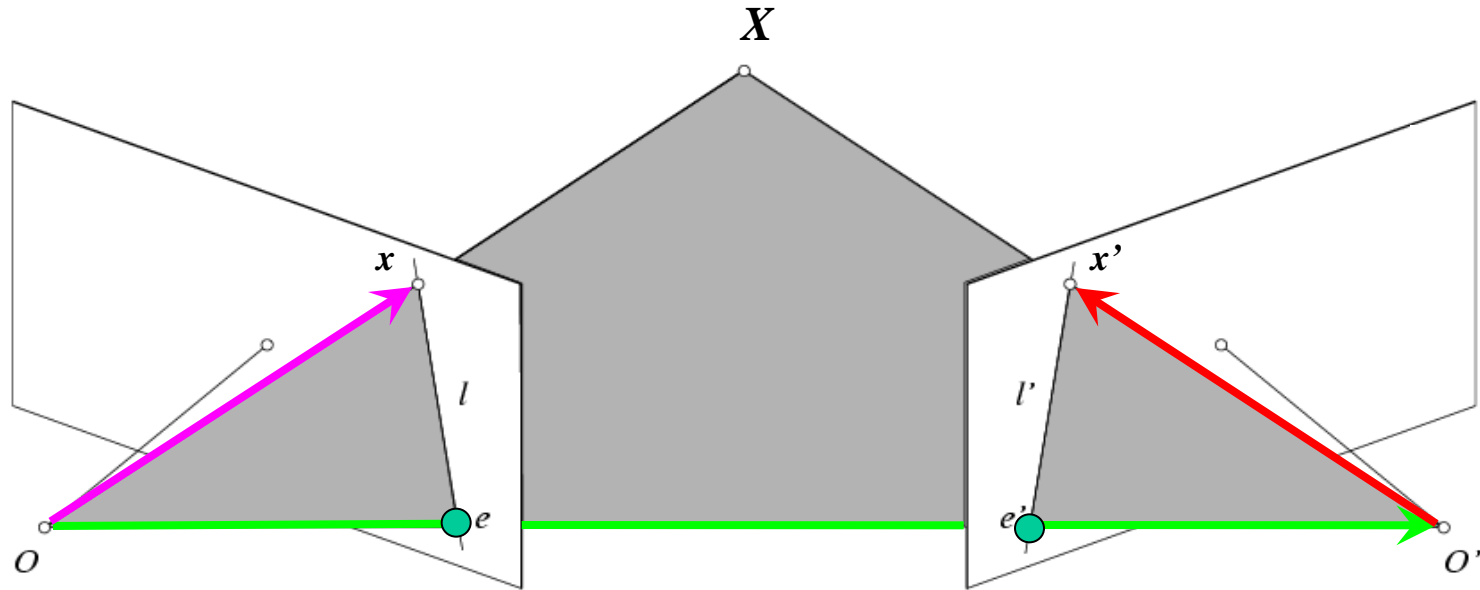


$$\mathbf{x}' \cdot [\mathbf{t} \times (\mathbf{R}\mathbf{x})] = 0 \quad \Rightarrow \quad \mathbf{x}'^T \mathbf{E} \mathbf{x} = 0 \quad \text{with} \quad \mathbf{E} = [\mathbf{t}_\times] \mathbf{R}$$

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

# Epipolar constraint: Uncalibrated case

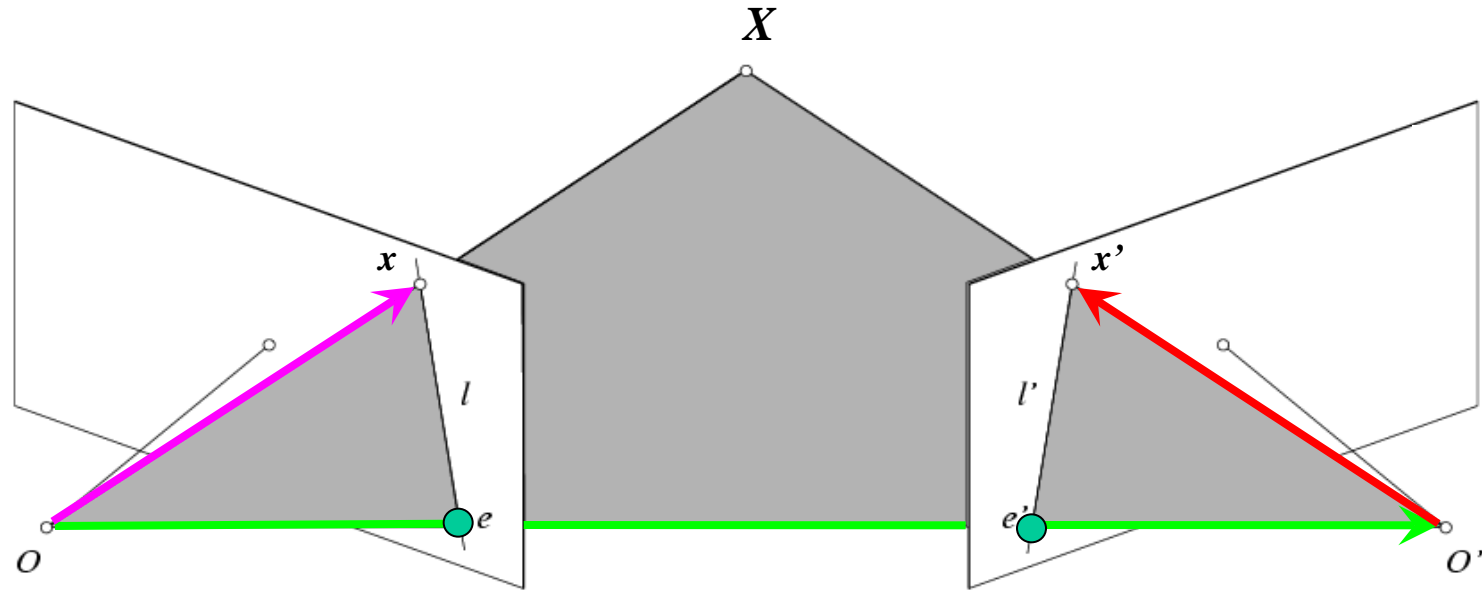
---



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

$$\hat{\mathbf{x}}'^T \mathbf{E} \hat{\mathbf{x}} = 0 \quad \hat{\mathbf{x}} = \mathbf{K}^{-1} \mathbf{x}, \quad \hat{\mathbf{x}}' = \mathbf{K}'^{-1} \hat{\mathbf{x}}'_{29}$$

# Epipolar constraint: Uncalibrated case



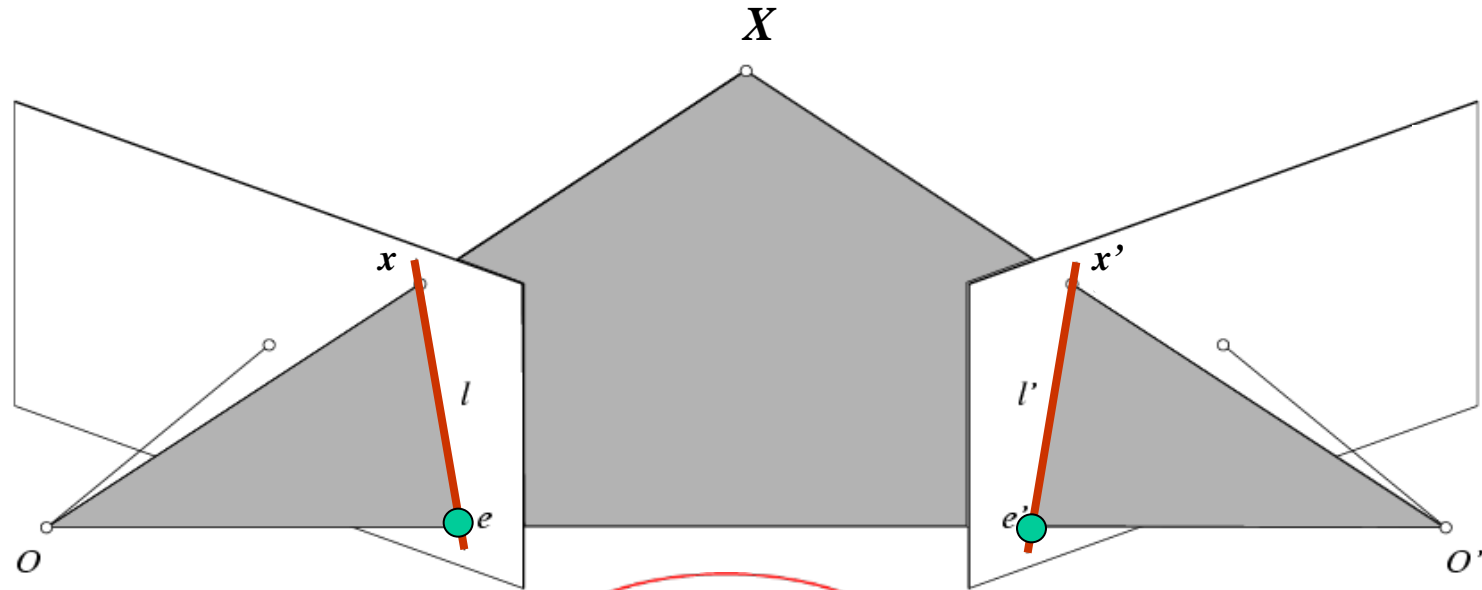
$$\hat{x}'^T E \hat{x} = 0 \quad \longrightarrow \quad x'^T F x = 0 \quad \text{with} \quad F = K'^{-T} E K^{-1}$$

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

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

**Fundamental Matrix**  
(Faugeras and Luong, 1992)

# Epipolar constraint: Uncalibrated case



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

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

# The eight-point algorithm

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

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

$\mathbf{A}$

Minimize:

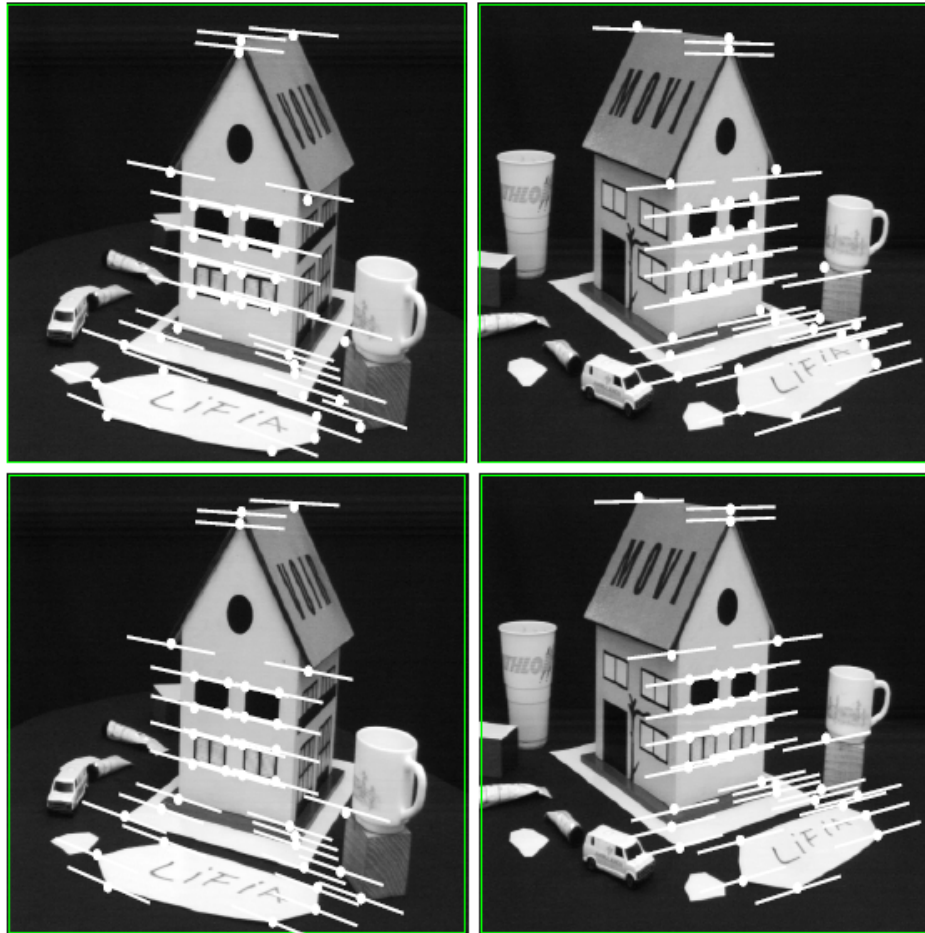
$$\sum_{i=1}^N (\mathbf{x}'_i^T \mathbf{F} \mathbf{x}_i)^2$$

under the constraint

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

Smallest  
eigenvalue of  
 $\mathbf{A}^T \mathbf{A}$

# Comparison of estimation algorithms



	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

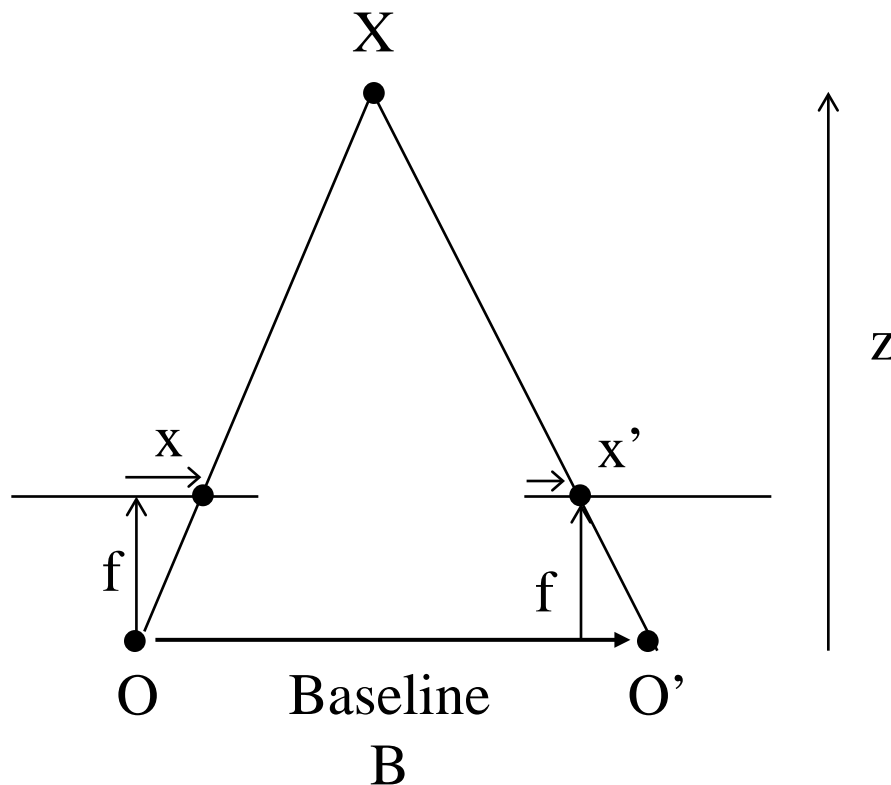




# Depth from disparity

---

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

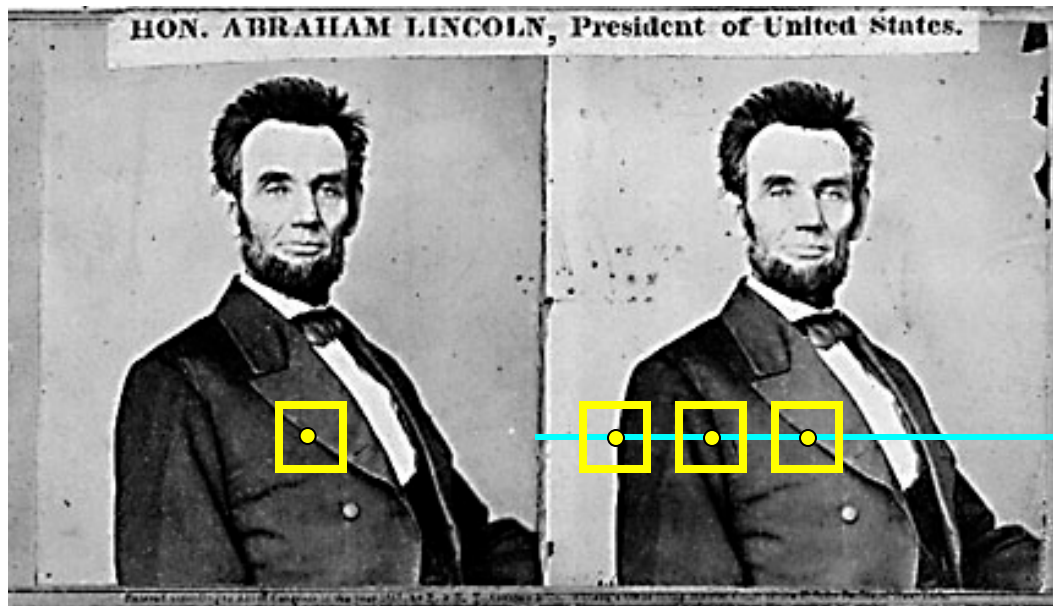


$$\text{disparity} = x - x' = \frac{B \cdot f}{z}$$

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  $\text{depth}(x) = fB/(x-x')$

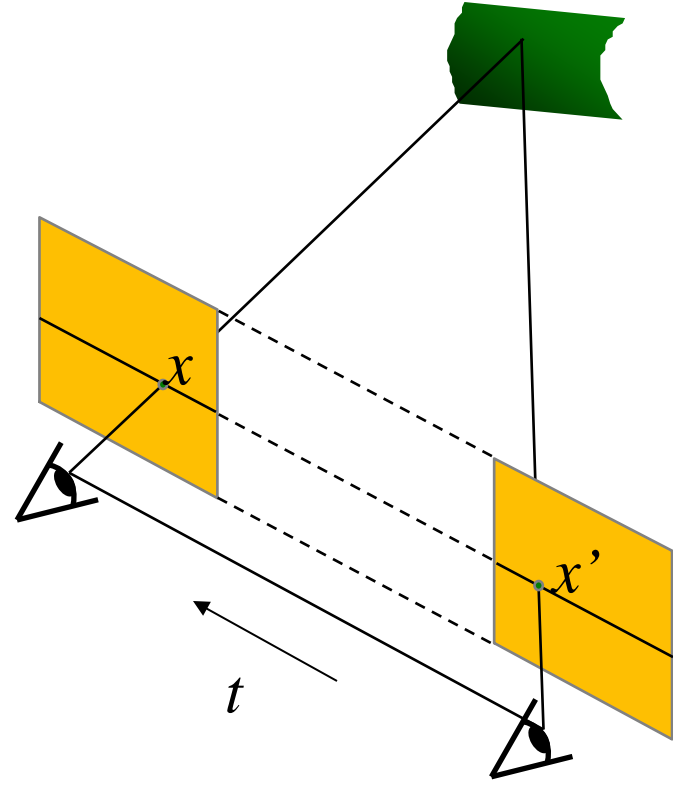
# Simplest Case: Parallel images

Epipolar constraint:

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

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

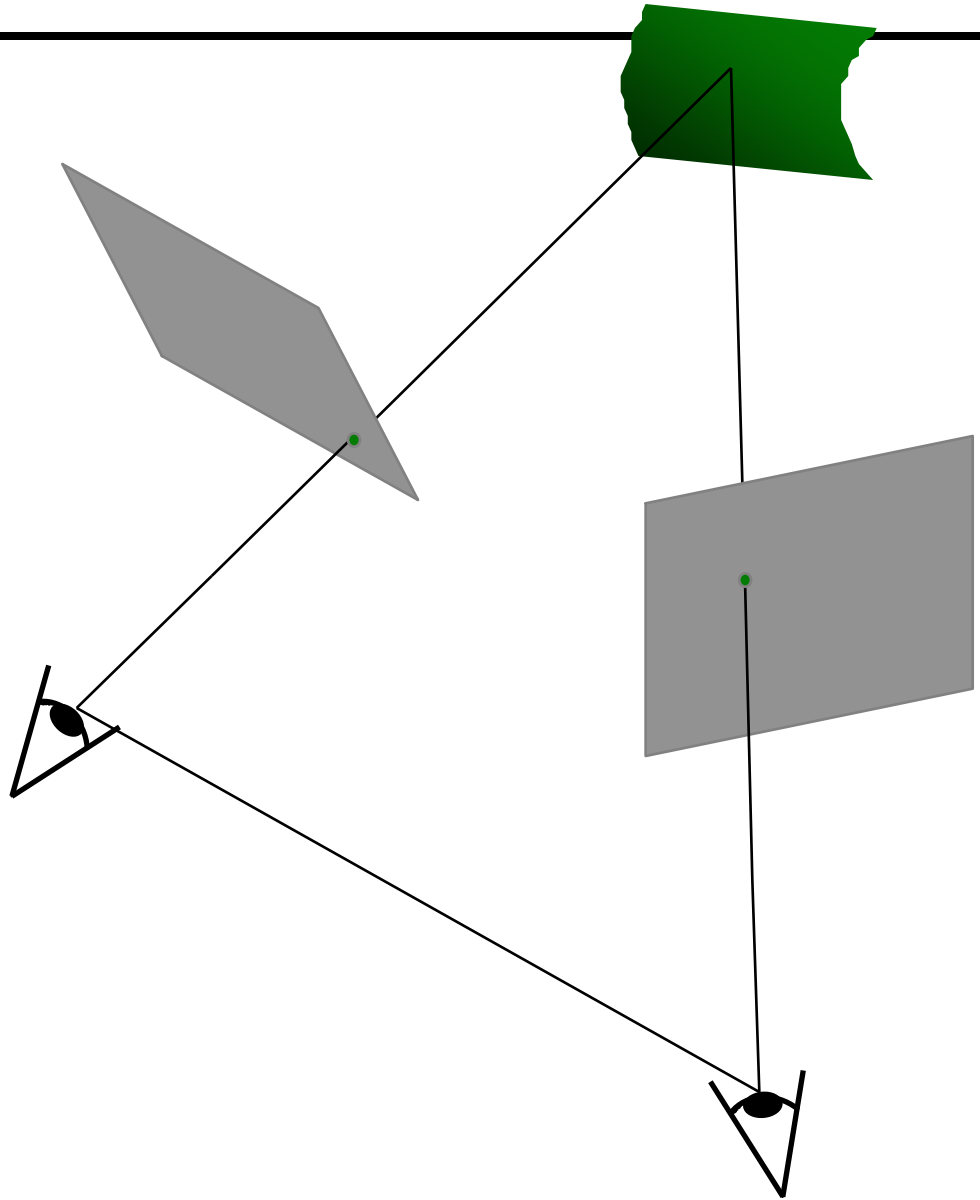
$$E = t \times R = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T \\ 0 & T & 0 \end{bmatrix}$$


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

The y-coordinates of corresponding points are the same

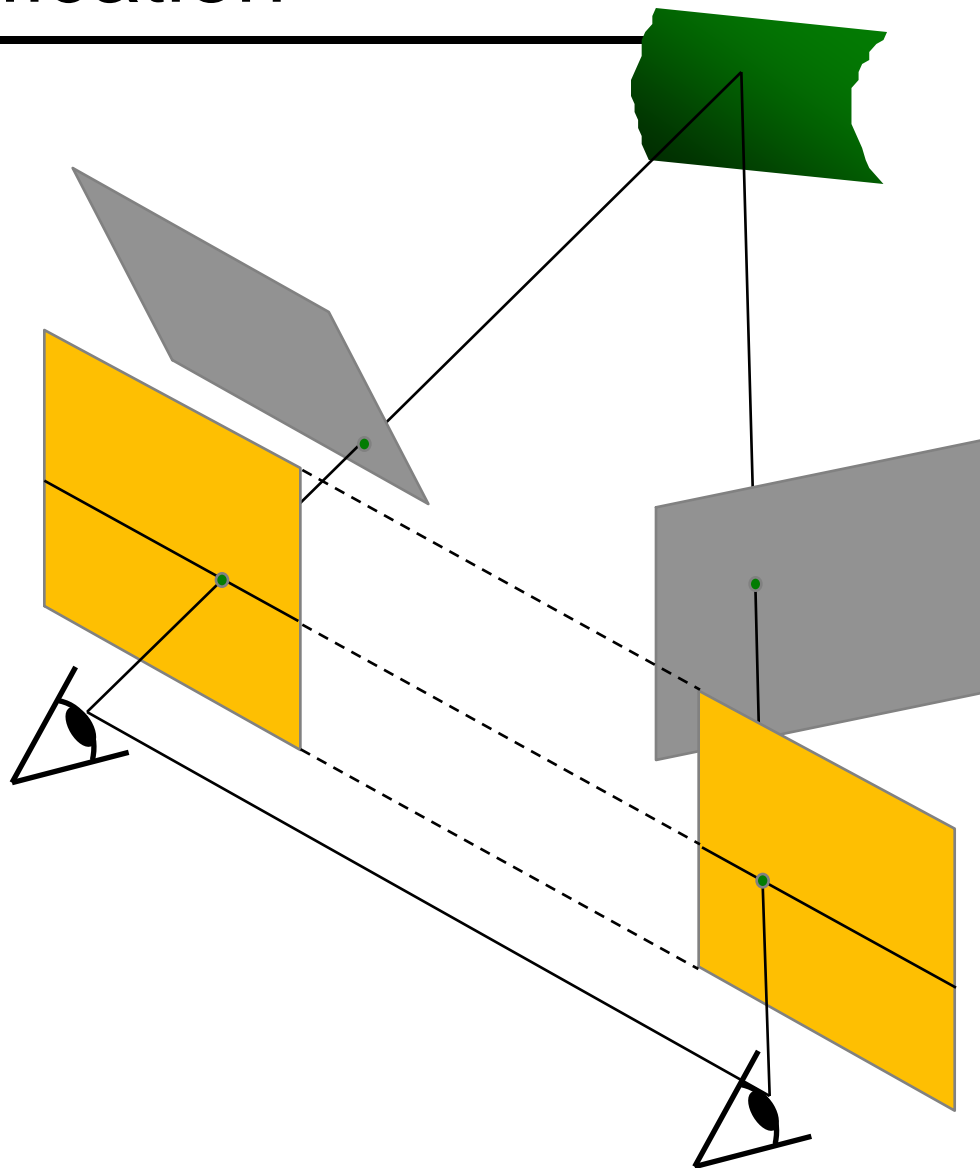
# Stereo image rectification

---



# 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

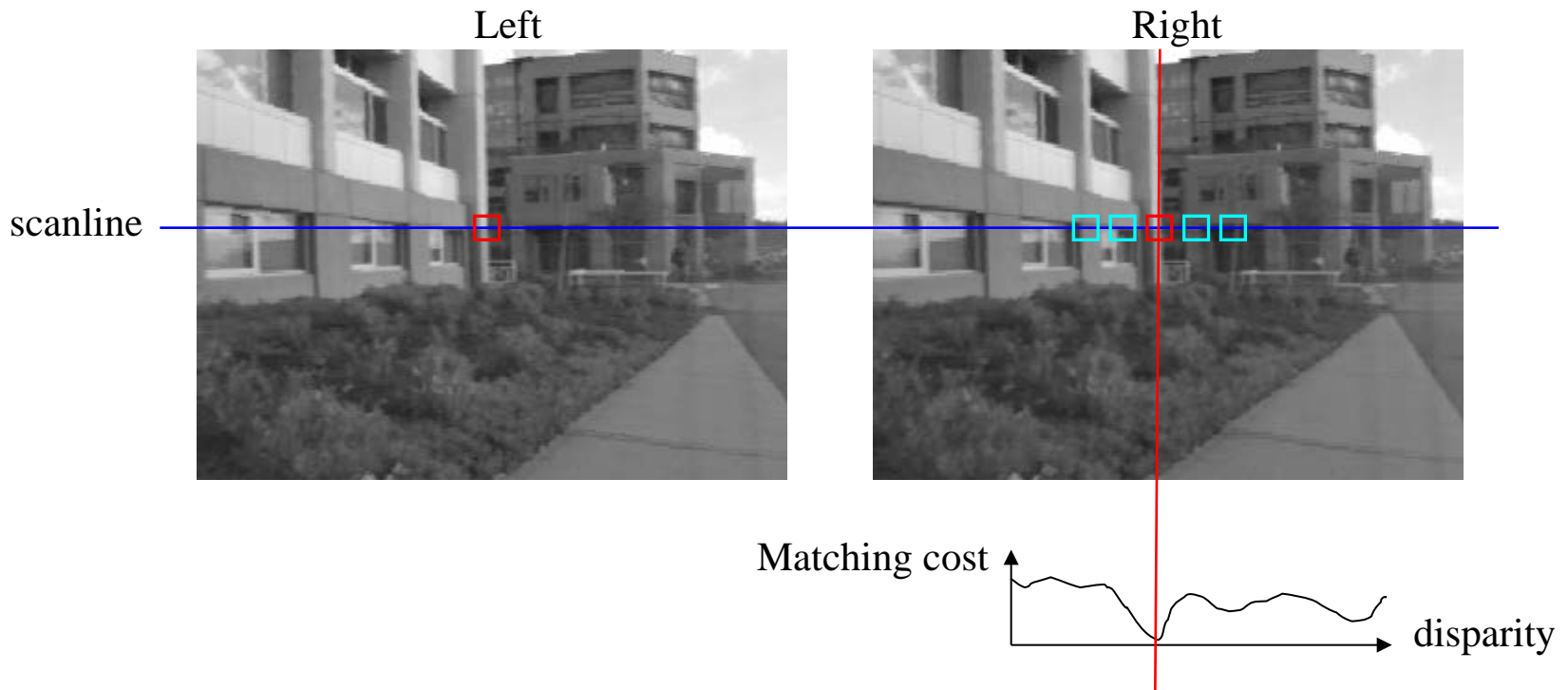
Unrectified



Rectified

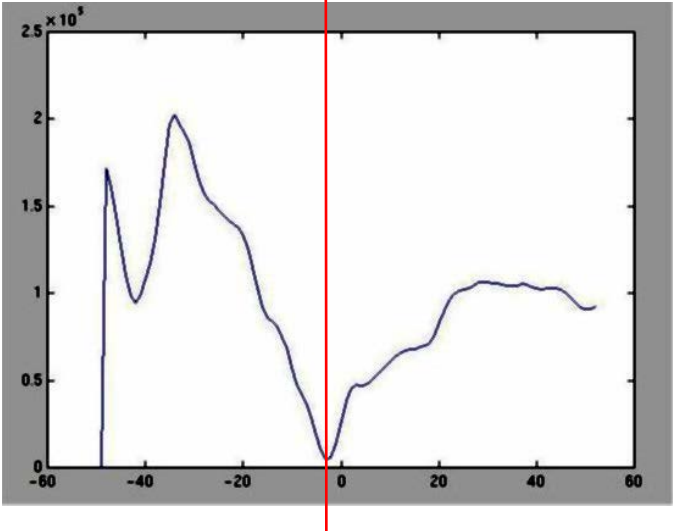
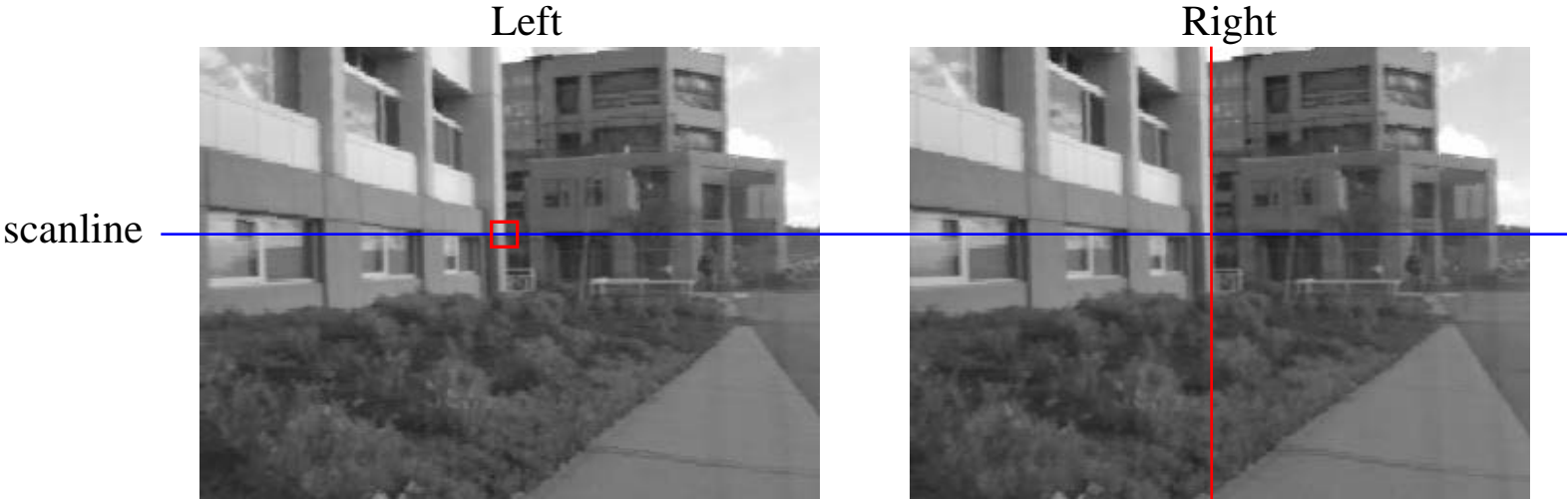


# Correspondence search: for HW3



- 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**

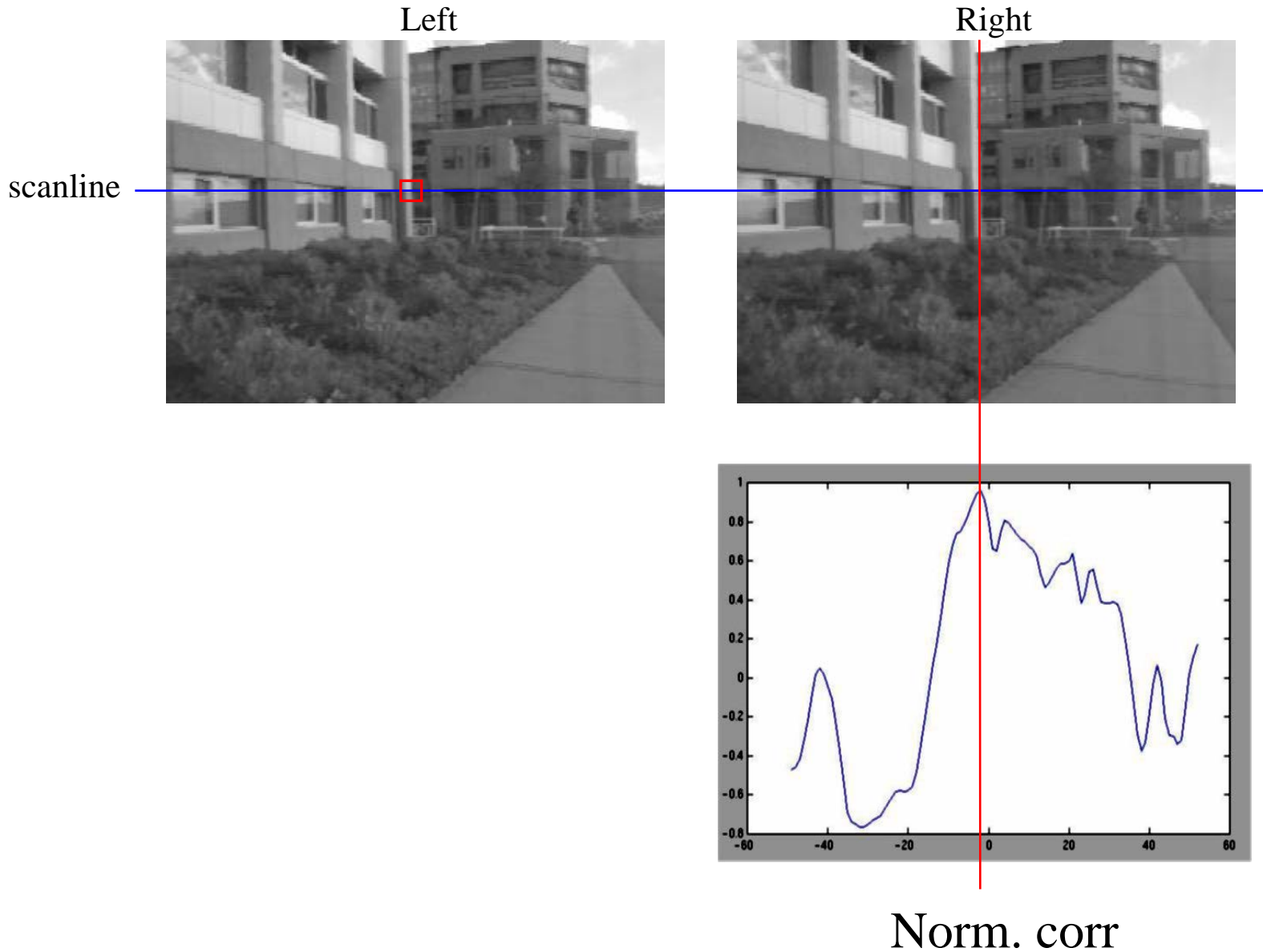
# Correspondence search



SSD



# Correspondence search

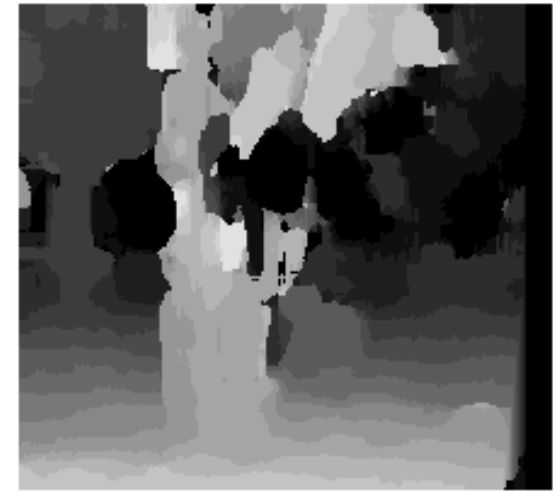


# Effect of window size

---



$W = 3$

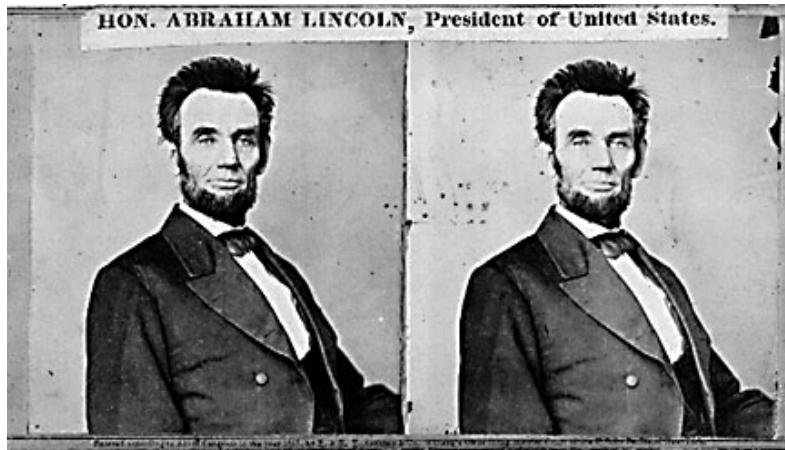


$W = 20$

- Smaller window
  - + More detail
  - More noise
- Larger window
  - + Smoother disparity maps
  - Less detail
  - Fails near boundaries

# Failures of correspondence search

---



Textureless surfaces



Occlusions, repetition

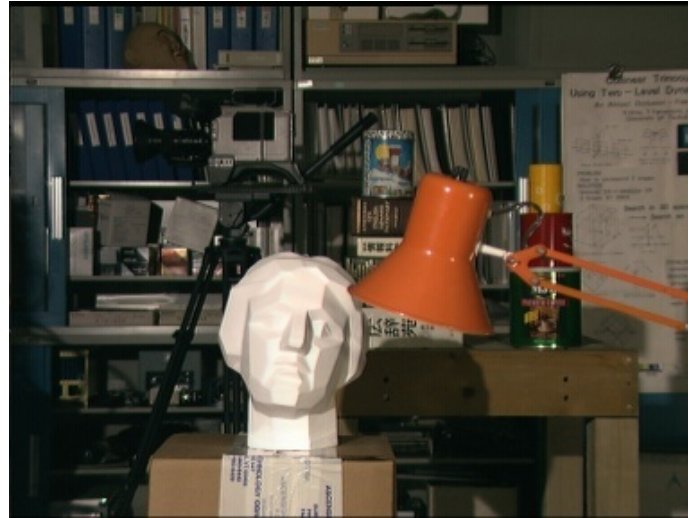


Non-Lambertian surfaces, specularities

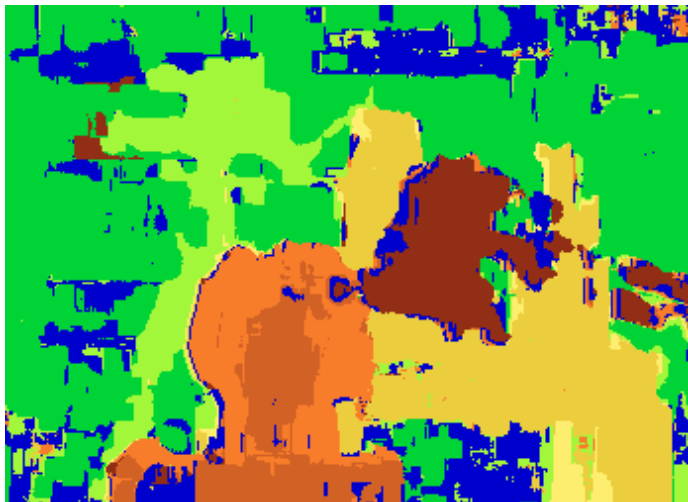
# Results with window search

---

Data



Window-based matching



Ground truth



# How can we improve window-based matching?

---

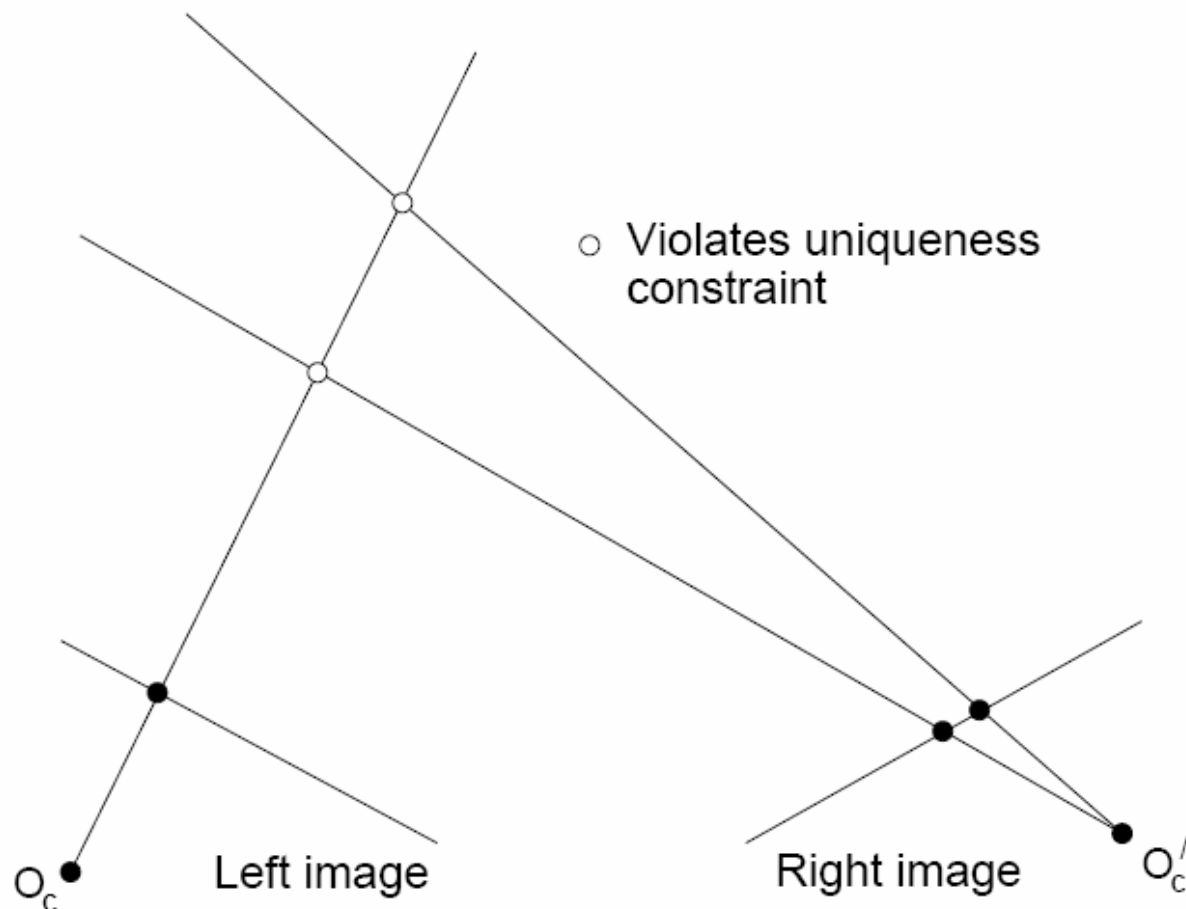
So far, matches are independent for each point

What constraints or priors can we add?

# Stereo constraints/priors

---

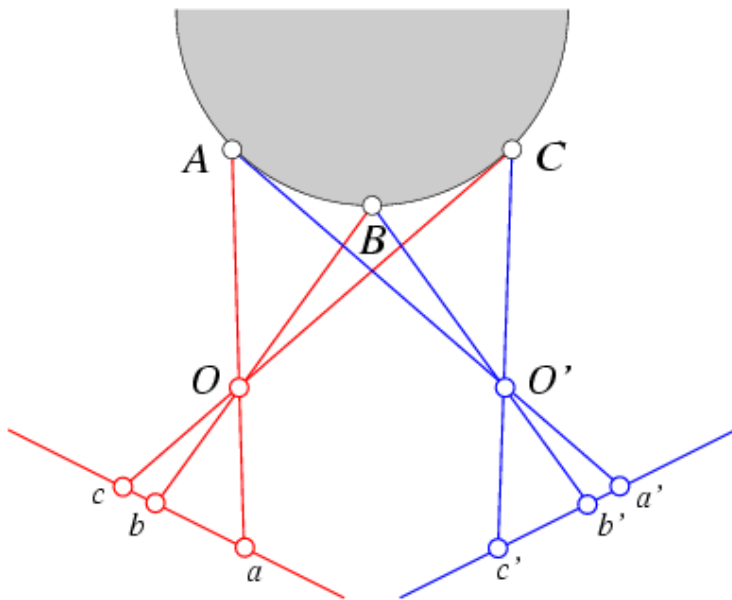
- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image



# Stereo constraints/priors

---

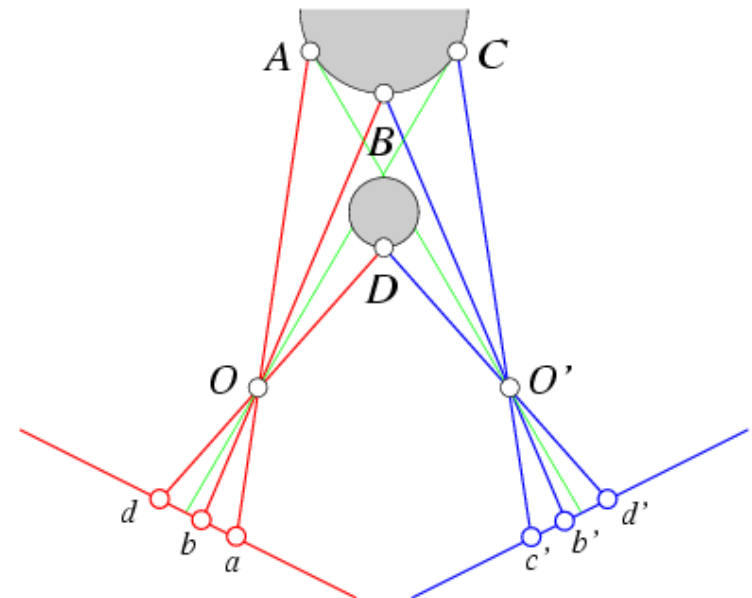
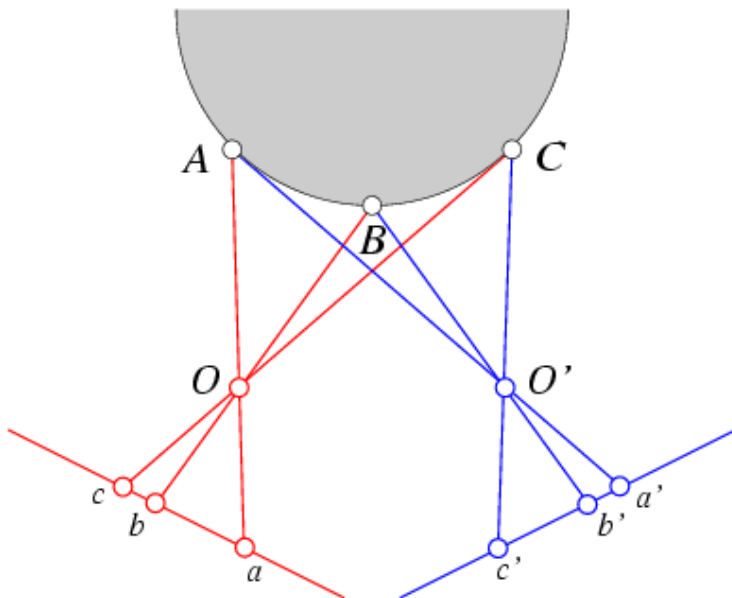
- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image
- Ordering
  - Corresponding points should be in the same order in both views



# 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





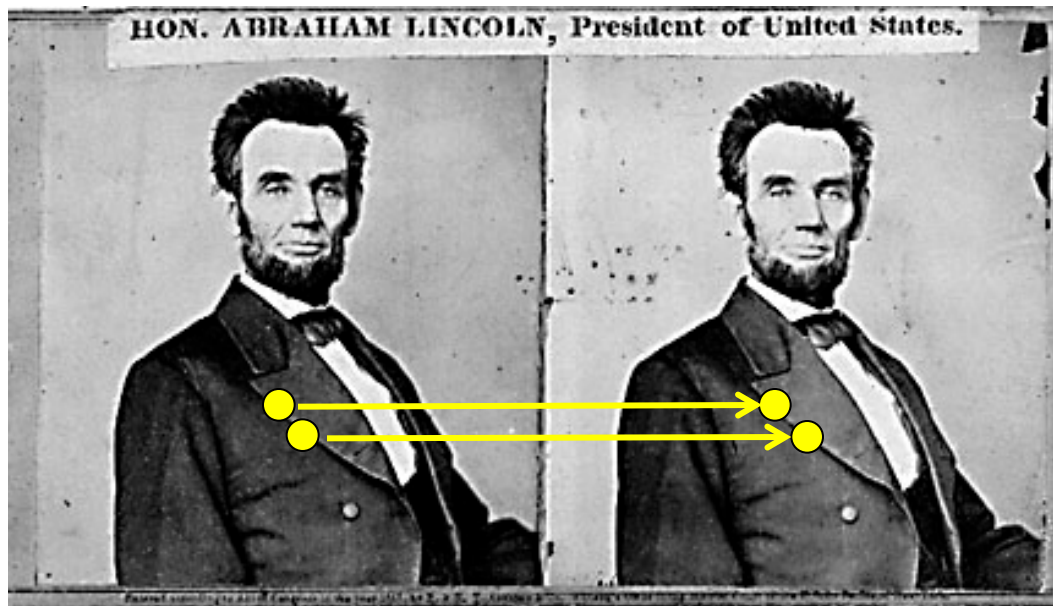
# Priors and constraints

---

- 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
  - If two pixels are adjacent, they should (usually) move about the same amount

# Matching windows: for HW3

## Similarity Measure

## Formula

Sum of Absolute Differences (SAD)

$$\sum_{(i,j) \in W} |I_1(i,j) - I_2(x+i, y+j)|$$

Sum of Squared Differences (SSD)

$$\sum_{(i,j) \in W} (I_1(i,j) - I_2(x+i, y+j))^2$$

Zero-mean SAD

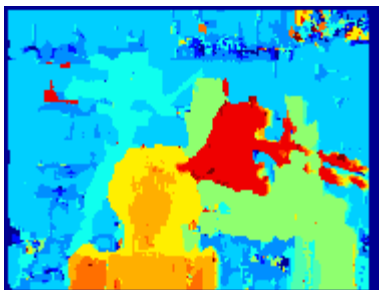
$$\sum_{(i,j) \in W} |I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i, y+j) + \bar{I}_2(x+i, y+j)|$$

Locally scaled SAD

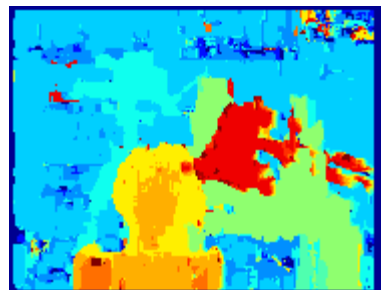
$$\sum_{(i,j) \in W} |I_1(i,j) - \frac{\bar{I}_1(i,j)}{\bar{I}_2(x+i, y+j)} I_2(x+i, y+j)|$$

Normalized Cross Correlation (NCC)

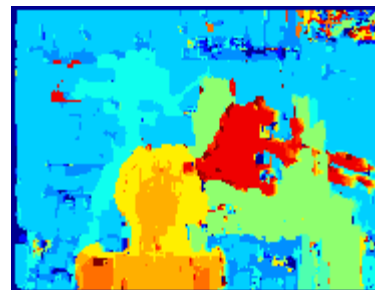
$$\frac{\sum_{(i,j) \in W} I_1(i,j) \cdot I_2(x+i, y+j)}{\sqrt{\sum_{(i,j) \in W} I_1^2(i,j) \cdot \sum_{(i,j) \in W} I_2^2(x+i, y+j)}}$$



SAD



SSD



NCC



Ground truth 54

# Real-time stereo

---



[Nomad robot](http://www.frc.ri.cmu.edu/projects/meteorobot/index.html) searches for meteorites in Antarctica  
<http://www.frc.ri.cmu.edu/projects/meteorobot/index.html>

Used for robot navigation (and other tasks)

- Several software-based real-time stereo techniques have been developed (most based on simple discrete search)

# Why does stereo fail?

---

Fronto-Parallel Surfaces: Depth is constant within the region of local support





# Why does stereo fail?

---

Monotonic Ordering - Points along an epipolar scanline appear in the same order in both stereo images

Occlusion – All points are visible in each image



# Why does stereo fail?

---

Image Brightness Constancy: Assuming Lambertian surfaces, the brightness of corresponding points in stereo images are the same.



# Why does stereo fail?

---

Match Uniqueness: For every point in one stereo image, there is at most one corresponding point in the other image.





# Stereo reconstruction pipeline

---

## Steps

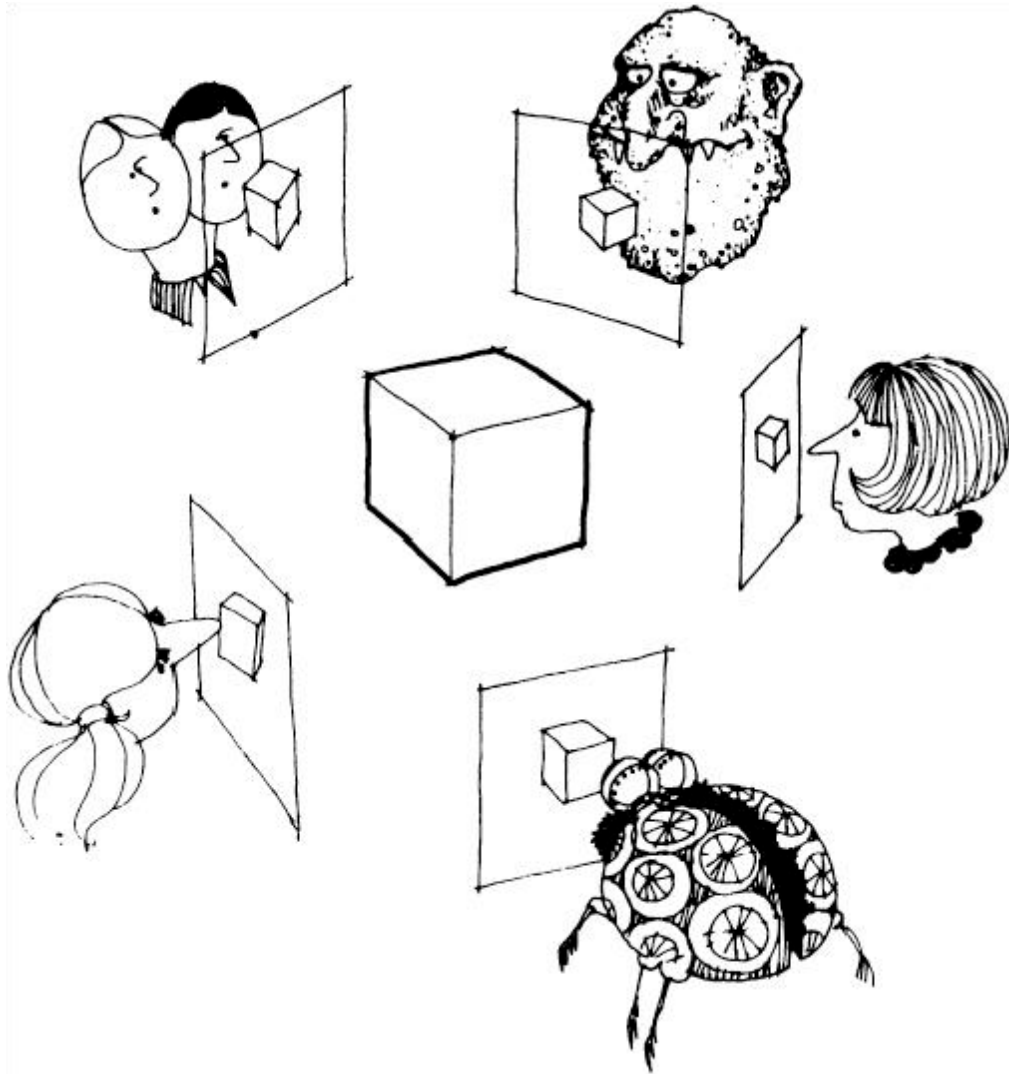
- Calibrate cameras
- Rectify images
- **Compute disparity**
- Estimate depth

## What will cause errors?

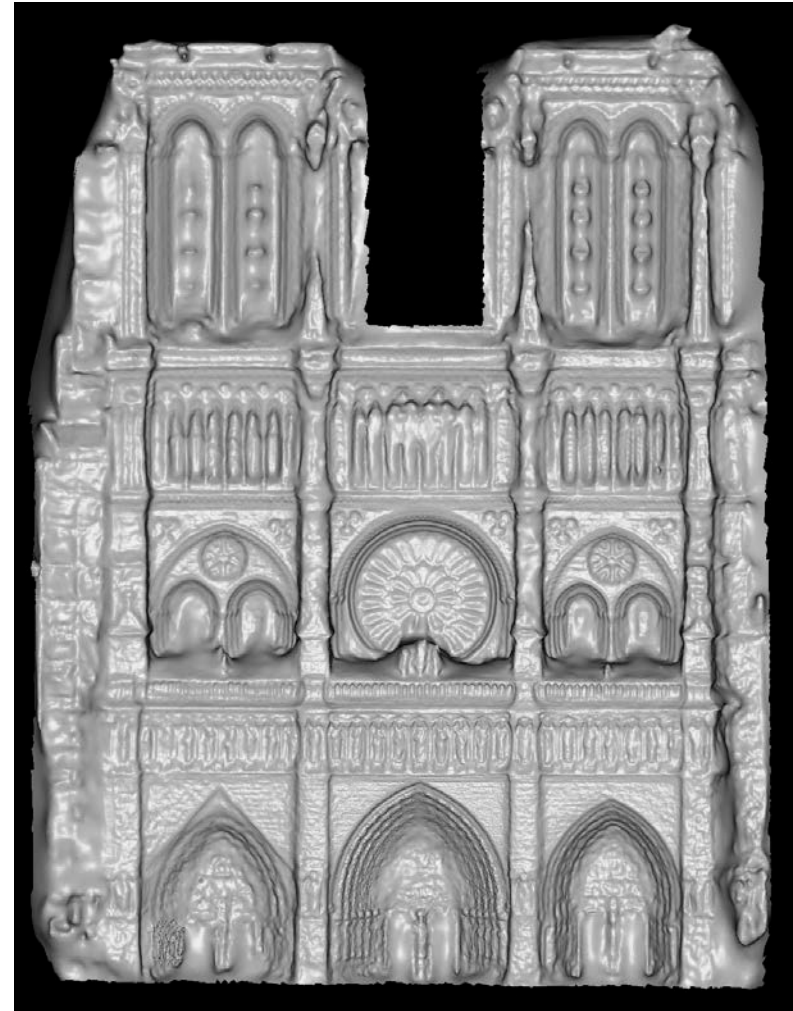
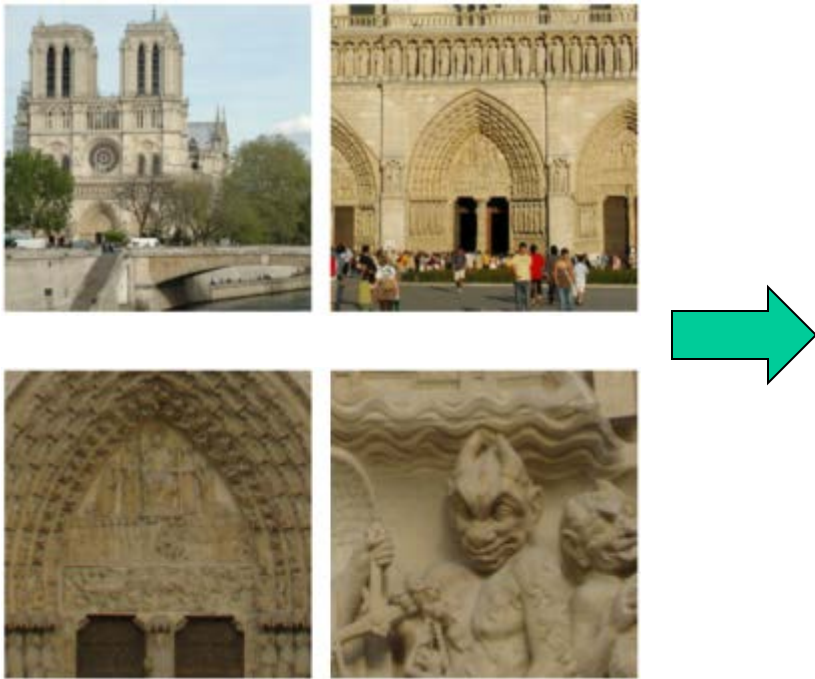
- Camera calibration errors
- Poor image resolution
- Occlusions
- Violations of brightness constancy (specular reflections)
- Large motions
- Low-contrast image regions

# 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](#),