Stereo

Slide credits: Ioannis Gkioulekas, Kris Kitani, Srinivasa Narasimhan

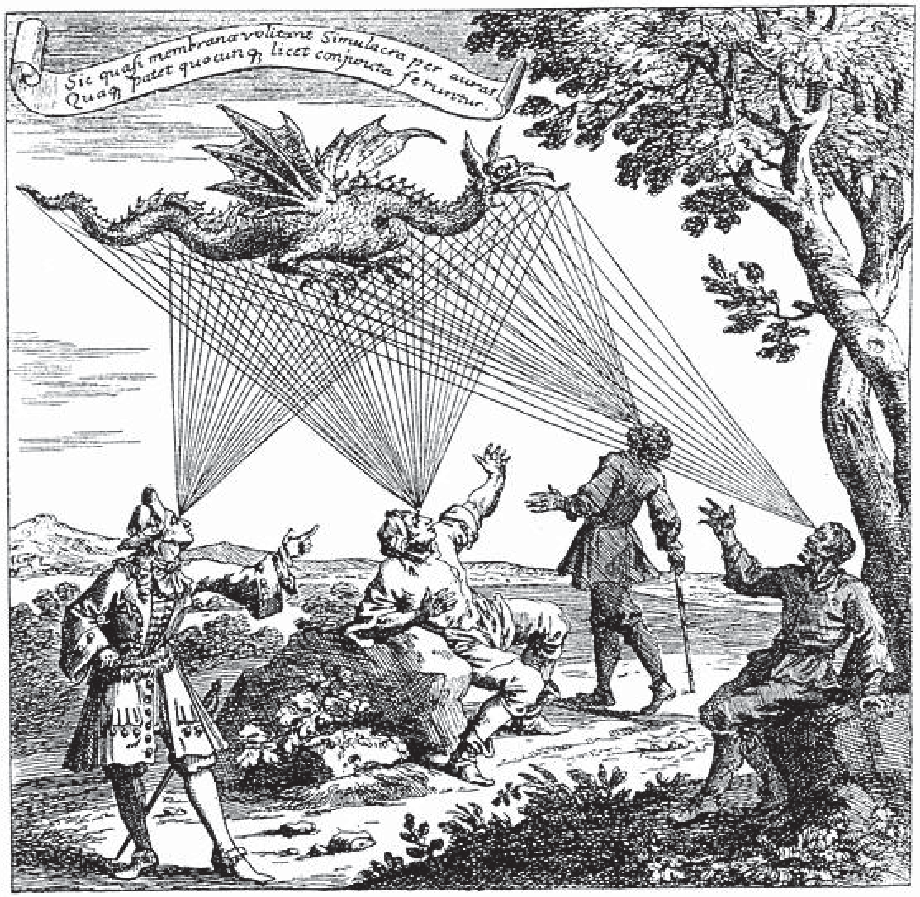
Overview of today's lecture

- Revisiting triangulation.
- Disparity.
- Stereo rectification.
- Stereo matching.
- Improving stereo matching.
- Structured light.

The Fundamental Matrix Song

https://www.youtube.com/watch?v=DgGV3l82NTk

Structure from motion

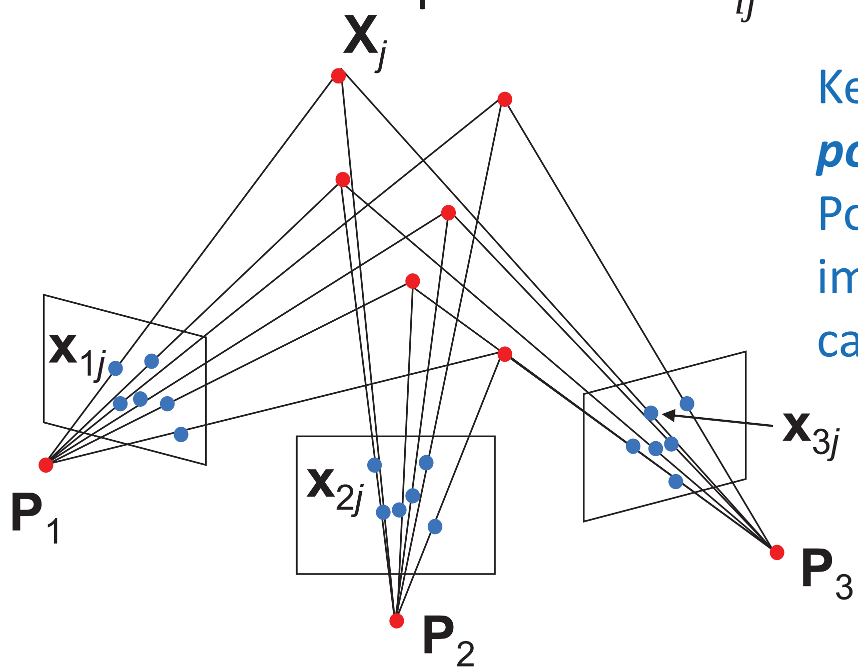


Драконъ, видимый подъ различными углами зрѣнія

По гравюрѣ на мѣди изъ „Oculus artificialis teledioptricus“ Цана. 1702 года.

Structure from motion

- Given: *m* images of *n* fixed 3D points
    - $\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, n$
- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$



Key assumption: ***matching points***
Possible only for textured image areas where features can be detected!

Sparse point cloud reconstruction

Revisiting triangulation

How would you reconstruct 3D points?

Left image

Right image

How would you reconstruct 3D points?

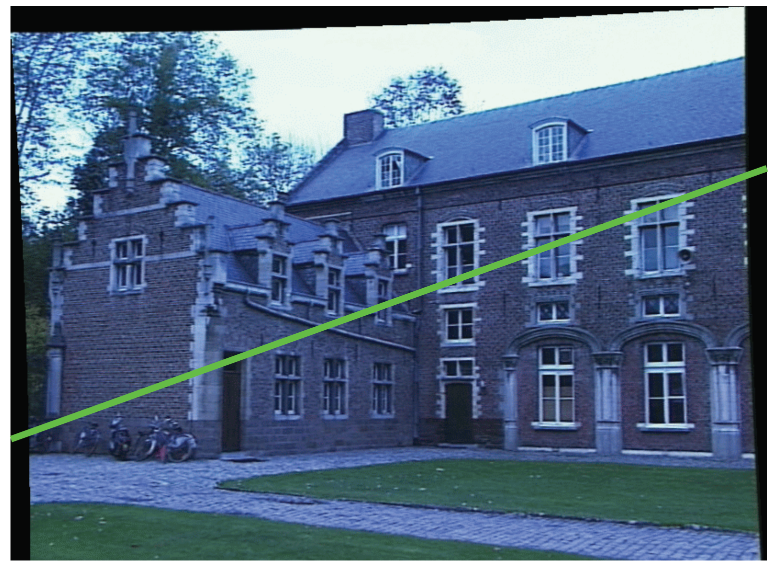
Left image

Right image

1. Select point in one image (how?)

How would you reconstruct 3D points?

Left image



Right image

1. Select point in one image (how?)
2. Form epipolar line for that point in second image (how?)

How would you reconstruct 3D points?

Left image

Right image

1. Select point in one image (how?)
2. Form epipolar line for that point in second image (how?)
3. Find matching point along line (how?)

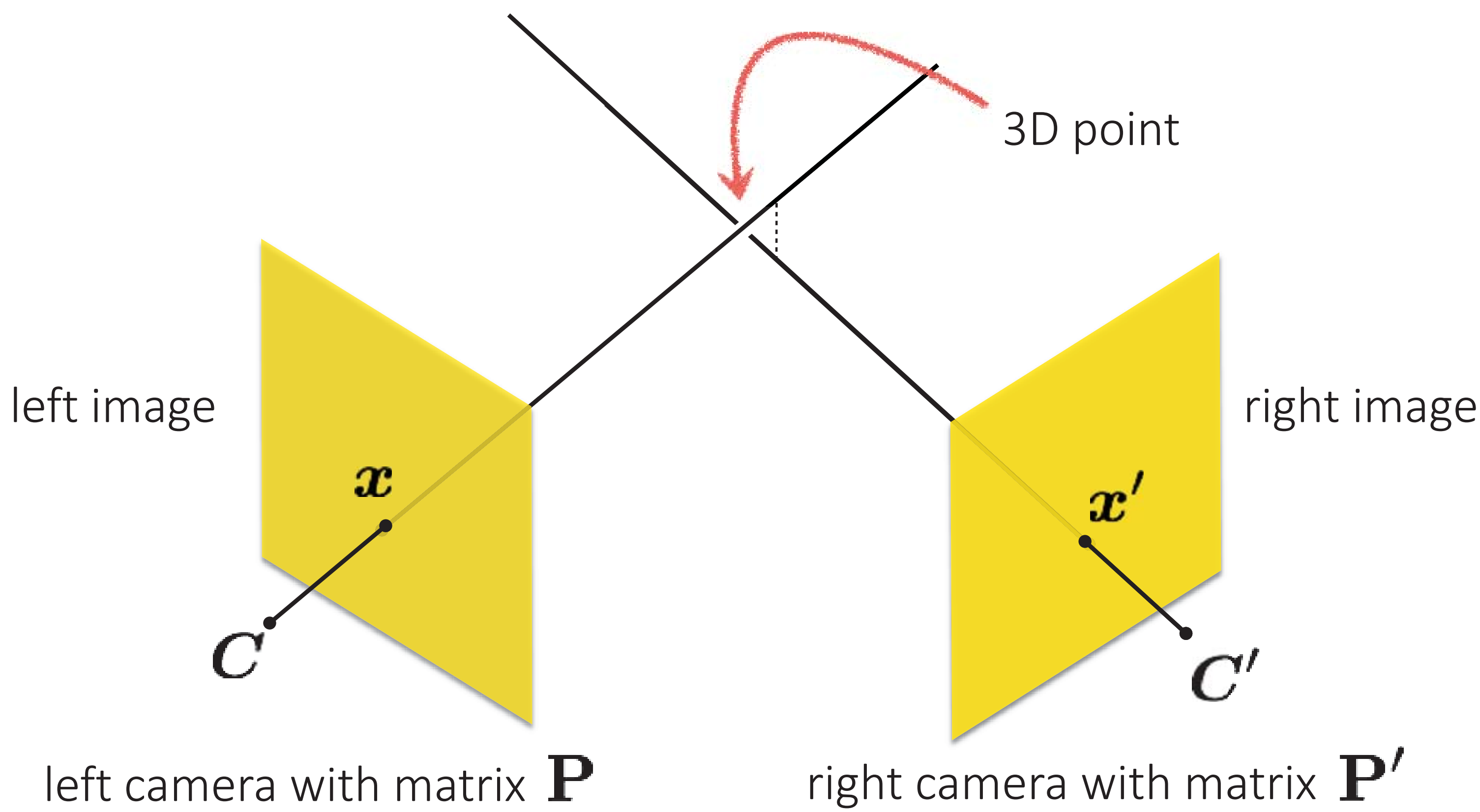
How would you reconstruct 3D points?

Left image

Right image

1. Select point in one image (how?)
2. Form epipolar line for that point in second image (how?)
3. Find matching point along line (how?)
4. Perform triangulation (how?)

Triangulation



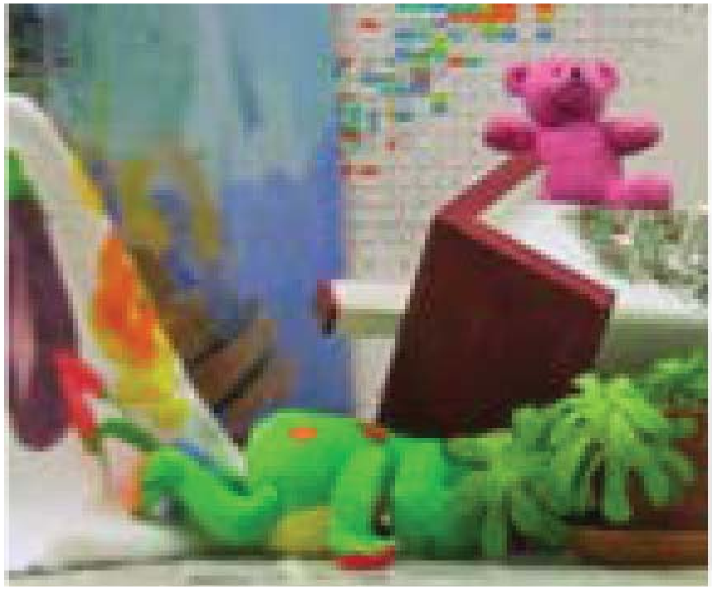
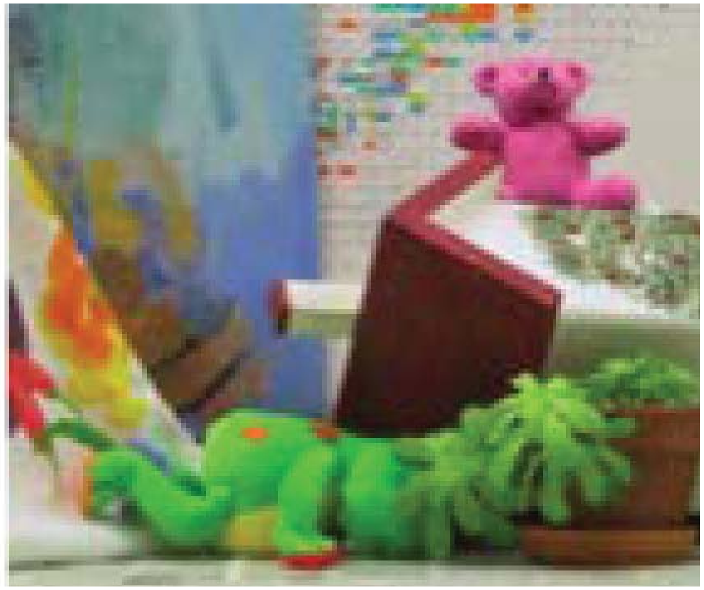
How would you reconstruct 3D points?

Left image

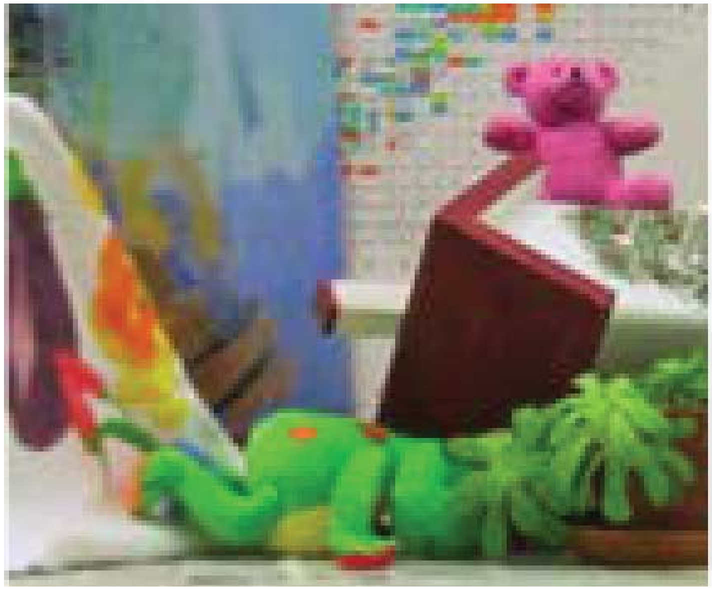
Right image

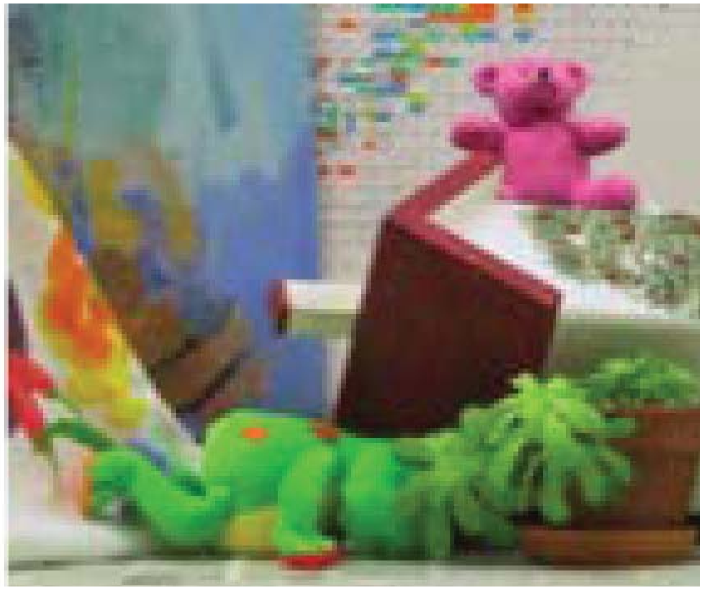
1. Select point in one image (how?)
2. Form epipolar line for that point in second image (how?)
3. Find matching point along line (how?)
4. Perform triangulation (how?)

Stereo rectification

What’s different between these two images?





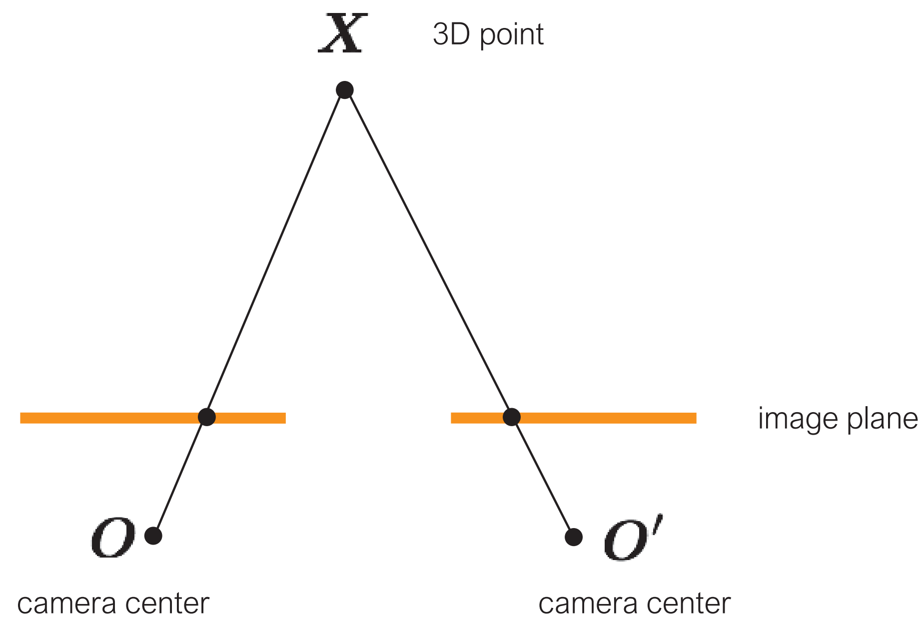
Objects that are close move more or less?

The amount of horizontal movement is
inversely proportional to …

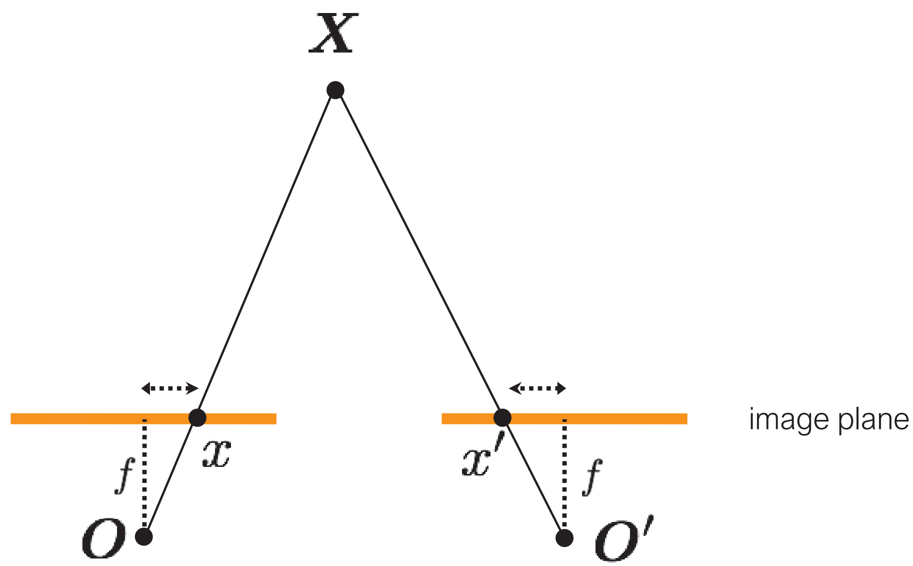
The amount of horizontal movement is inversely proportional to …

… the distance from the camera.

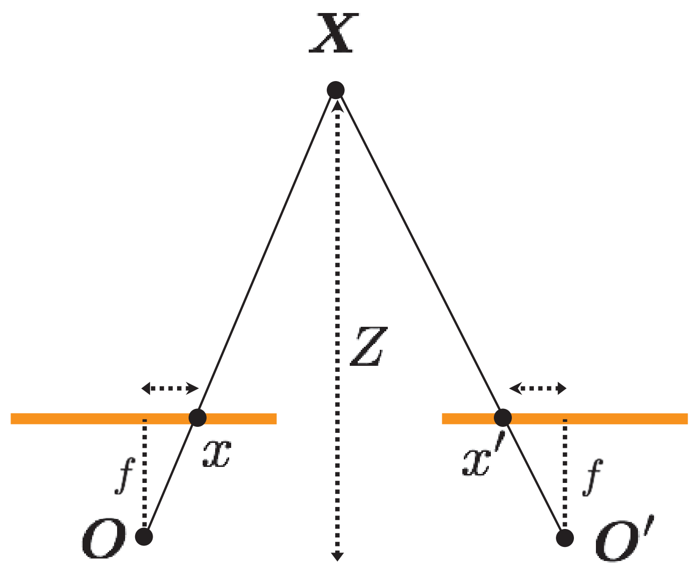
More formally…



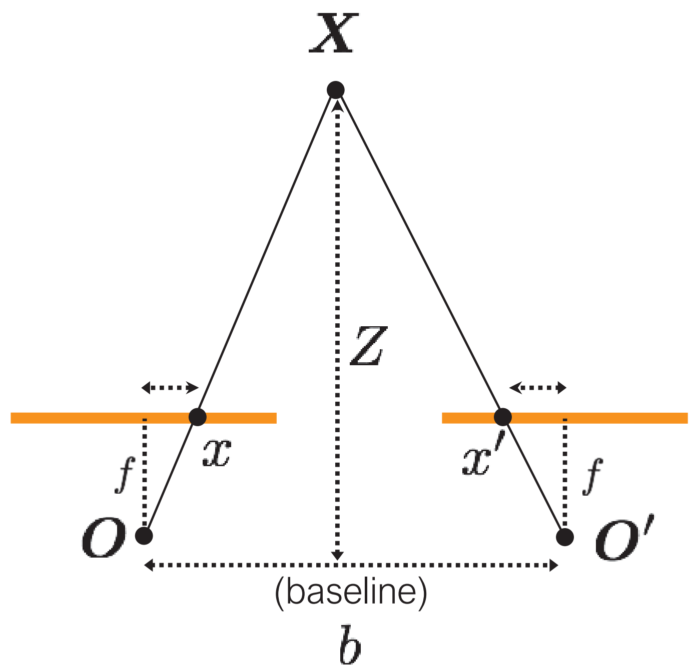
X
3D point
image plane
O
camera center
O'
camera center



X
x
x'
f
f
O
O'
image plane

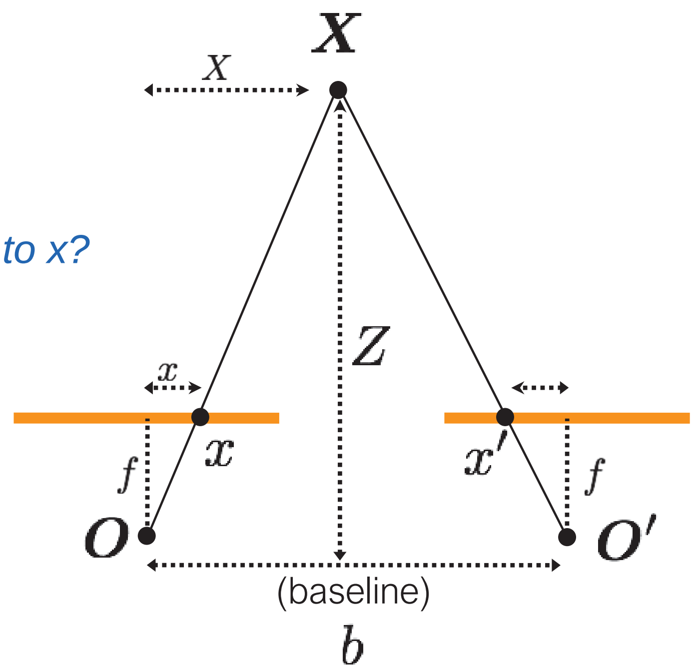


X
Z
x
x′
f
f
O
O′

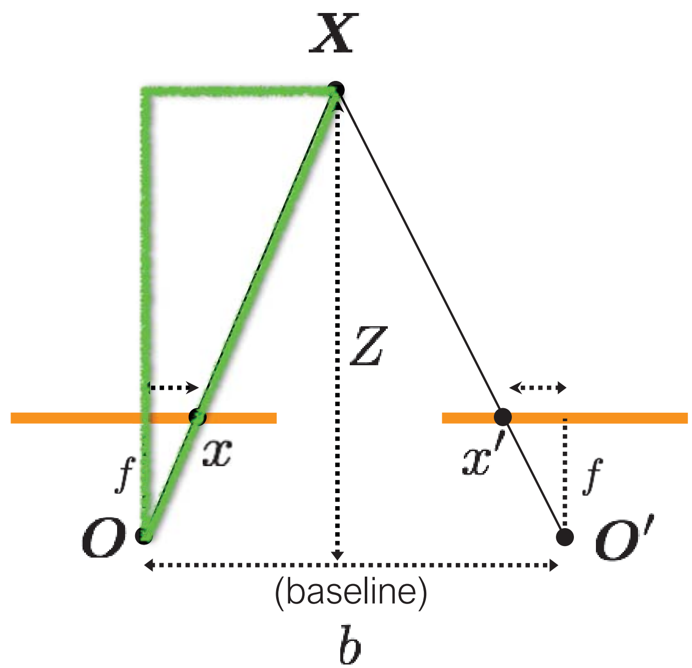


X
Z
x
x′
f
f
O
O′
(baseline)
b

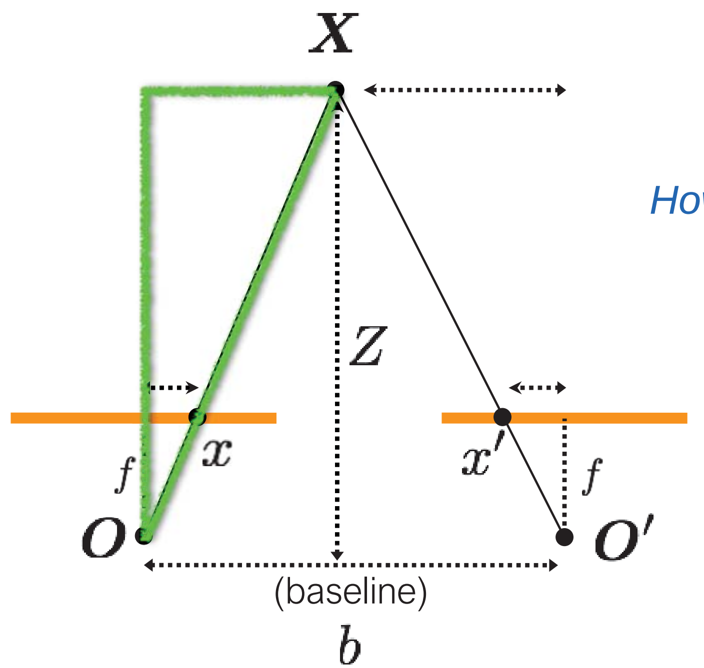
How is X related to x?



$$\frac{X}{Z} = \frac{x}{f}$$

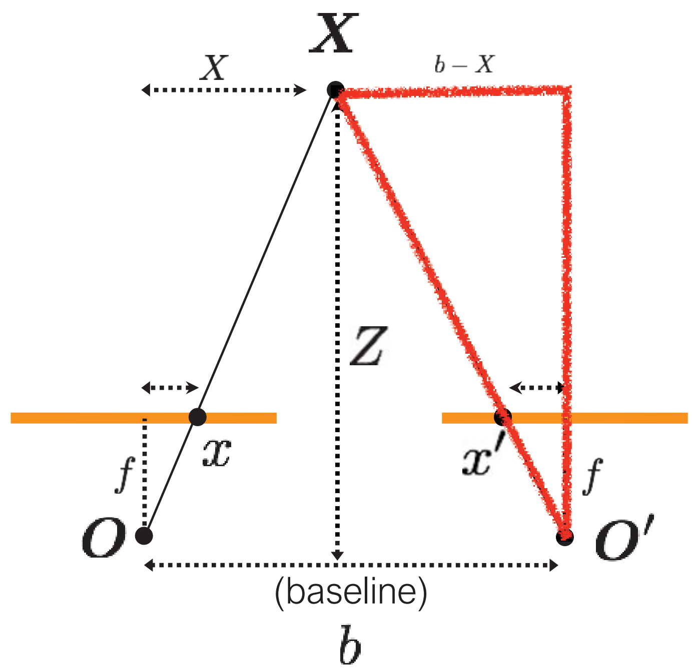


$$\frac{X}{Z} = \frac{x}{f}$$



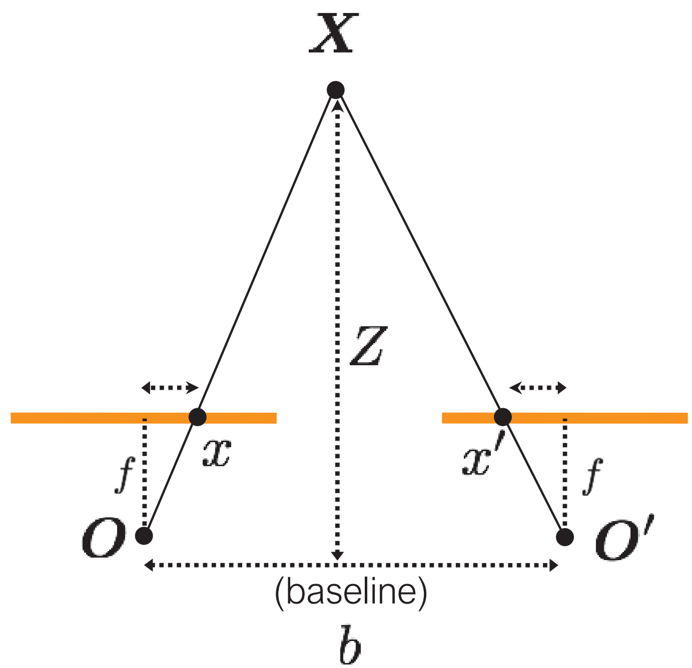
How is X related to x'?

$$\frac{X}{Z} = \frac{x}{f}$$



$$\frac{b - X}{Z} = \frac{x'}{f}$$

$$\frac{X}{Z} = \frac{x}{f}$$



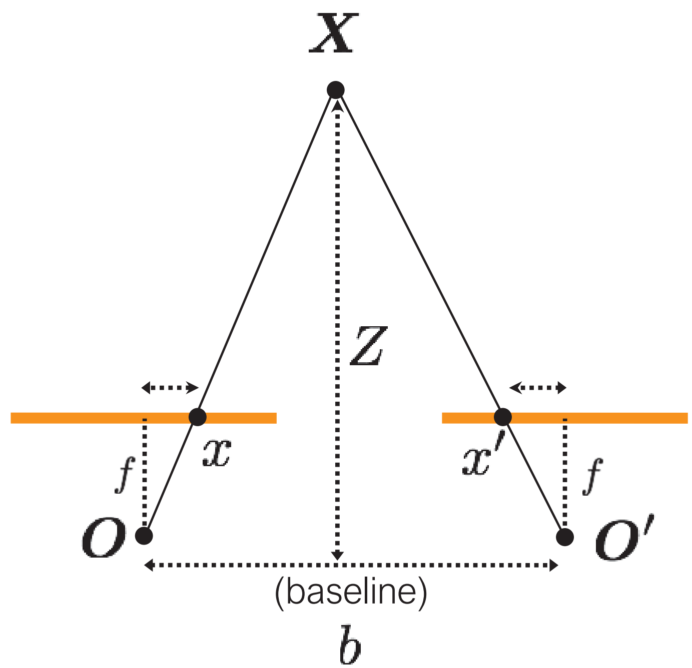
$$\frac{b - X}{Z} = \frac{x'}{f}$$

Disparity

$$d = x - x'$$ (wrt to camera origin of image plane)

$$= \frac{bf}{Z}$$

$$\frac{X}{Z} = \frac{x}{f}$$



$$\frac{b - X}{Z} = \frac{x'}{f}$$

Disparity

$$d = x - x'$$

$$= \frac{bf}{Z}$$

inversely proportional to depth

Real-time stereo sensing

Nomad robot searches for meteorites in Antartica

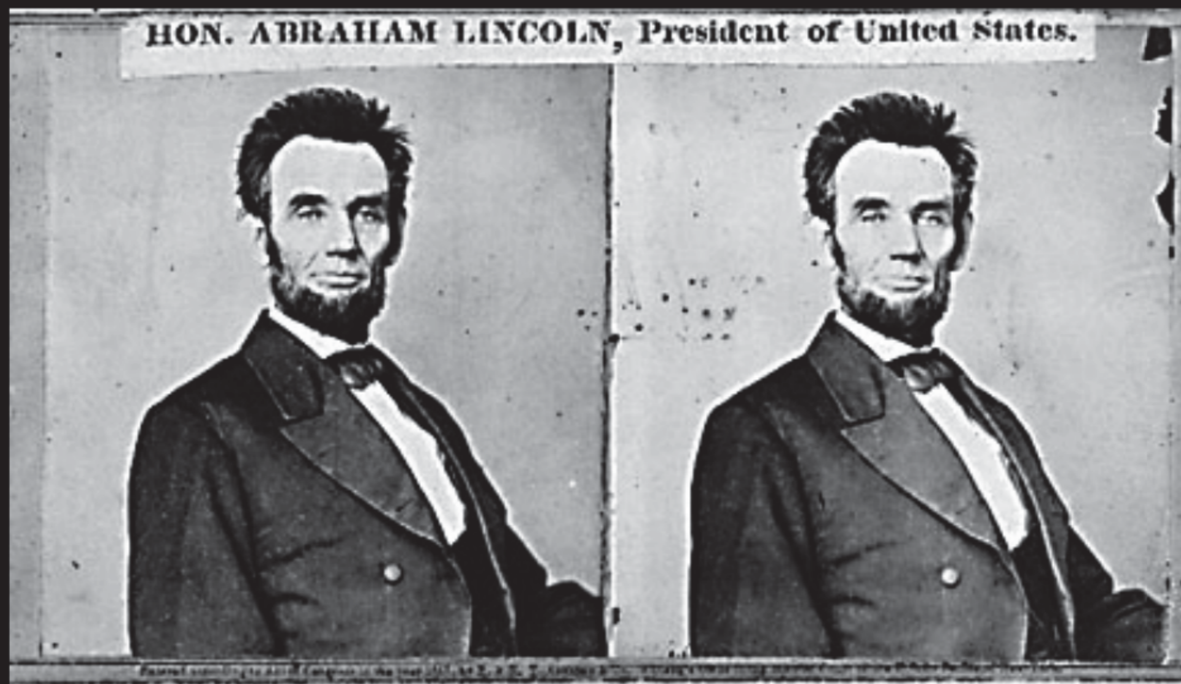
http://www.frc.ri.cmu.edu/projects/meteorobot/index.html

Subaru
Eyesight system

Pre-collision
braking

What other vision system uses disparity for depth sensing?

Stereoscopes: A 19th Century Pastime

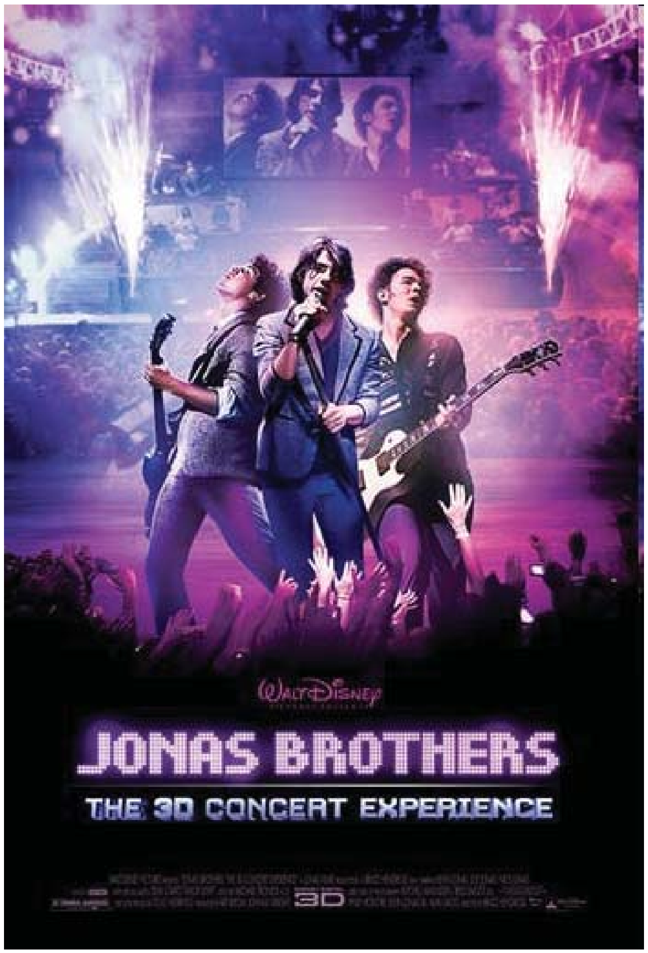

HON. ABRAHAM LINCOLN, President of United States.

Public Library, Stereoscopic Looking Room, Chicago, by Phillips, 1923

Teesta suspension bridge-Darjeeling, India

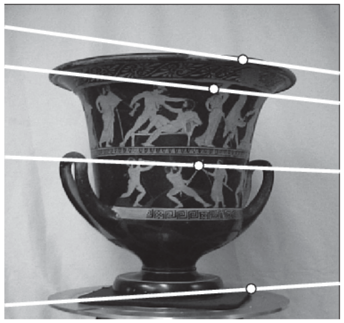
Mark Twain at Pool Table", no date, UCR Museum of Photography

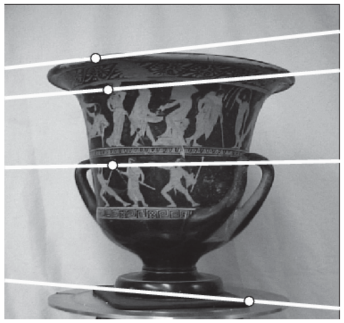
This is how 3D movies work



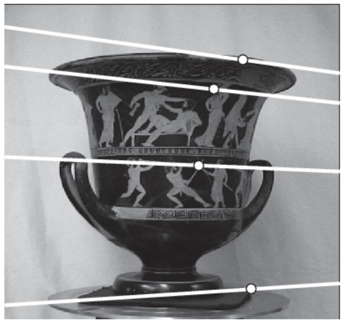
Is disparity the only depth cue the human visual system uses?

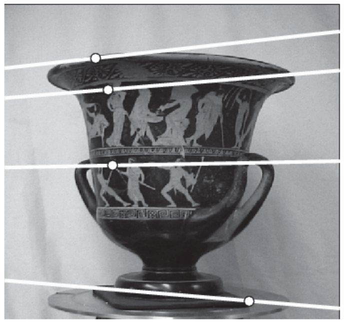
So can I compute depth from any two images of the same object?





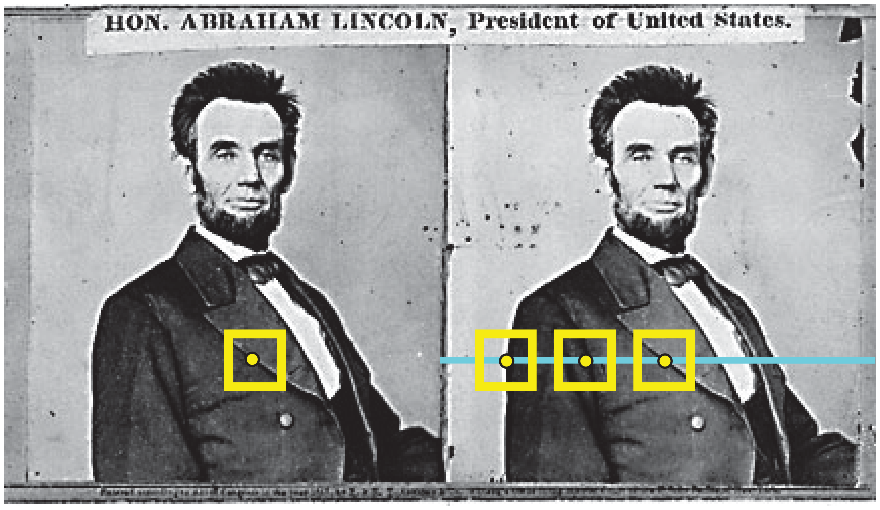
So can I compute depth from any two images of the same object?





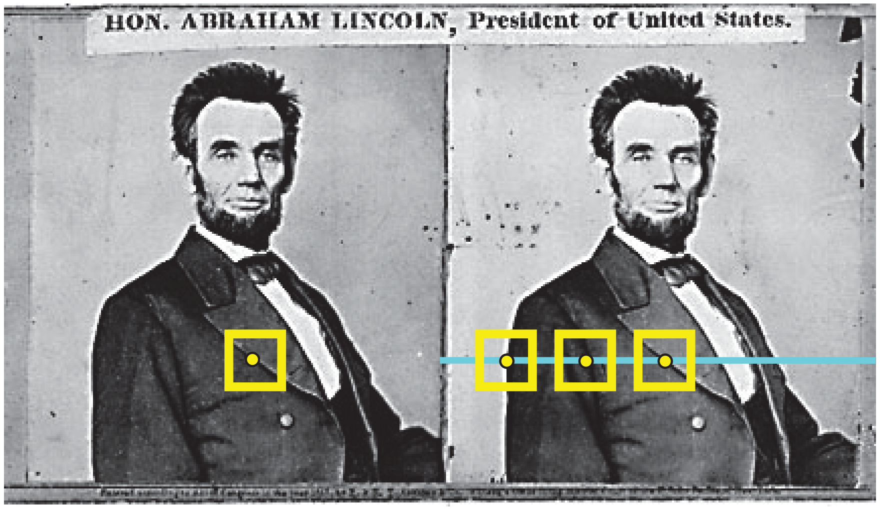
1. Need sufficient baseline

2. Images need to be ‘rectified’ first (make epipolar lines horizontal)

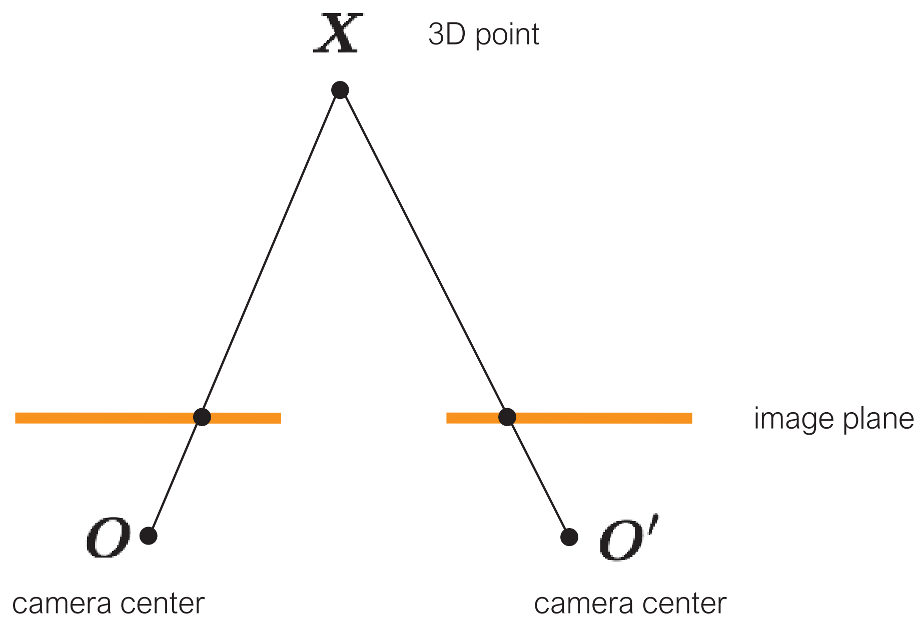


1. Rectify images
   (make epipolar lines horizontal)
2. For each pixel
   a. Find epipolar line
   b. Scan line for best match
   c. Compute depth from disparity

$$Z = \frac{bf}{d}$$

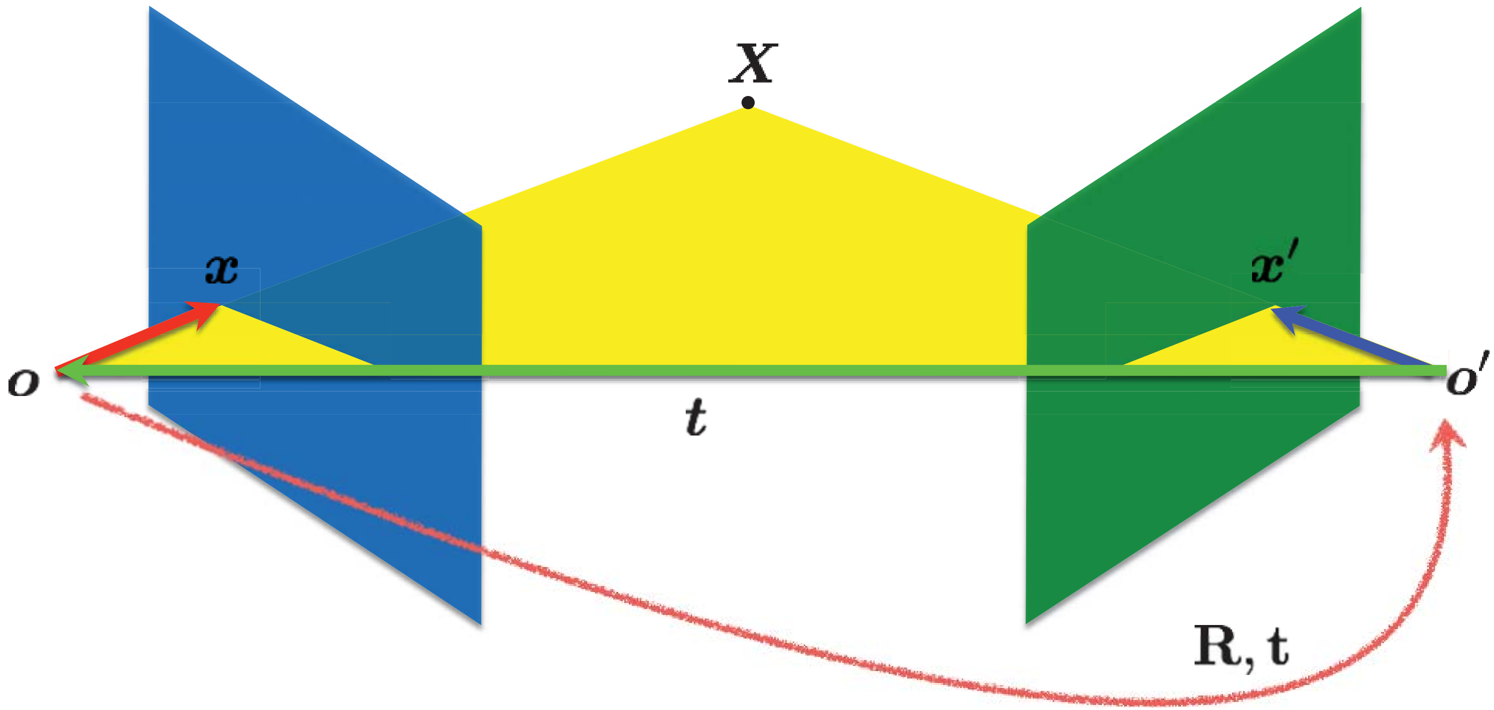


How can you make the epipolar lines horizontal?



What's special about these two cameras?

They have the same image plane
No rotation, only horizontal translation



$$\boldsymbol{x}' = \mathbf{R}(\boldsymbol{x} - \boldsymbol{t})$$

Recall: Essential matrix

$$E = R[t_\times]$$

Longuet-Higgins equation

$$\boldsymbol{x}'^\top \mathbf{E} \boldsymbol{x} = 0$$

Epipolar lines

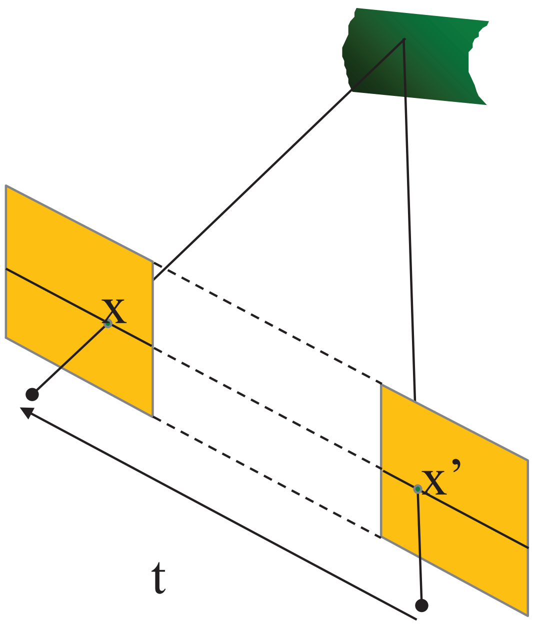
$$\boldsymbol{x}^\top \boldsymbol{l} = 0 \qquad \boldsymbol{x}'^\top \boldsymbol{l}' = 0$$

$$\boldsymbol{l}' = \mathbf{E}\boldsymbol{x} \qquad \boldsymbol{l} = \mathbf{E}^T \boldsymbol{x}'$$

Epipoles

$$\boldsymbol{e}'^\top \mathbf{E} = \mathbf{0} \qquad \mathbf{E}\boldsymbol{e} = \mathbf{0}$$

When are epipolar lines horizontal?



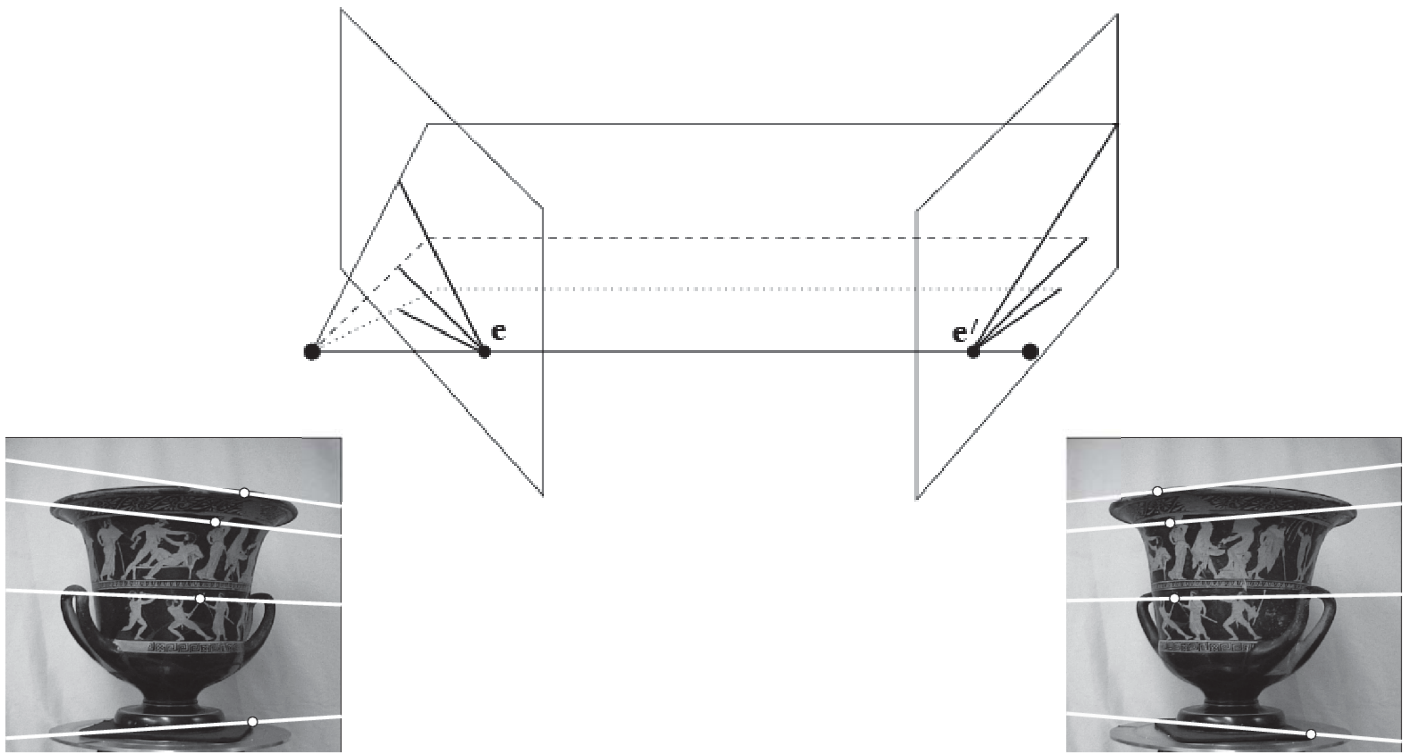
When this relationship holds:

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

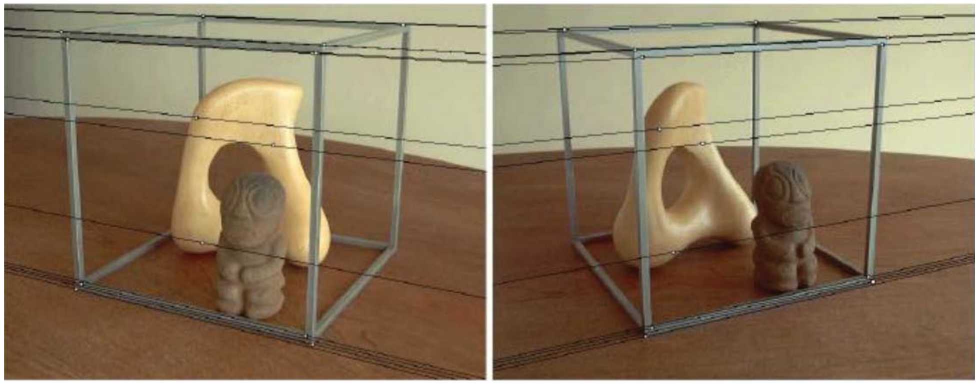
Prove as part of Ex4:

$$\boldsymbol{l} = \boldsymbol{E}\boldsymbol{x}$$

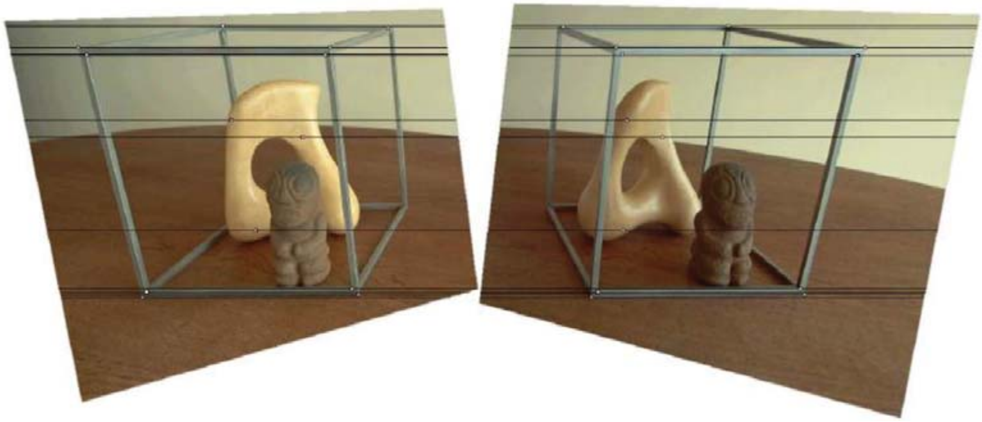
Is an horizontal line for every point x

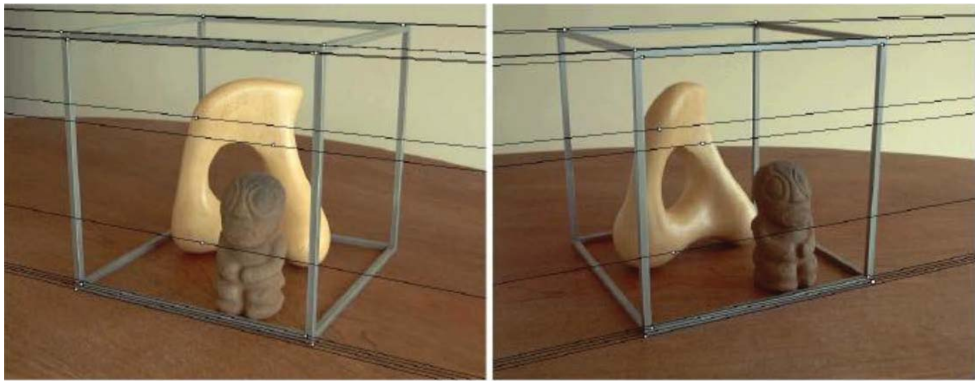


It’s hard to make the image planes exactly parallel

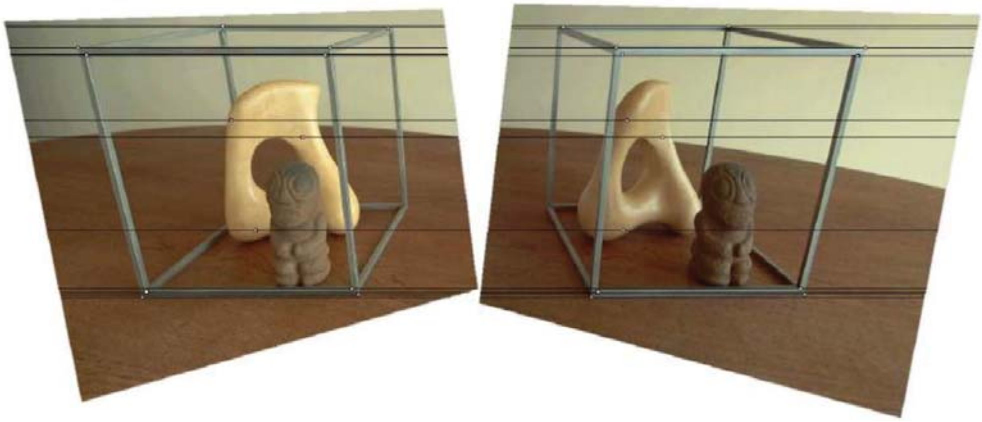


How can you make the epipolar lines horizontal?

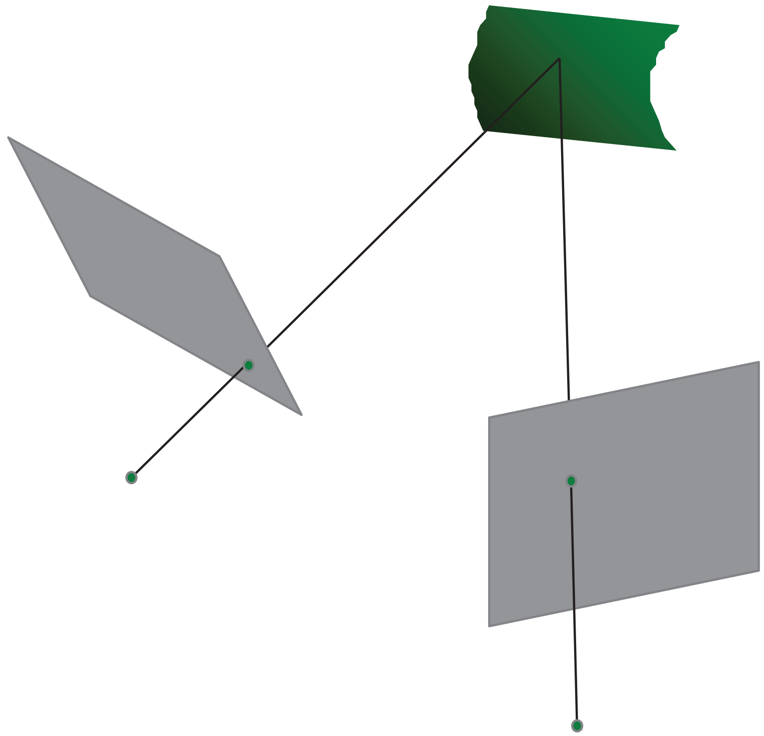




Use stereo rectification?

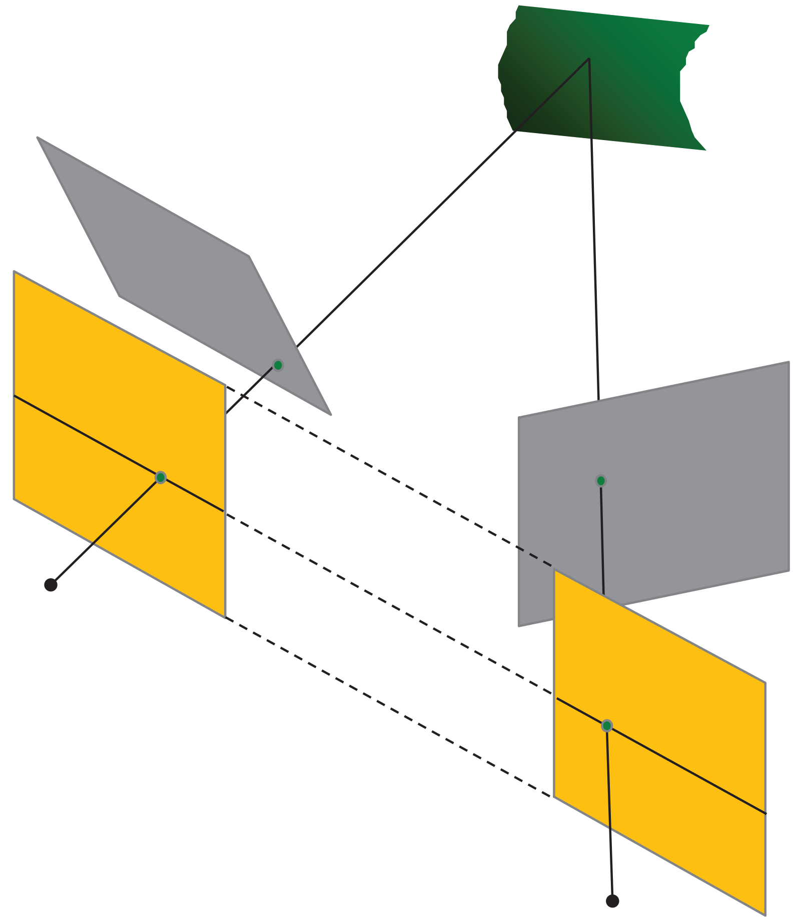


What is stereo rectification?



What is stereo rectification?

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

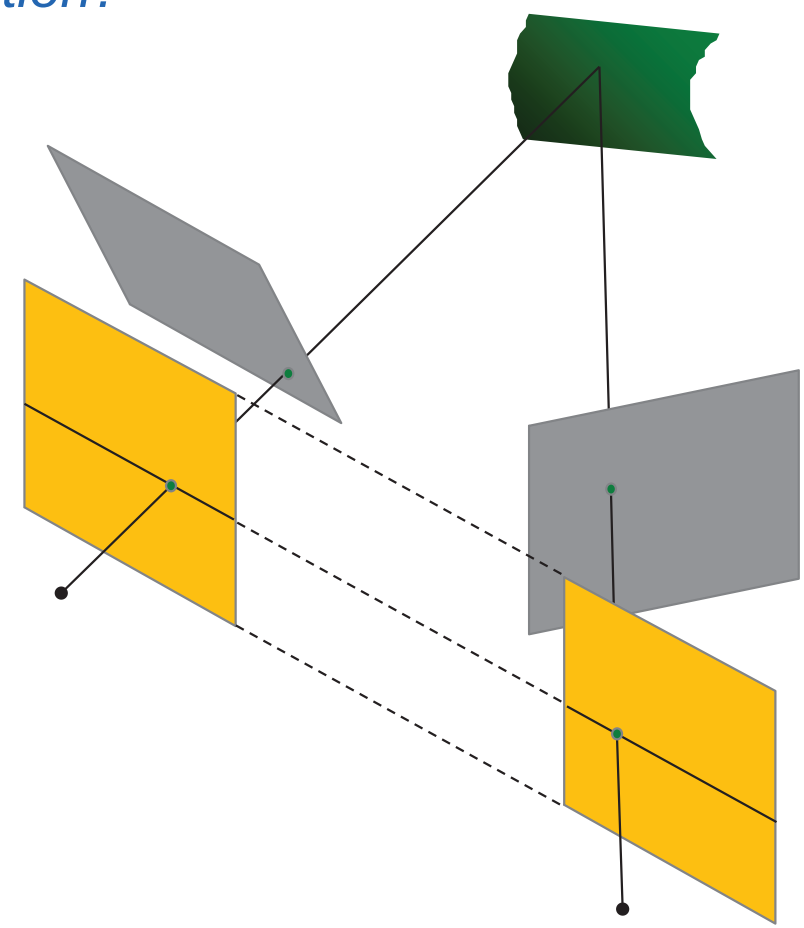


How can you do this?

What is stereo rectification?

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

Need two homographies (3x3 transform), one for each input image reprojection



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

What is stereo rectification?

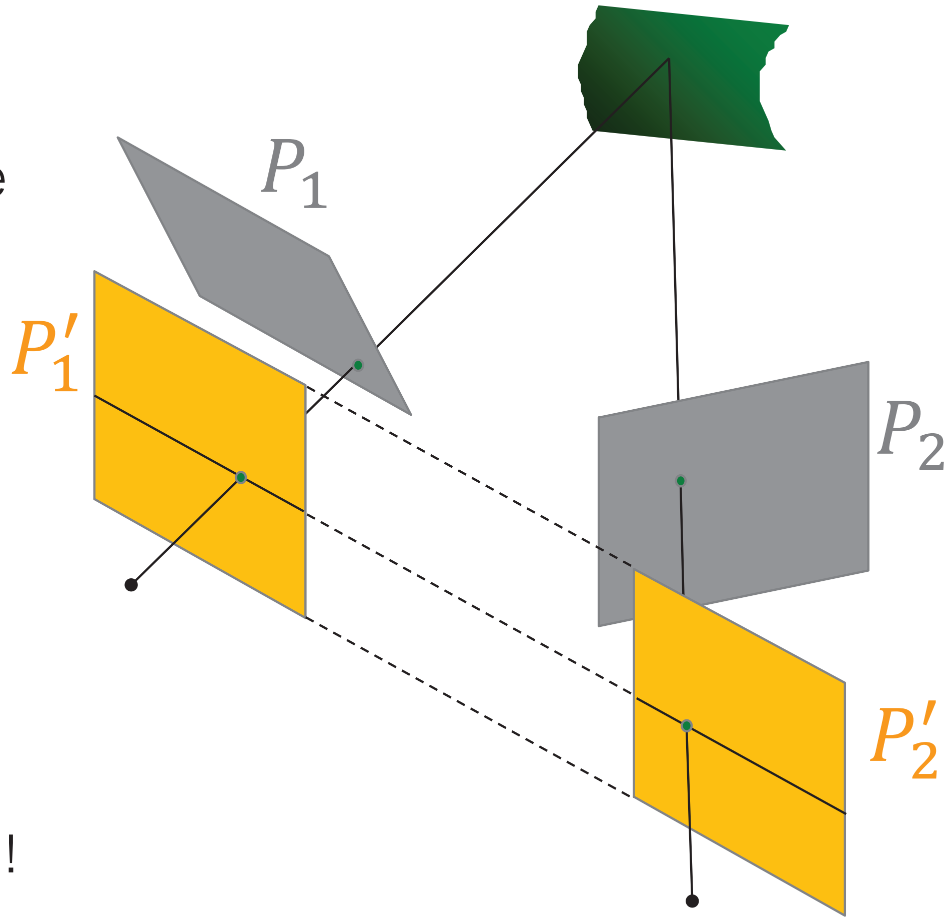
Reproject image planes onto a common plane parallel to the line between camera centers

$$P_1 = [\mathrm{I}|0]$$
$$P_2 = [R \quad |\mathrm{T}]$$
$$x = P_1 X$$
$$x_2 = P_2 X$$

There is no 3x3 mapping:

$x_2 \neq Hx$

Mapping depends on 3D position!



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

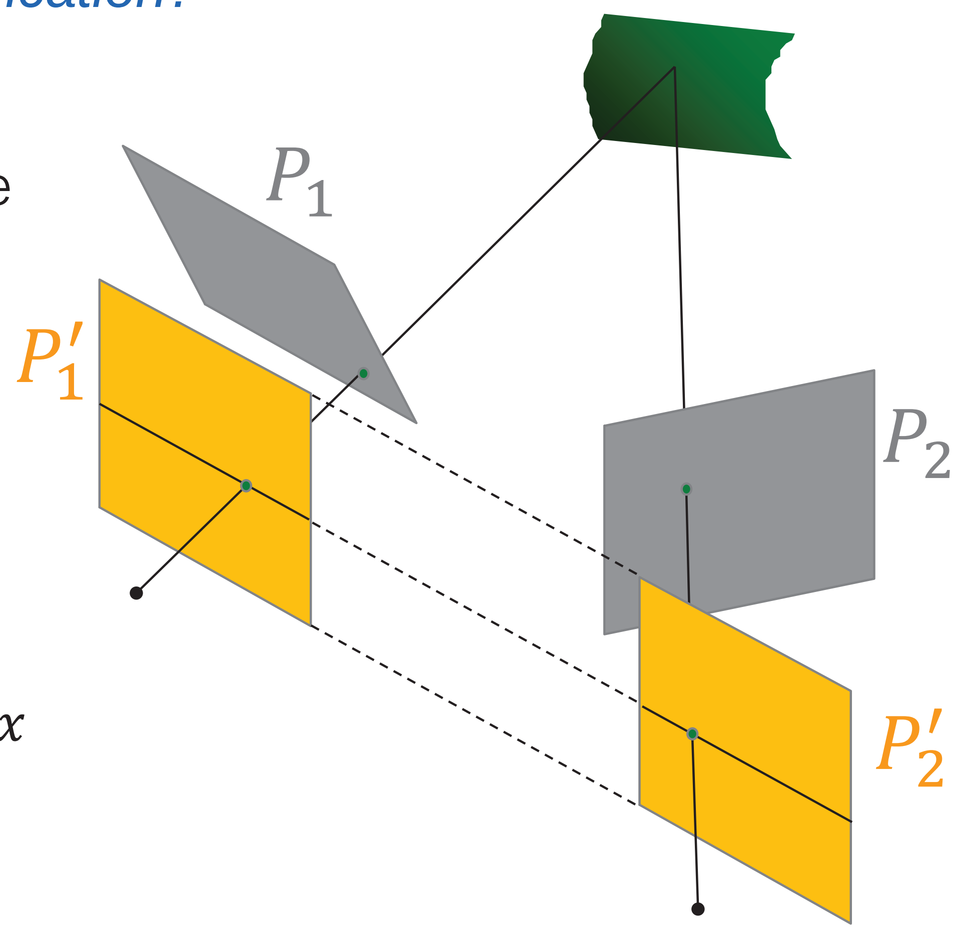
What is stereo rectification?

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

$$P_1 = [\mathrm{I}|0]$$
$$P_1' = [R'|0]$$

$$x = P_1 X$$
$$x' = P_1' X$$

Homography mapping $x' = R'x$



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

Stereo Rectification

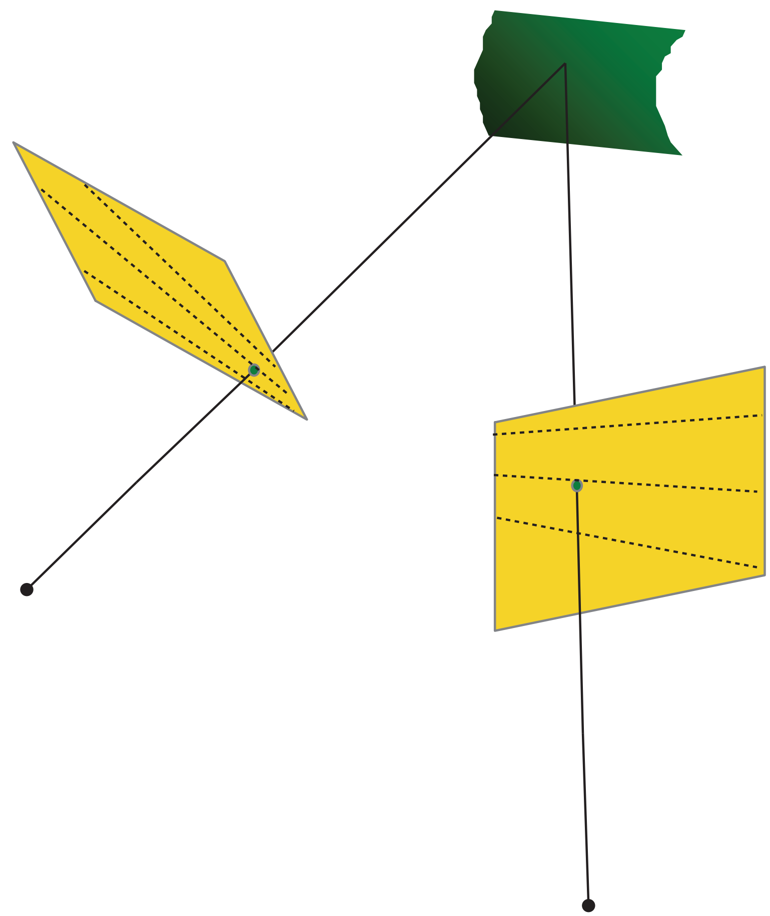
1. **Rotate** the right camera by $\mathbf{R}^{\mathsf{T}}$
   (aligns camera coordinate system orientation only)

2. Rotate (**rectify**) the left camera so that the epipole
   is at infinity

3. Rotate (**rectify**) the right camera so that the epipole
   is at infinity

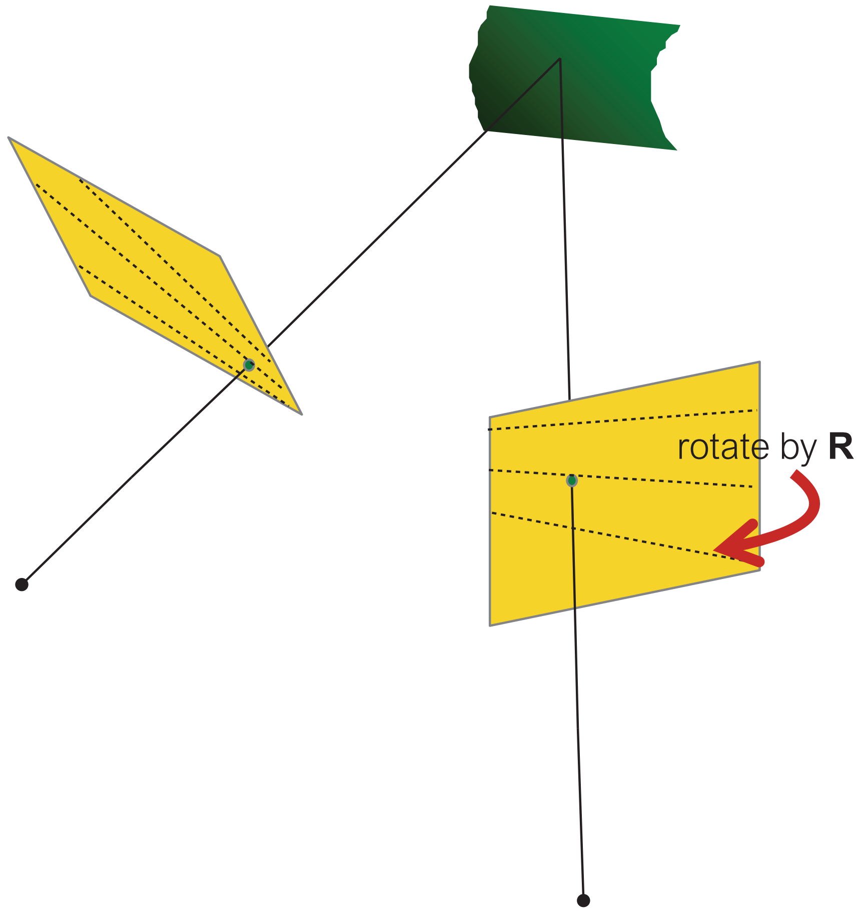
4. Adjust the **scale**

Stereo Rectification:



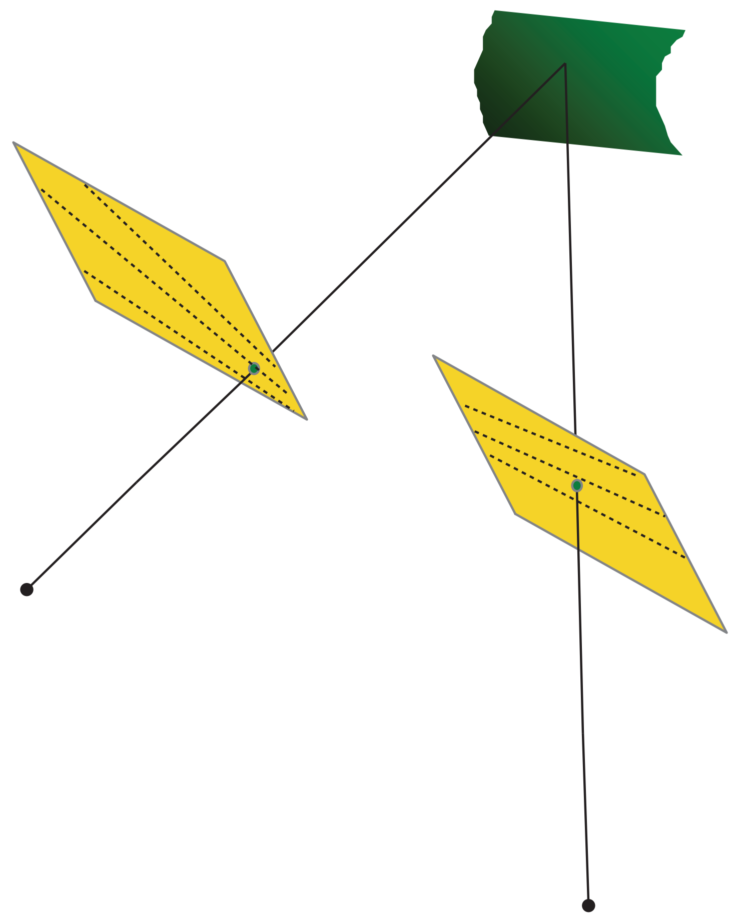
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R_{rect}}$
4. Scale both images by **H**

Stereo Rectification:



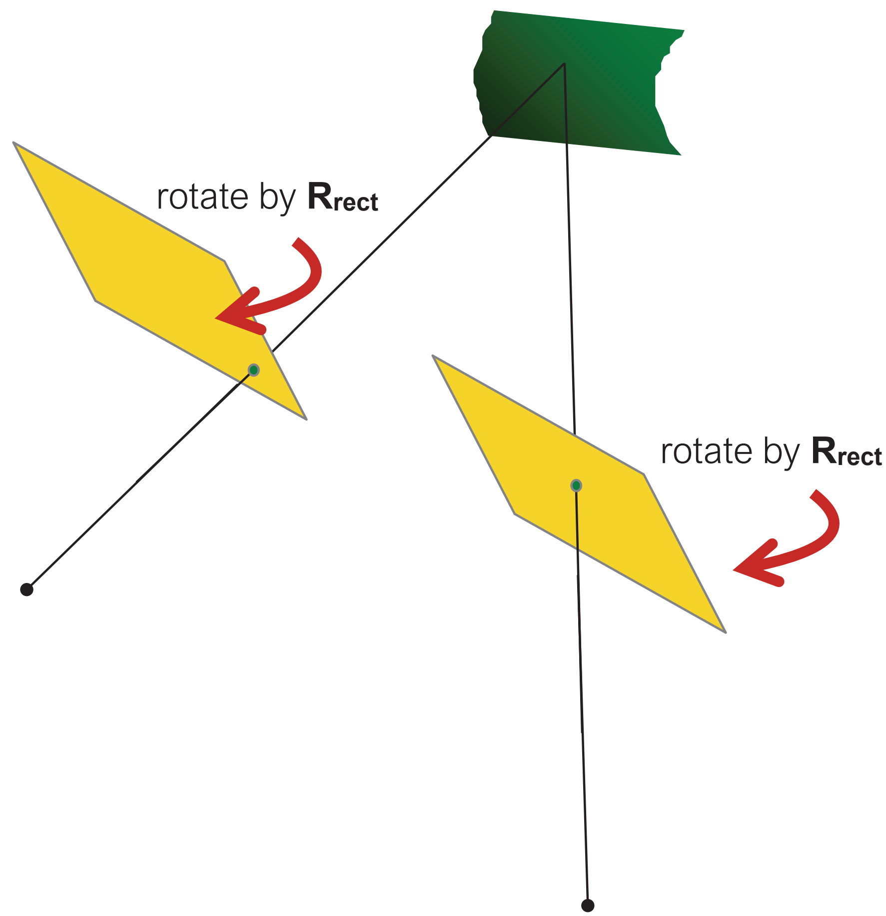
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R}_{\mathbf{rect}}$
4. Scale both images by **H**

Stereo Rectification:



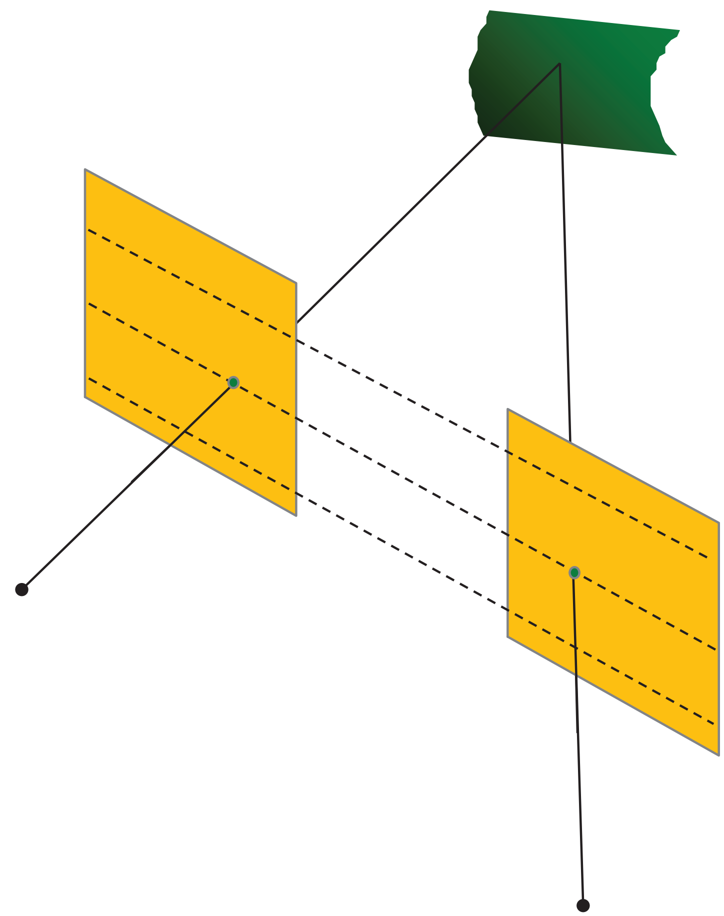
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R_{rect}}$
4. Scale both images by **H**

Stereo Rectification:



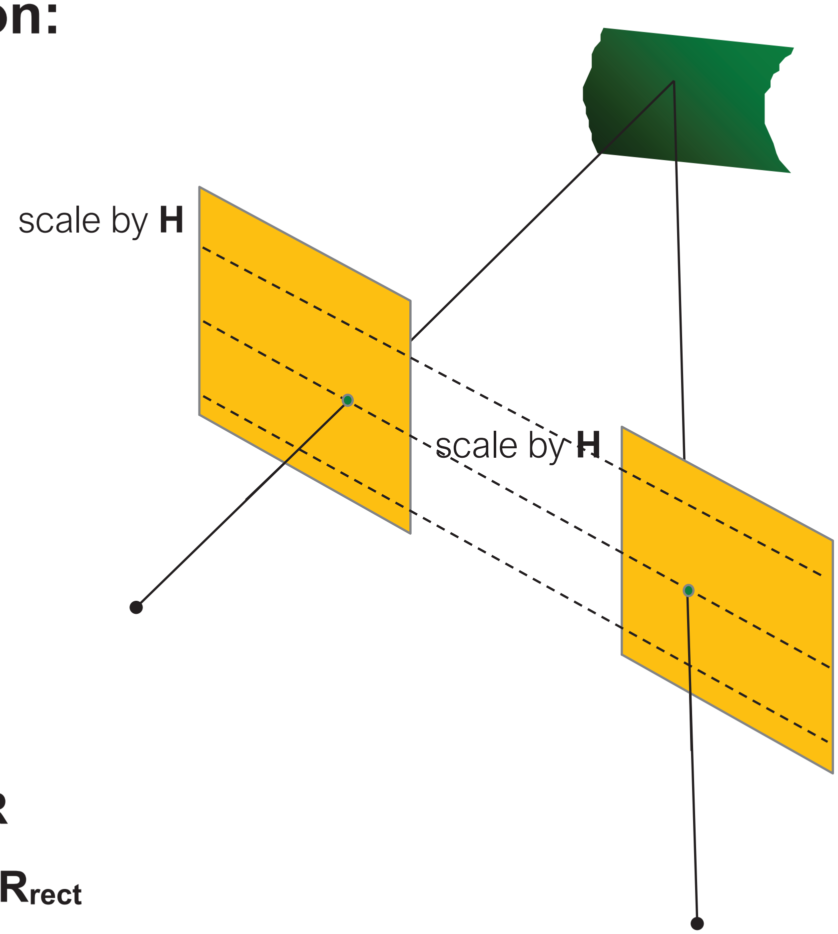
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R}_{rect}$
4. Scale both images by **H**

Stereo Rectification:



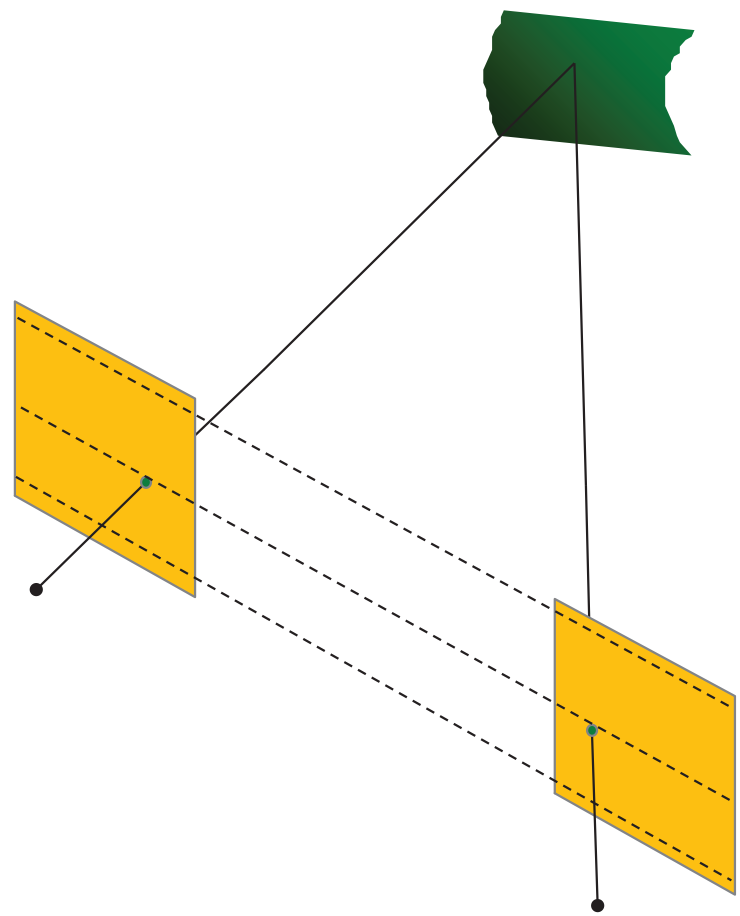
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R_{rect}}$
4. Scale both images by **H**

Stereo Rectification:



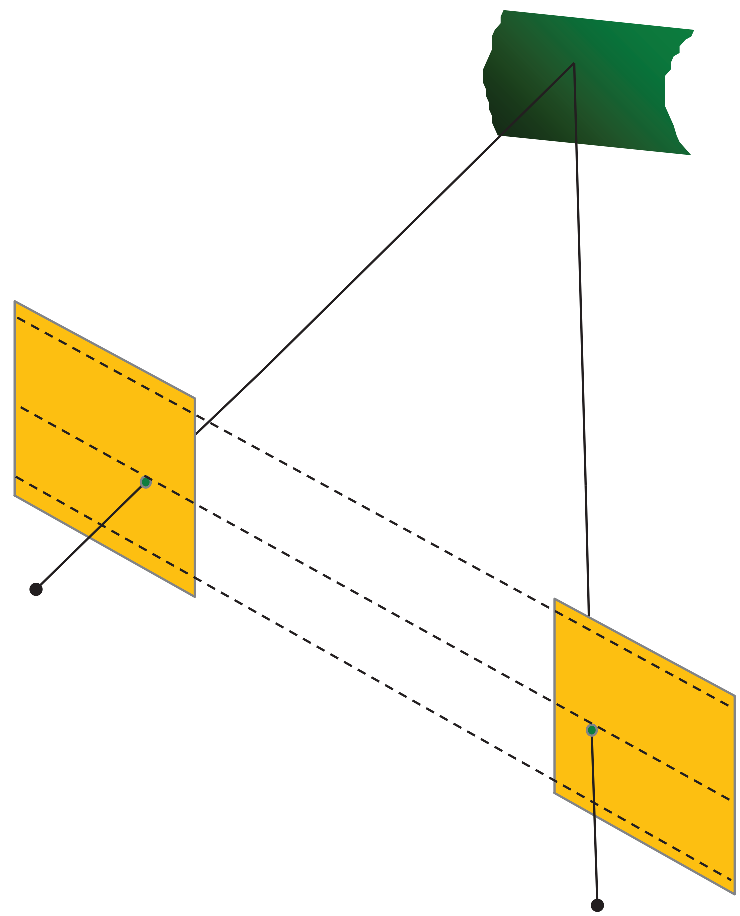
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R_{rect}}$
4. Scale both images by **H**

Stereo Rectification:



1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R_{rect}}$
4. Scale both images by **H**

Stereo Rectification:



1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R_{rect}}$
4. Scale both images by **H**

Step 1: Compute **E** to get **R**

$$E = R[t_\times]$$

SVD: $\mathbf{E} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ Let $\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

We get FOUR solutions:

$$\mathbf{E} = [\mathbf{R}|\mathbf{T}]$$

$\mathbf{R}_1 = \mathbf{U}\mathbf{W}\mathbf{V}^\top$ $\mathbf{R}_2 = \mathbf{U}\mathbf{W}^\top\mathbf{V}^\top$

two possible rotations

$\mathbf{T}_1 = U_3$ $\mathbf{T}_2 = -U_3$

two possible translations

Recall: Essential matrix

$$E = R[t_{\times}]$$

Longuet-Higgins equation

$$\boldsymbol{x}'^{\top}\mathbf{E}\boldsymbol{x} = 0$$

Epipolar lines

$$\boldsymbol{x}^{\top}\boldsymbol{l} = 0 \qquad \boldsymbol{x}'^{\top}\boldsymbol{l}' = 0$$

$$\boldsymbol{l}' = \mathbf{E}\boldsymbol{x} \qquad \boldsymbol{l} = \mathbf{E}^{T}\boldsymbol{x}'$$

Epipoles

$$\boldsymbol{e}'^{\top}\mathbf{E} = \mathbf{0} \qquad \mathbf{E}\boldsymbol{e} = \mathbf{0}$$

What do we achieve with this rotation?

$$E = R[t_\times]$$

$$x'^T E x = 0$$

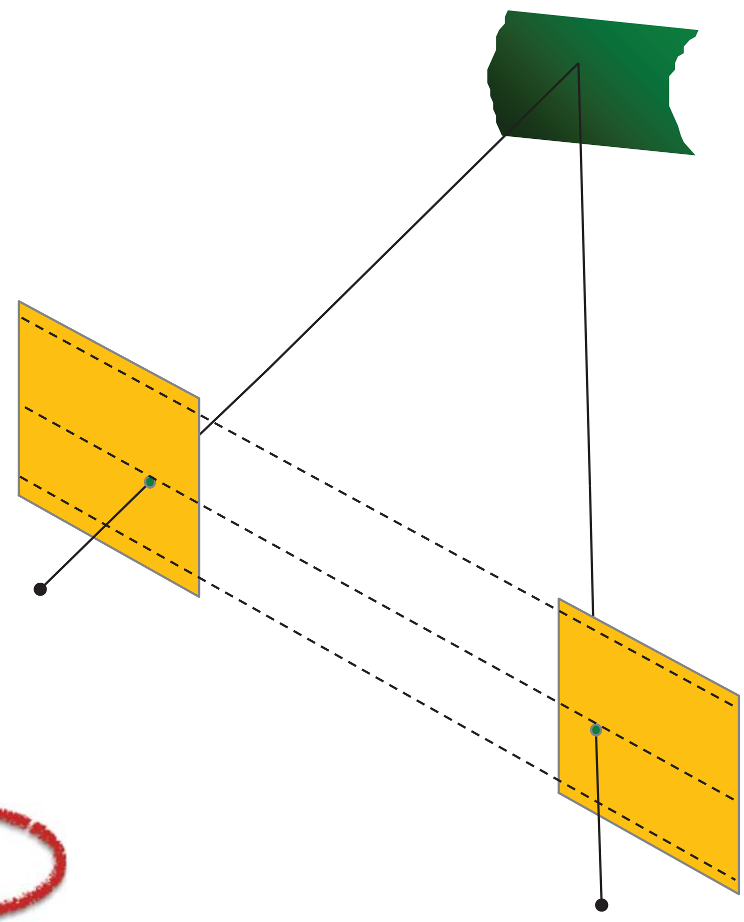
Define: $\widetilde{x'} = R^T x'$

$$\widetilde{x'^T}\tilde{E}x = 0, \quad \tilde{E} = [t_\times]$$

What's the epipole? e=e'=t

We want e=[1,0,0]

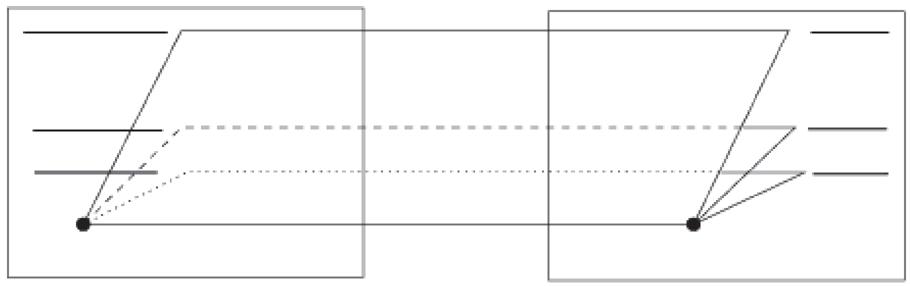
Stereo Rectification:

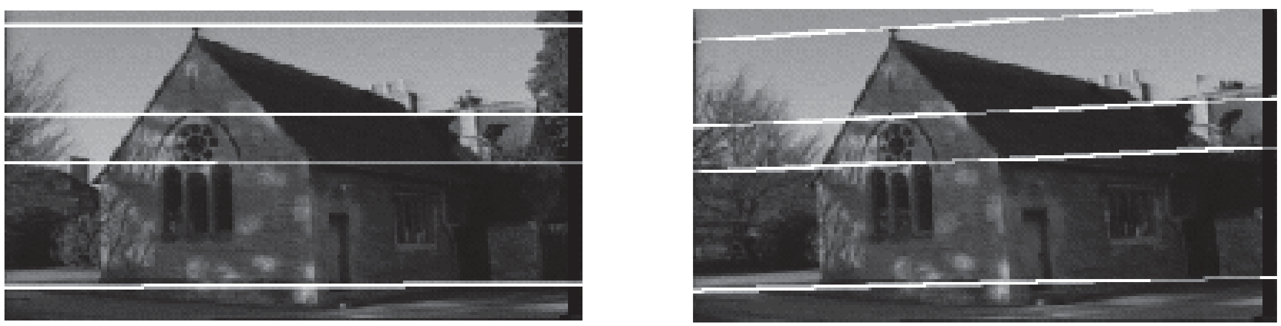


1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R}_{\mathbf{rect}}$
4. Scale both images by **H**

When do epipolar lines become horizontal?

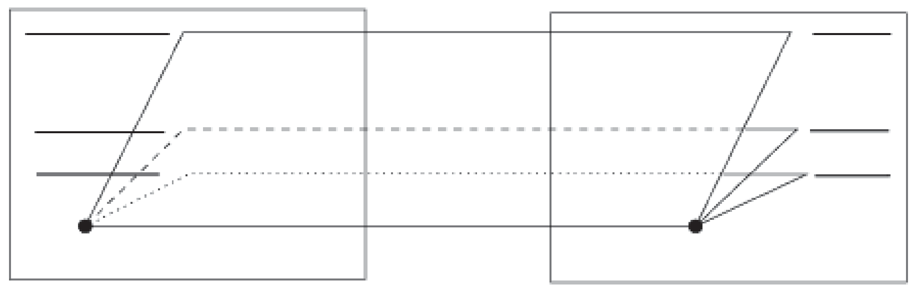
Parallel cameras

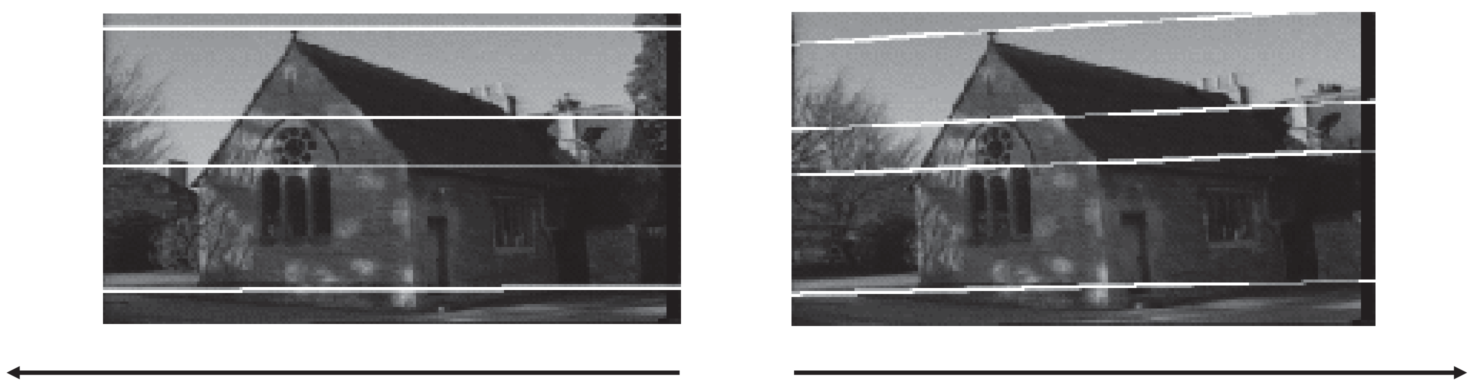




Where is the epipole?

Parallel cameras





epipole at infinity

Setting the epipole to infinity

(Building **R**$_{\text{rect}}$ from **e**)

Let $$R_{\text{rect}} = \begin{bmatrix} \boldsymbol{r}_1^\top \\ \boldsymbol{r}_2^\top \\ \boldsymbol{r}_3^\top \end{bmatrix}$$

Given: epipole **e** (translation from **E)**

$$\boldsymbol{r}_1 = \boldsymbol{e}_1 = \frac{T}{||T||}$$

epipole coincides with translation vector

$$\boldsymbol{r}_2 = \frac{1}{\sqrt{T_x^2 + T_y^2}} \begin{bmatrix} -T_y & T_x & 0 \end{bmatrix}$$

cross product of e and the direction vector of the optical axis

$$\boldsymbol{r}_3 = \boldsymbol{r}_1 \times \boldsymbol{r}_2$$

orthogonal vector

If $\boldsymbol{r}_1 = \boldsymbol{e}_1 = \frac{T}{||T||}$ and $\boldsymbol{r}_2$ $\boldsymbol{r}_3$ orthogonal

then $R_{\text{rect}}\boldsymbol{e}_1 = \begin{bmatrix} \boldsymbol{r}_1^\top \boldsymbol{e}_1 \\ \boldsymbol{r}_2^\top \boldsymbol{e}_1 \\ \boldsymbol{r}_3^\top \boldsymbol{e}_1 \end{bmatrix} = \begin{bmatrix} ? \\ ? \\ ? \end{bmatrix}$

If $\quad \boldsymbol{r}_1 = \boldsymbol{e}_1 = \dfrac{T}{||T||} \quad$ and $\quad \boldsymbol{r}_2 \quad \boldsymbol{r}_3 \quad$ orthogonal

then $\quad R_{\text{rect}} \boldsymbol{e}_1 = \begin{bmatrix} \boldsymbol{r}_1^\top \boldsymbol{e}_1 \\ \boldsymbol{r}_2^\top \boldsymbol{e}_1 \\ \boldsymbol{r}_3^\top \boldsymbol{e}_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

Where is this point located on the image plane?

If $\boldsymbol{r}_1 = \boldsymbol{e}_1 = \frac{T}{||T||}$ and $\boldsymbol{r}_2$ $\boldsymbol{r}_3$ orthogonal

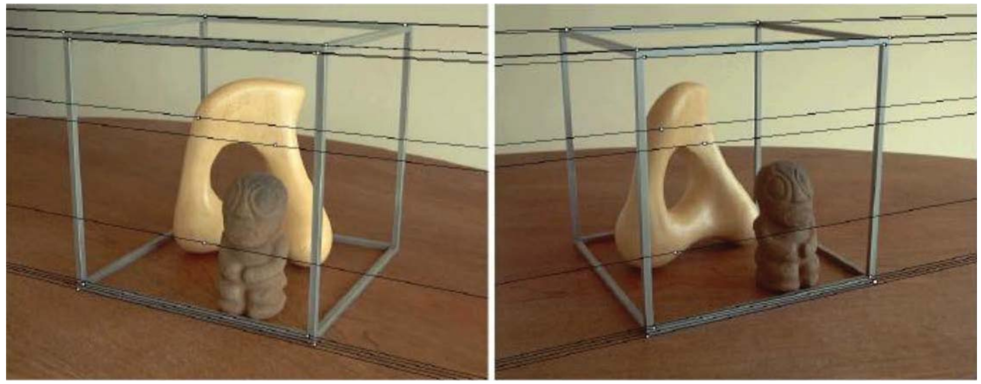
then $$R_{\text{rect}}\boldsymbol{e}_1 = \begin{bmatrix} \boldsymbol{r}_1^\top \boldsymbol{e}_1 \\ \boldsymbol{r}_2^\top \boldsymbol{e}_1 \\ \boldsymbol{r}_3^\top \boldsymbol{e}_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Where is this point located on the image plane?

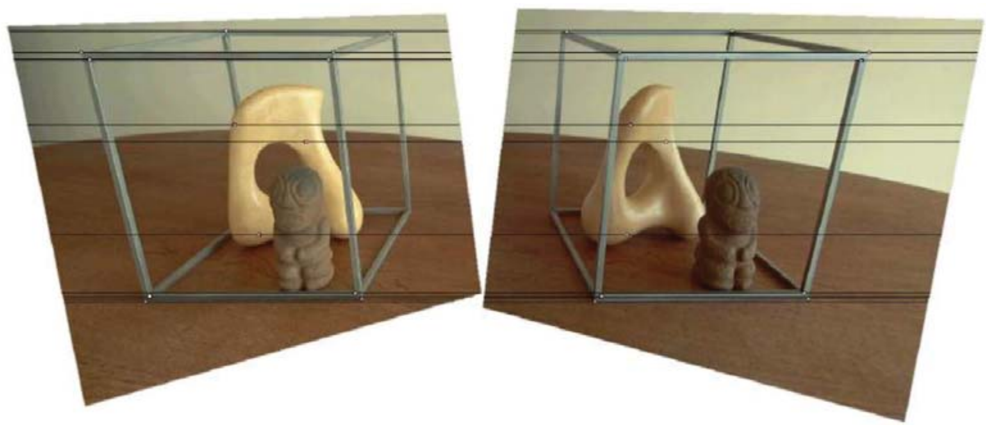
At x-infinity

Stereo Rectification Algorithm

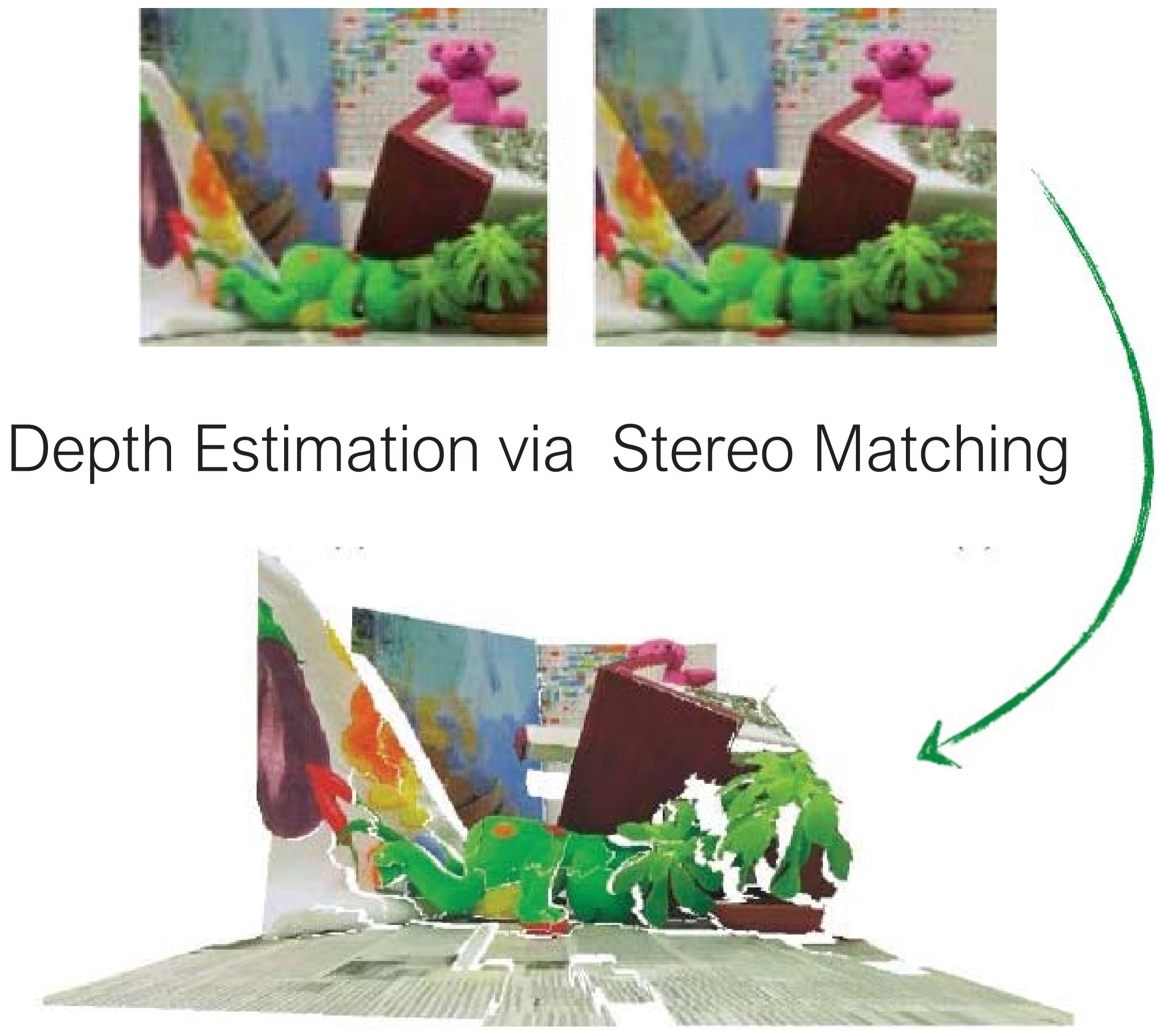
1. Estimate $\mathbf{E}$ using the 8 point algorithm (SVD)
2. Estimate the epipole $\mathbf{e}$ (SVD of $\mathbf{E}$)
3. Build $\mathbf{R}_{rect}$ from $\mathbf{e}$
4. Decompose $\mathbf{E}$ into $\mathbf{R}$ and $\mathbf{T}$
5. Set $\mathbf{R}_1=\mathbf{R}_{rect}$ and $\mathbf{R}_2 = \mathbf{R}_{rect}\mathbf{R}^\mathbf{T}$
6. Rotate each left camera point (warp image)
   $[x' \; y' \; z'] = \mathbf{R}_1 [x \; y \; z]$
7. Rectified points as $\mathbf{p} = f/z'[x' \; y' \; z']$
8. Repeat 6 and 7 for right camera points using $\mathbf{R}_2$

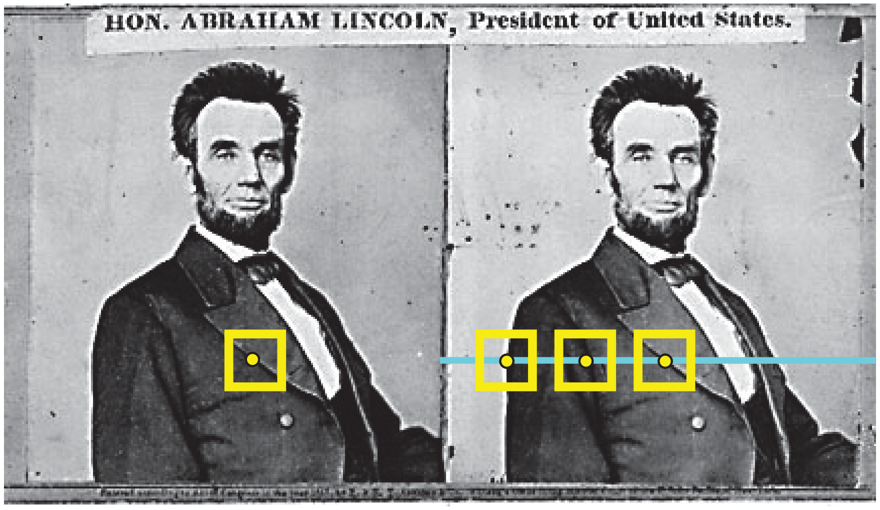


What can we do after rectification?



Stereo matching


Depth Estimation via Stereo Matching

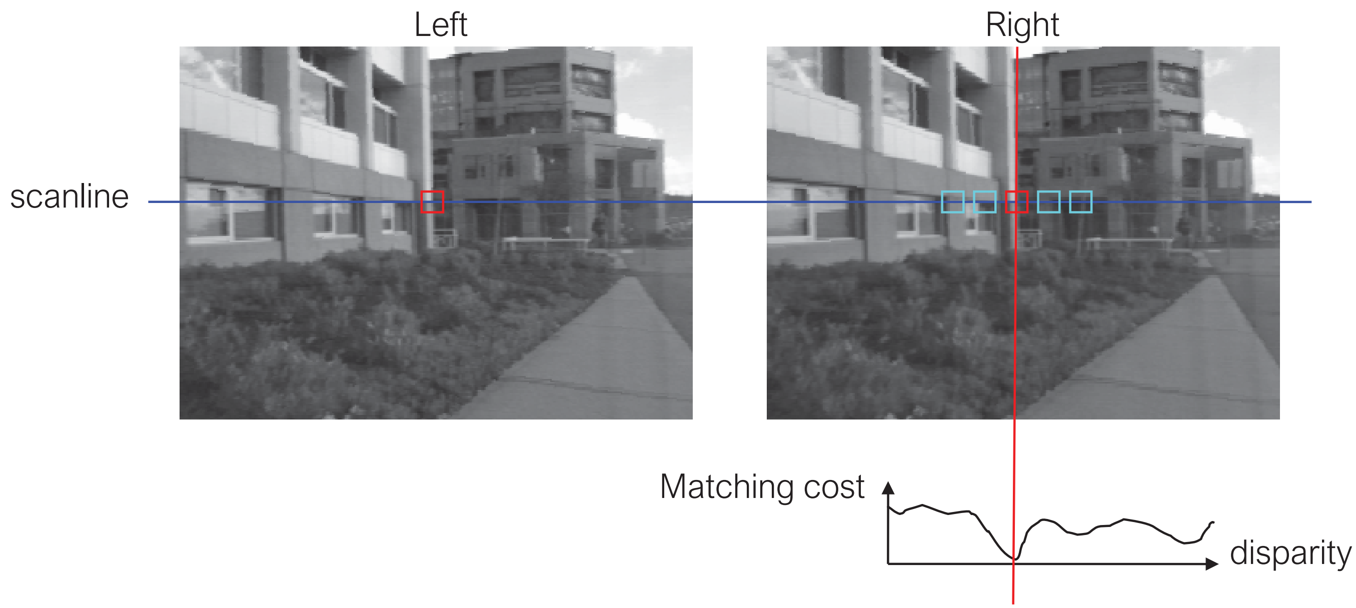


1. Rectify images
   (make epipolar lines horizontal)
2. For each pixel
   a. Find epipolar line
   b. Scan line for best match
   c. Compute depth from disparity

$$Z = \frac{bf}{d}$$

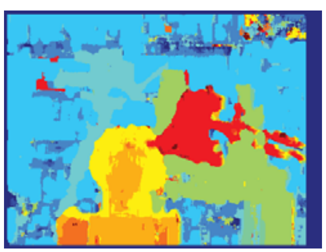
How would you do this?

Stereo Block Matching

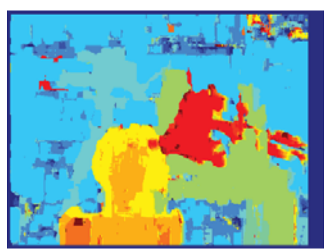


- Slide a window along the epipolar line and compare contents of that window with the reference window in the left image
- Matching cost: SSD or normalized correlation

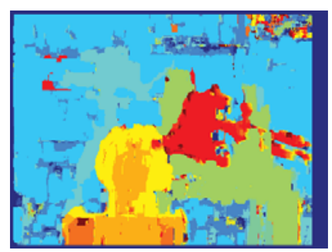
Similarity Measure	Formula
Sum of Absolute Differences (SAD)	$\sum_{(i,j)\in W} \lvert I_1(i,j) - I_2(x+i,y+j)\rvert$
Sum of Squared Differences (SSD)	$\sum_{(i,j)\in W} \left(I_1(i,j) - I_2(x+i,y+j)\right)^2$
Zero-mean SAD	$\sum_{(i,j)\in W} \lvert I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i,y+j) + \bar{I}_2(x+i,y+j)\rvert$
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

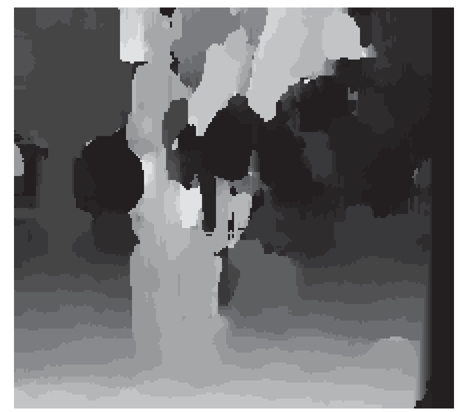


NCC

Ground truth

Effect of window size

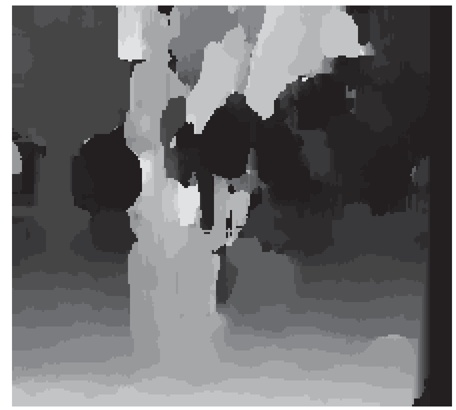
W = 3



W = 20

Effect of window size

W = 3



W = 20

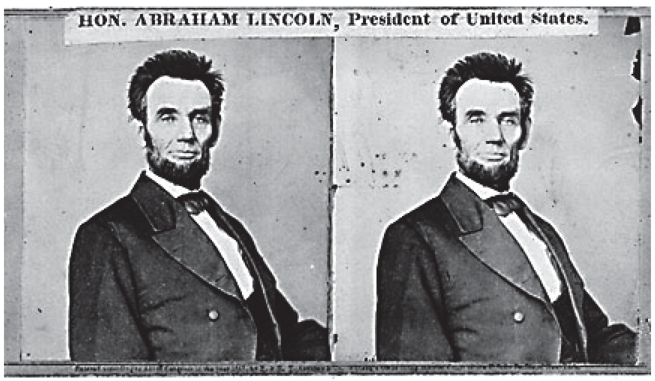
Smaller window

+ More detail
- More noise

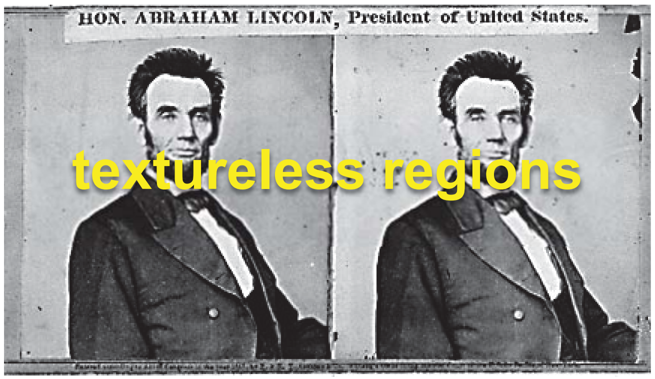
Larger window

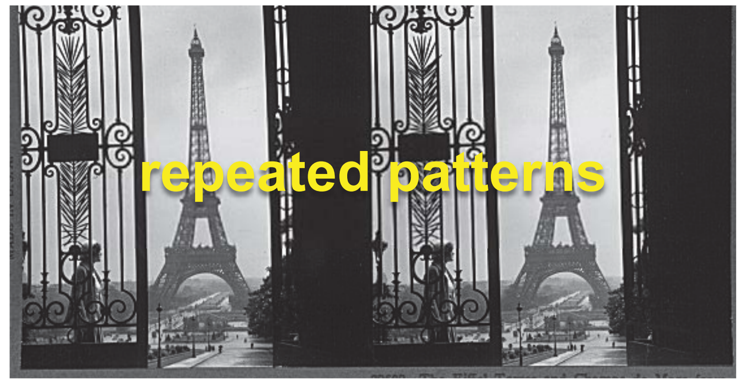
+ Smoother disparity maps
- Less detail
- Fails near boundaries

When will stereo block matching fail?



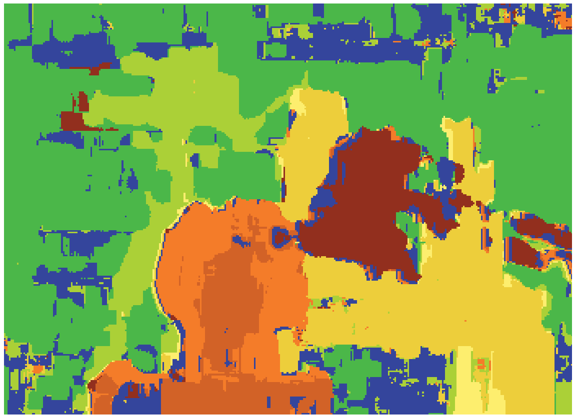
When will stereo block matching fail?





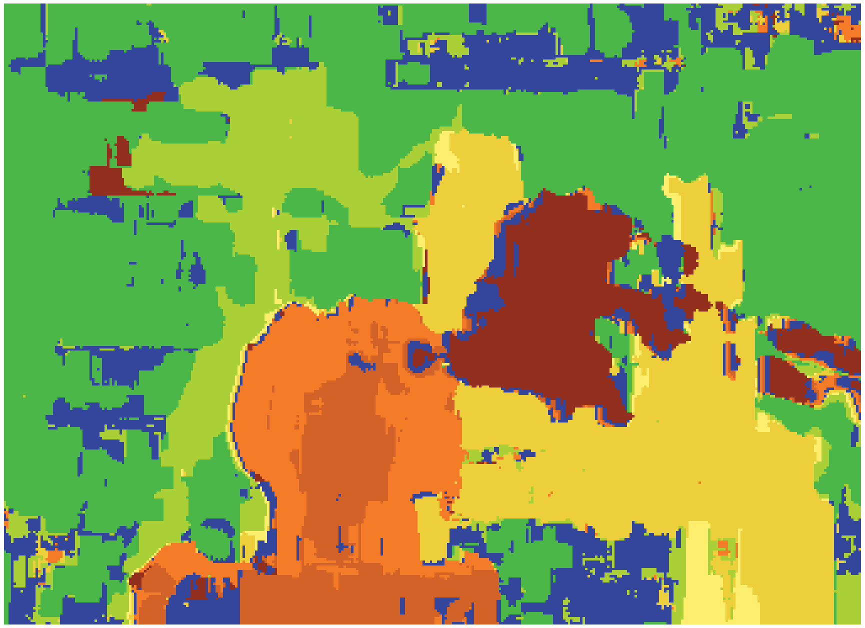
Improving stereo matching

Block matching

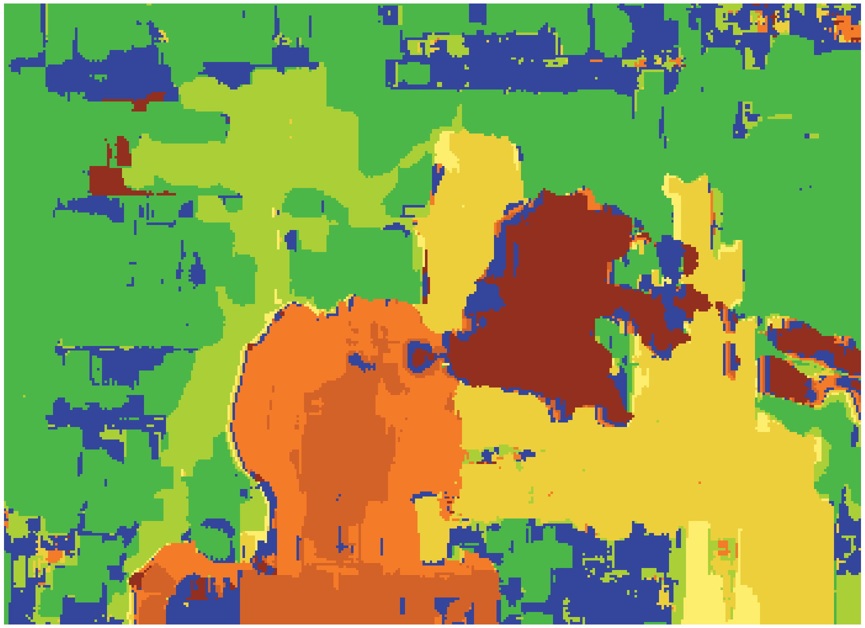


Ground truth

What are some problems with the result?



How can we improve depth estimation?



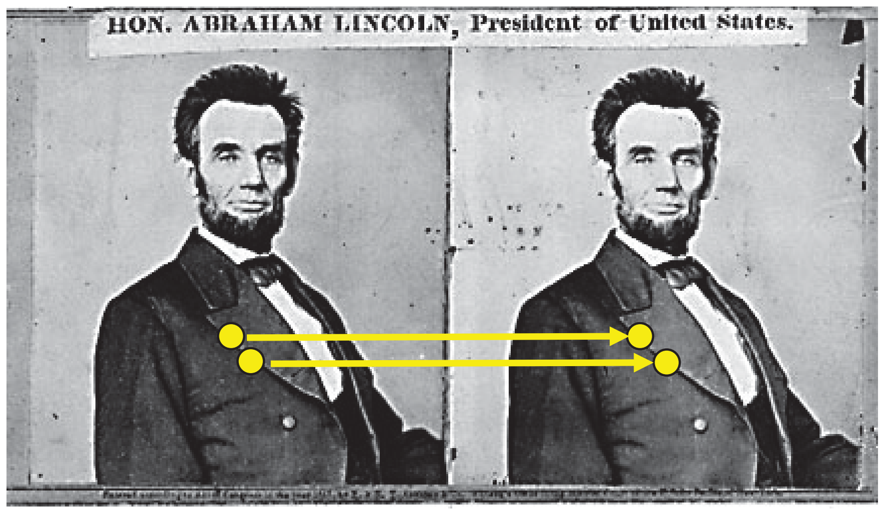
How can we improve depth estimation?

Too many discontinuities.
We expect disparity values to change slowly.

Let's make an assumption:
depth should change smoothly

Stereo matching 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

energy function
(for one pixel)

$$E(d) = \underbrace{E_d(d)}_{\text{data term}} + \lambda \underbrace{E_s(d)}_{\text{smoothness term}}$$

Want each pixel to find a good match in the other image

(block matching result)

Adjacent pixels should (usually) move about the same amount

(smoothness function)

$$E(d) = \boxed{E_d(d)} + \lambda E_s(d)$$

$$E_d(d) = \sum_{(x,y)\in I} C(x, y, d(x, y))$$

data term

SSD distance between windows centered at I(x, y) and J(x+ d(x,y), y)

$$E(d) = E_d(d) + \lambda \boxed{E_s(d)}$$

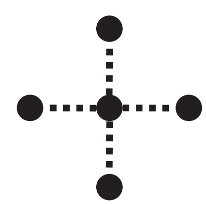
$$E_d(d) = \sum_{(x,y)\in I} C(x, y, d(x, y))$$

SSD distance between windows
centered at I(x, y) and J(x+ d(x,y), y)

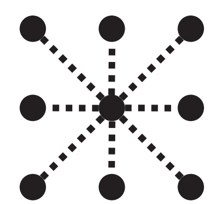
$$E_s(d) = \sum_{(p,q)\in \mathcal{E}} V(d_p, d_q)$$

smoothness term

$\mathcal{E}$: set of neighboring pixels



4-connected
neighborhood

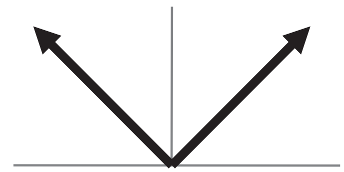


8-connected
neighborhood

smoothness term

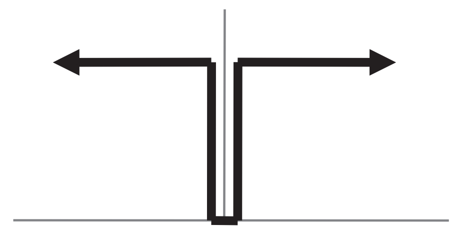
$$\boxed{E_s(d) = \sum_{(p,q)\in\mathcal{E}} V(d_p, d_q)}$$

$$V(d_p, d_q) = aff(I_p, I_q)|d_p - d_q|$$



L$_1$ distance

$$V(d_p, d_q) = aff(I_p, I_q)\begin{cases} 0 \; if \; d_p = d_q \\ 1 \; if \; d_p \neq d_q \end{cases}$$



“Potts model”

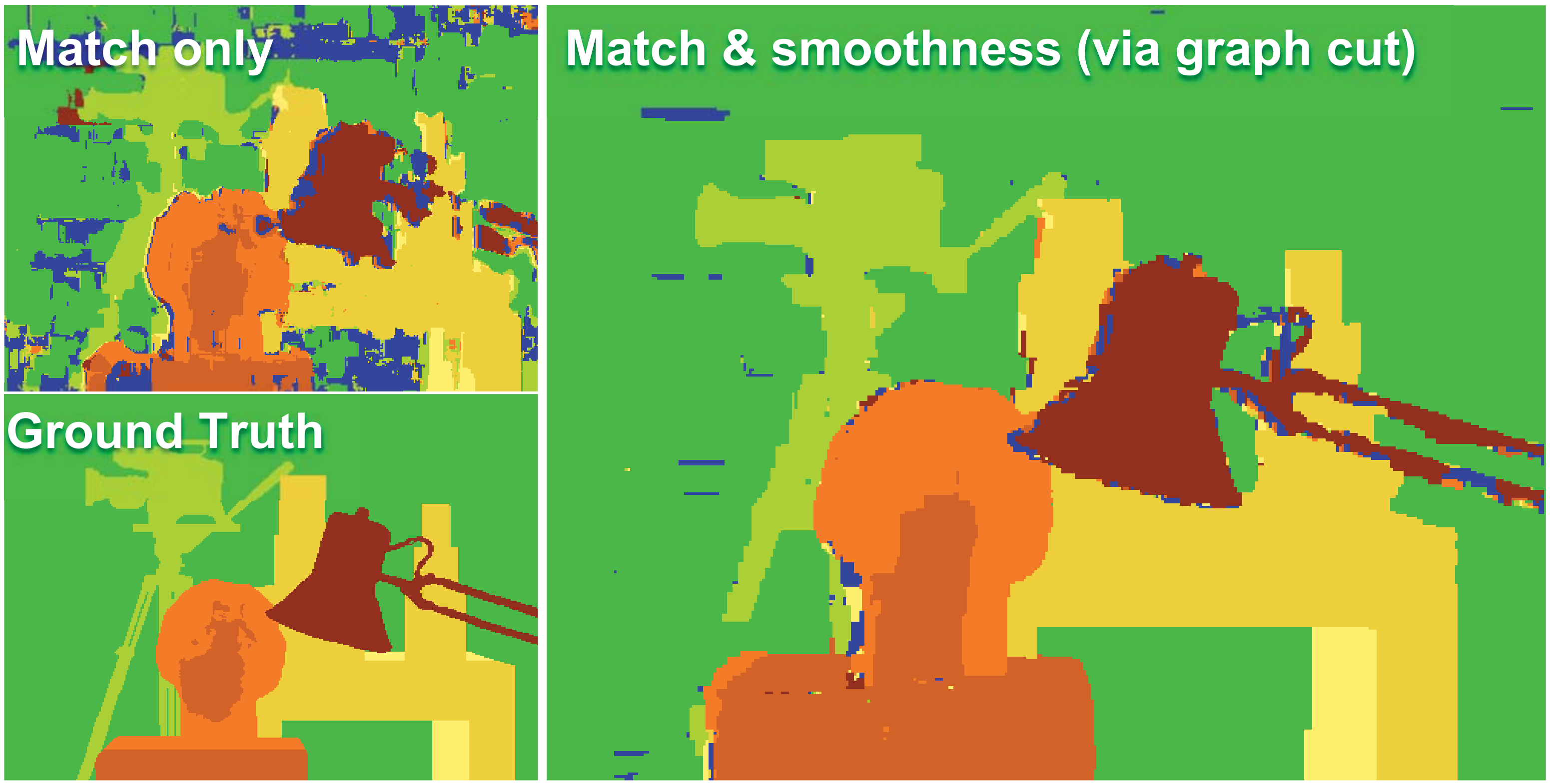
$$aff(I_p, I_q) = e^{\frac{-|I_p - I_q|^2}{\sigma}}$$

Penalty for discontinuities is lower if edge in the image exists

Energy minimization

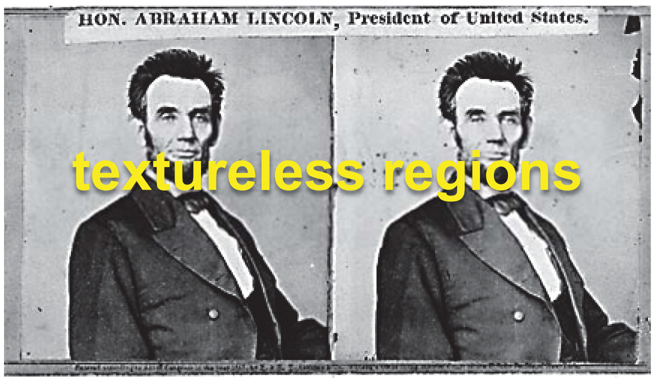
$$E(d) = E_d(d) + \lambda E_s(d)$$

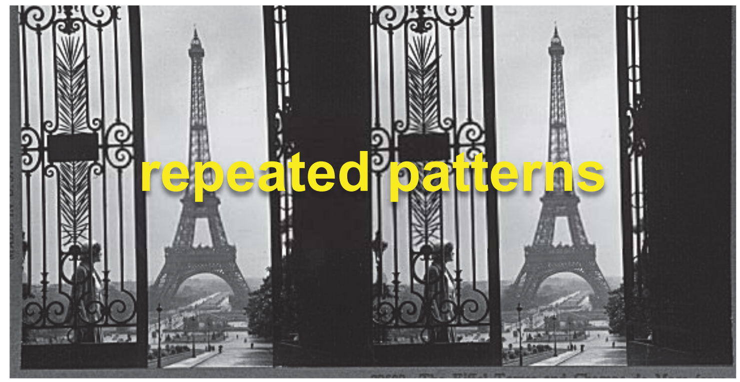
Can minimize this using the graph cut framework presented for segmentation



Y. Boykov, O. Veksler, and R. Zabih, Fast Approximate Energy Minimization via Graph Cuts, PAMI 2001

All of these cases remain difficult, what can we do?

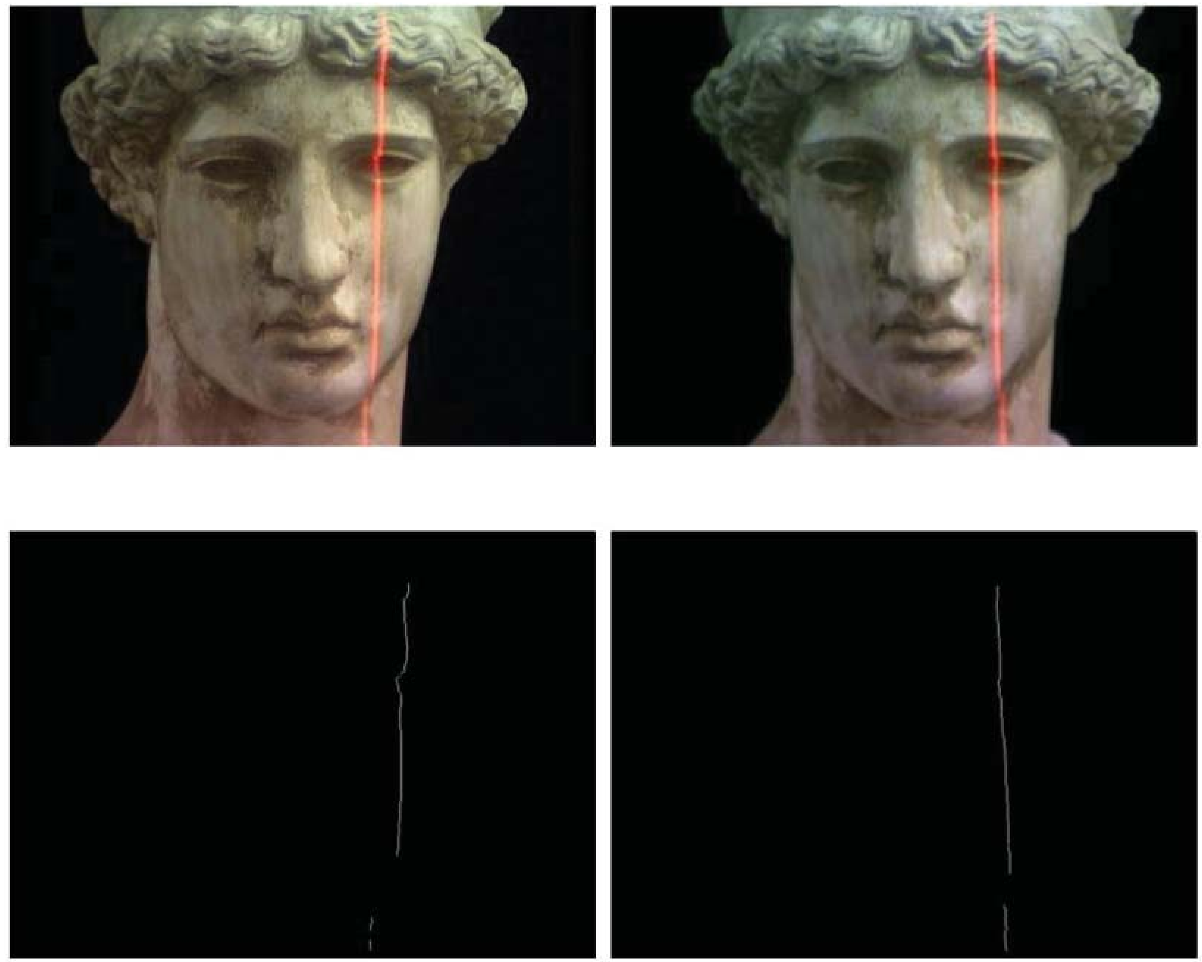




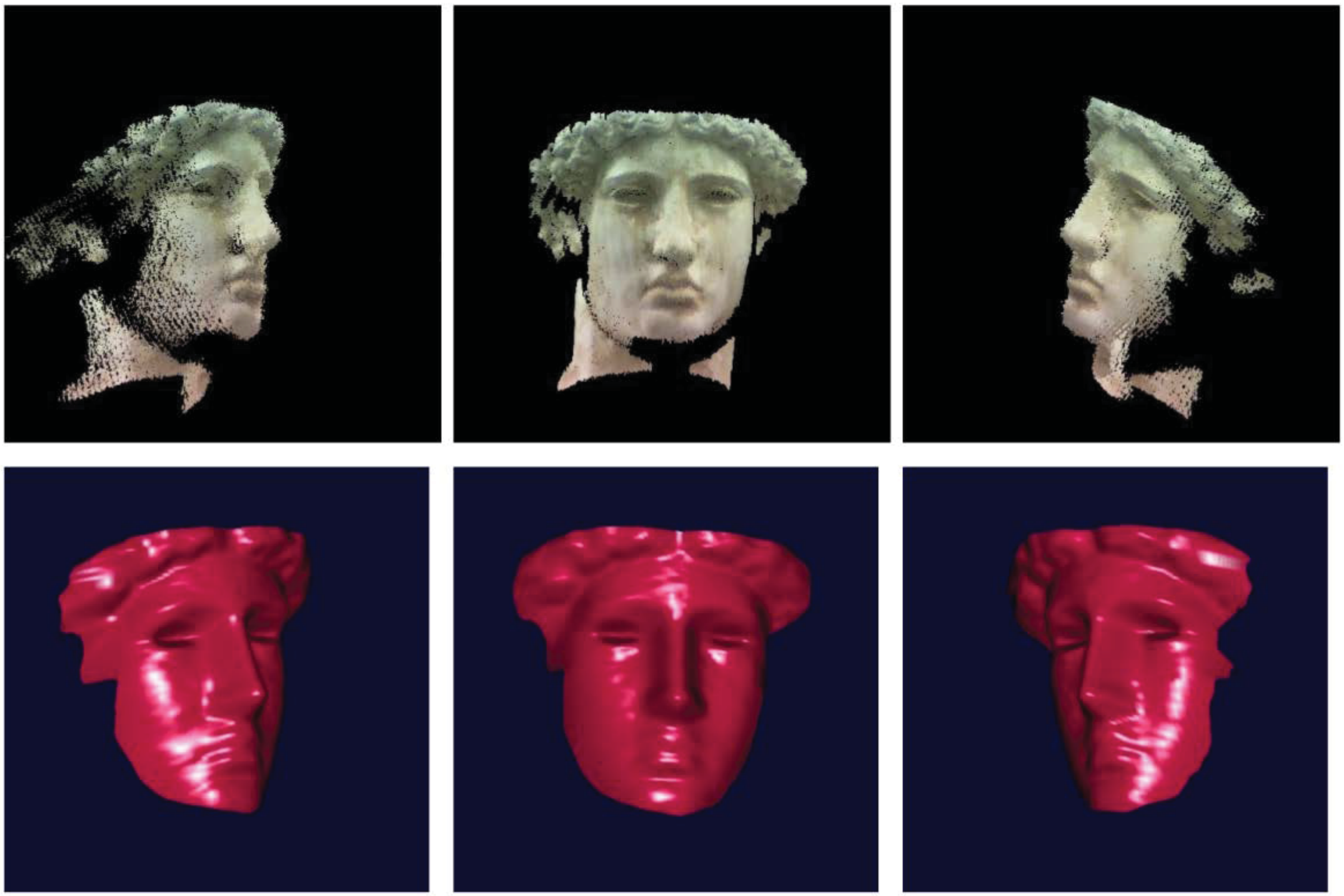
Structured light

Use controlled (“structured”) light to make correspondences easier

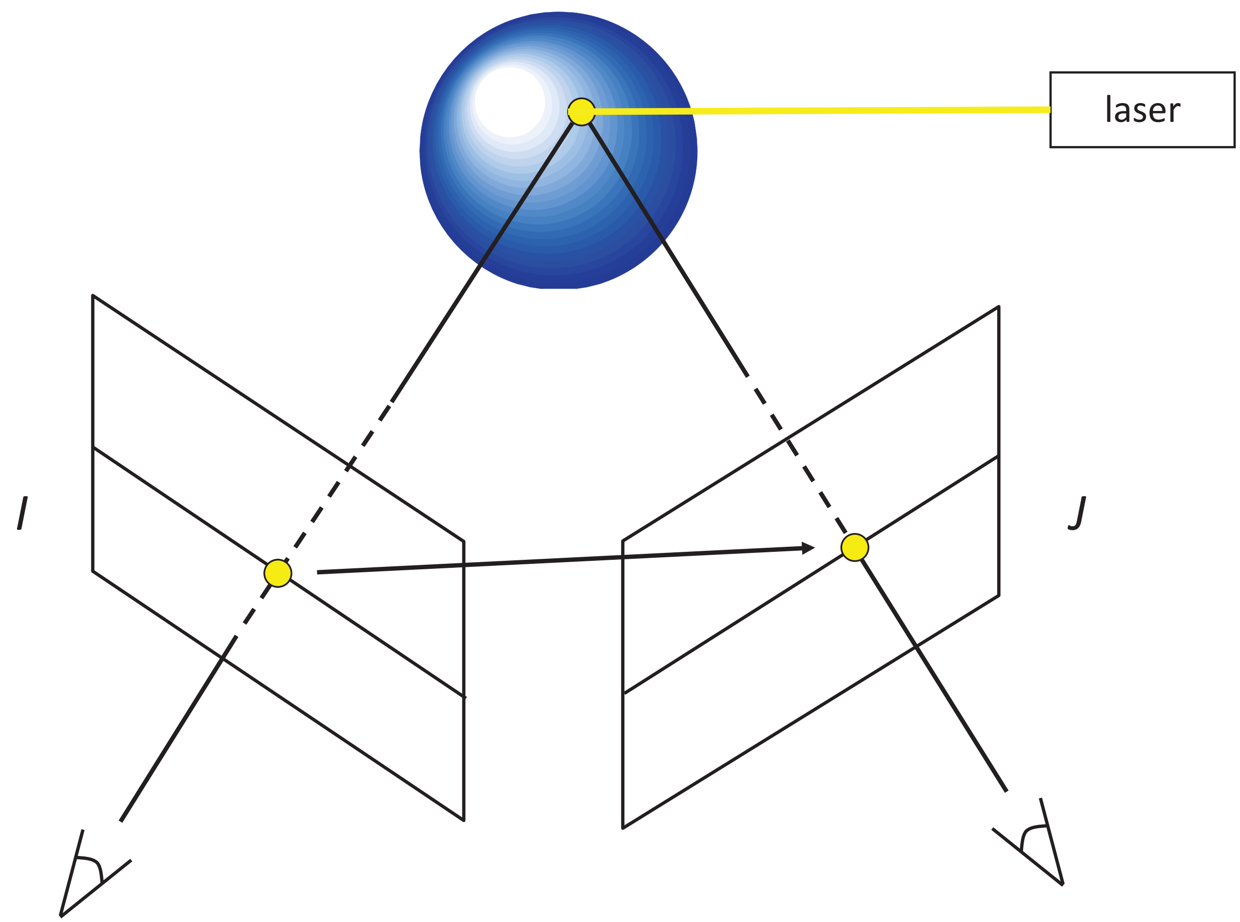
Disparity between laser points on the same scanline in the images determines the 3-D coordinates of the laser point on object



Use controlled ("structured") light to make correspondences easier

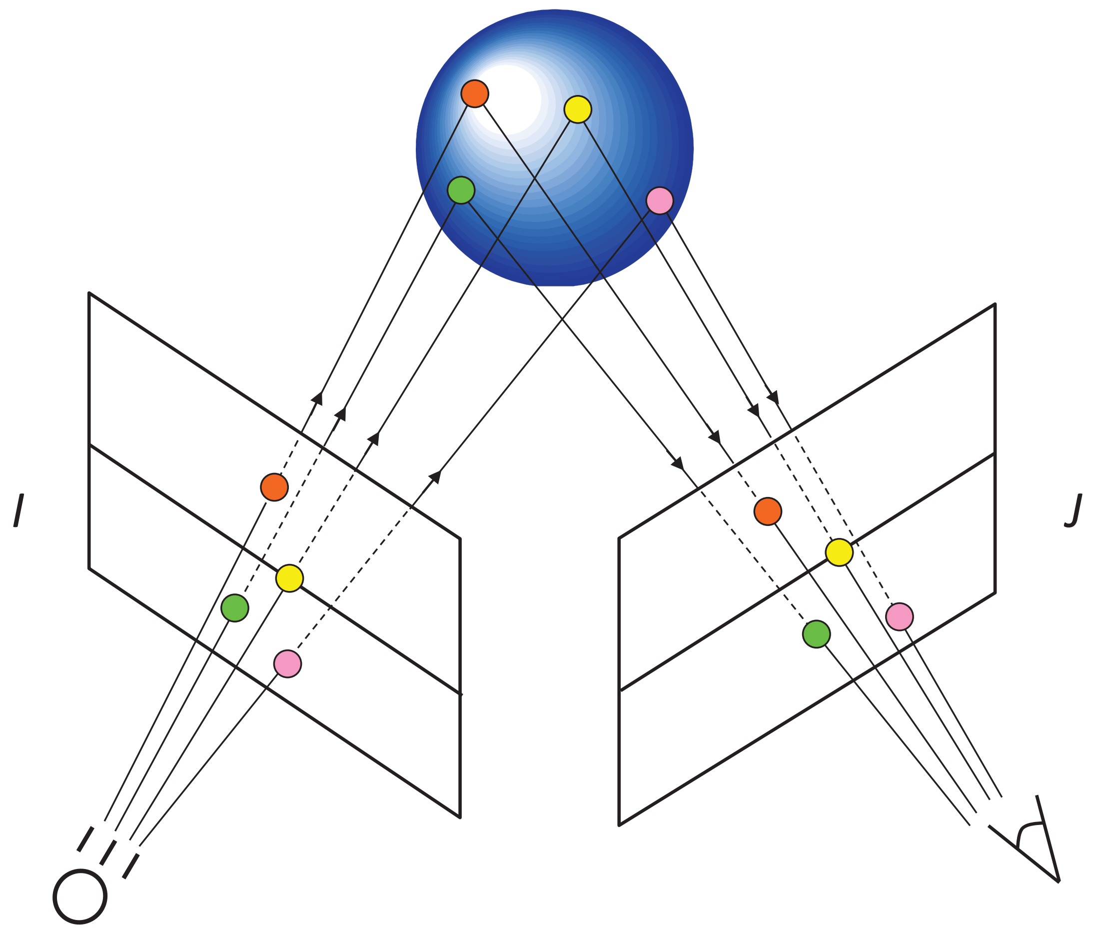


Structured light and two cameras

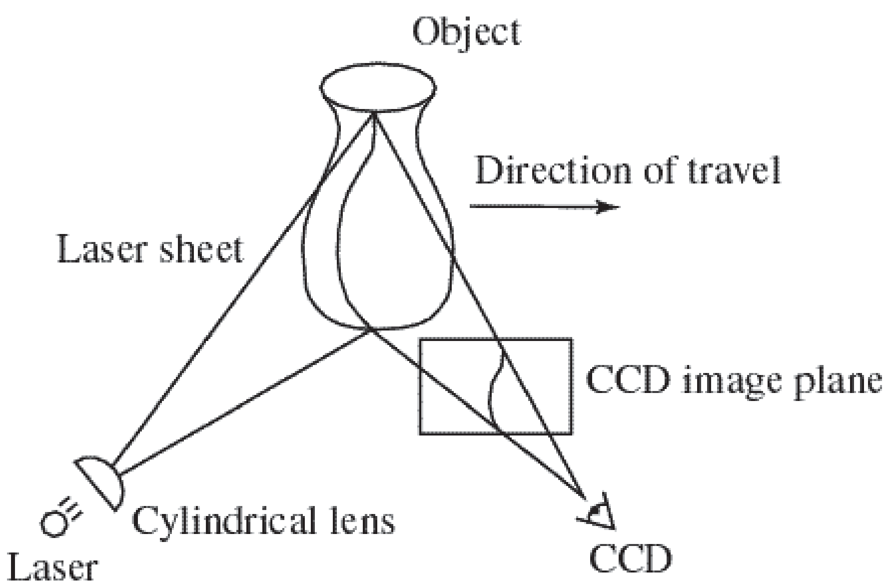


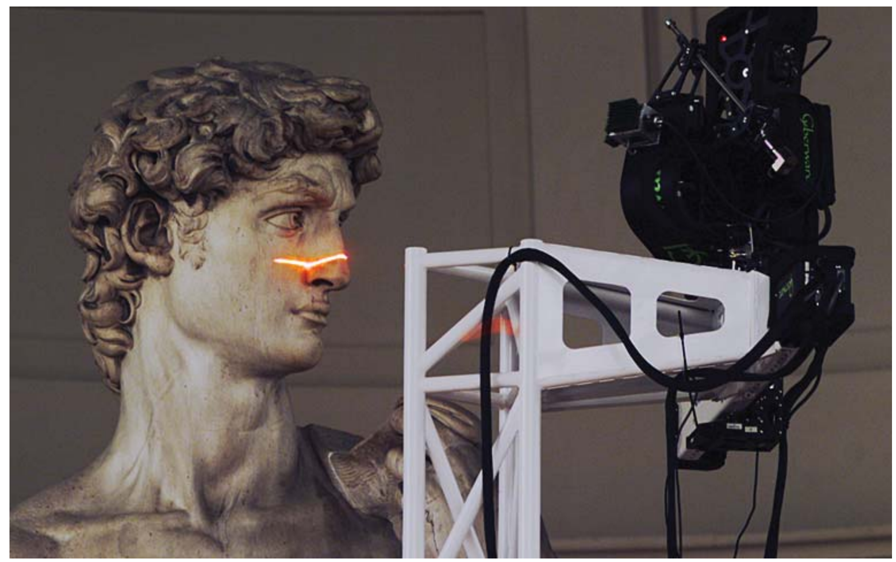
Structured light and one camera

Projector acts like
“reverse” camera



Example: Laser scanner

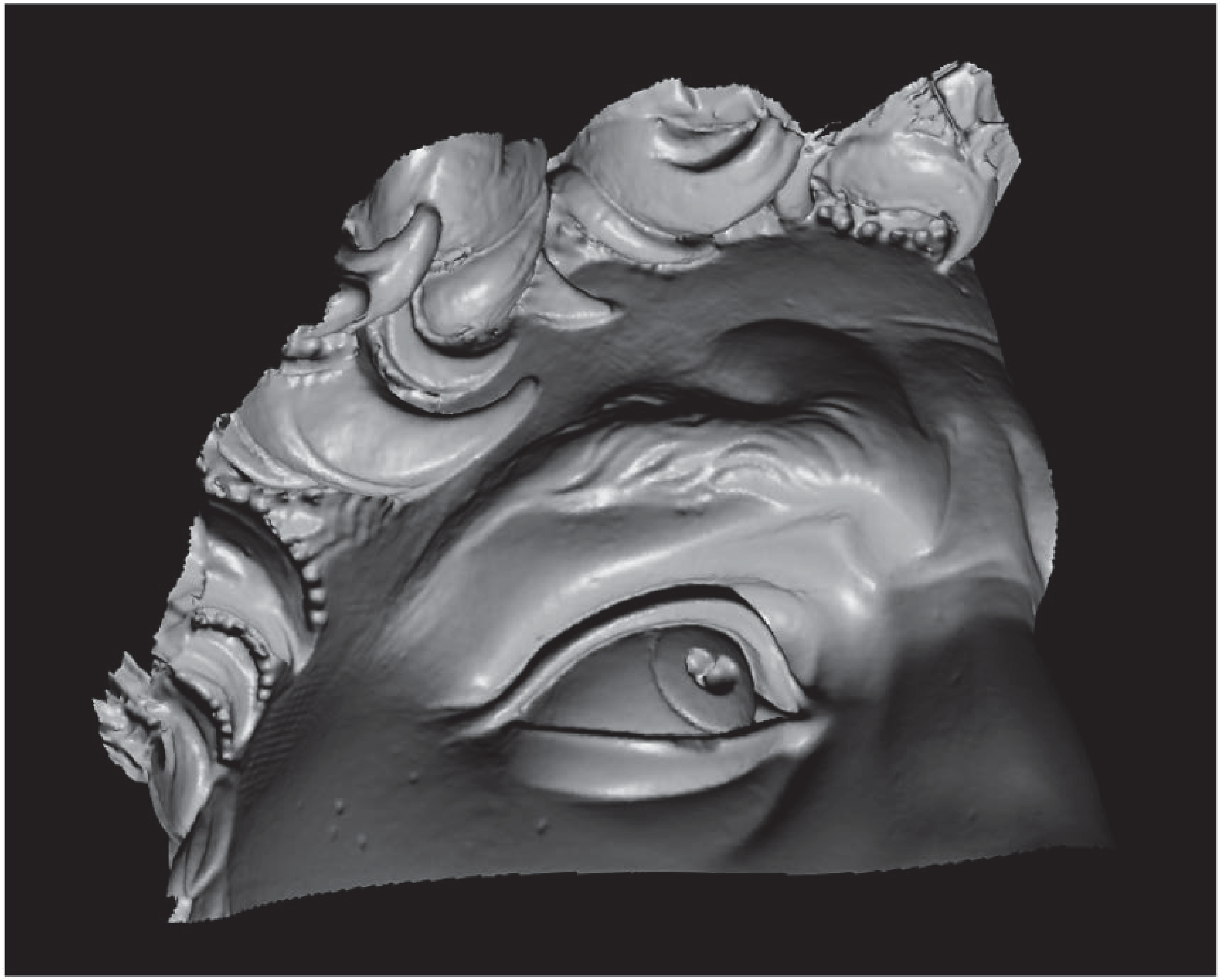




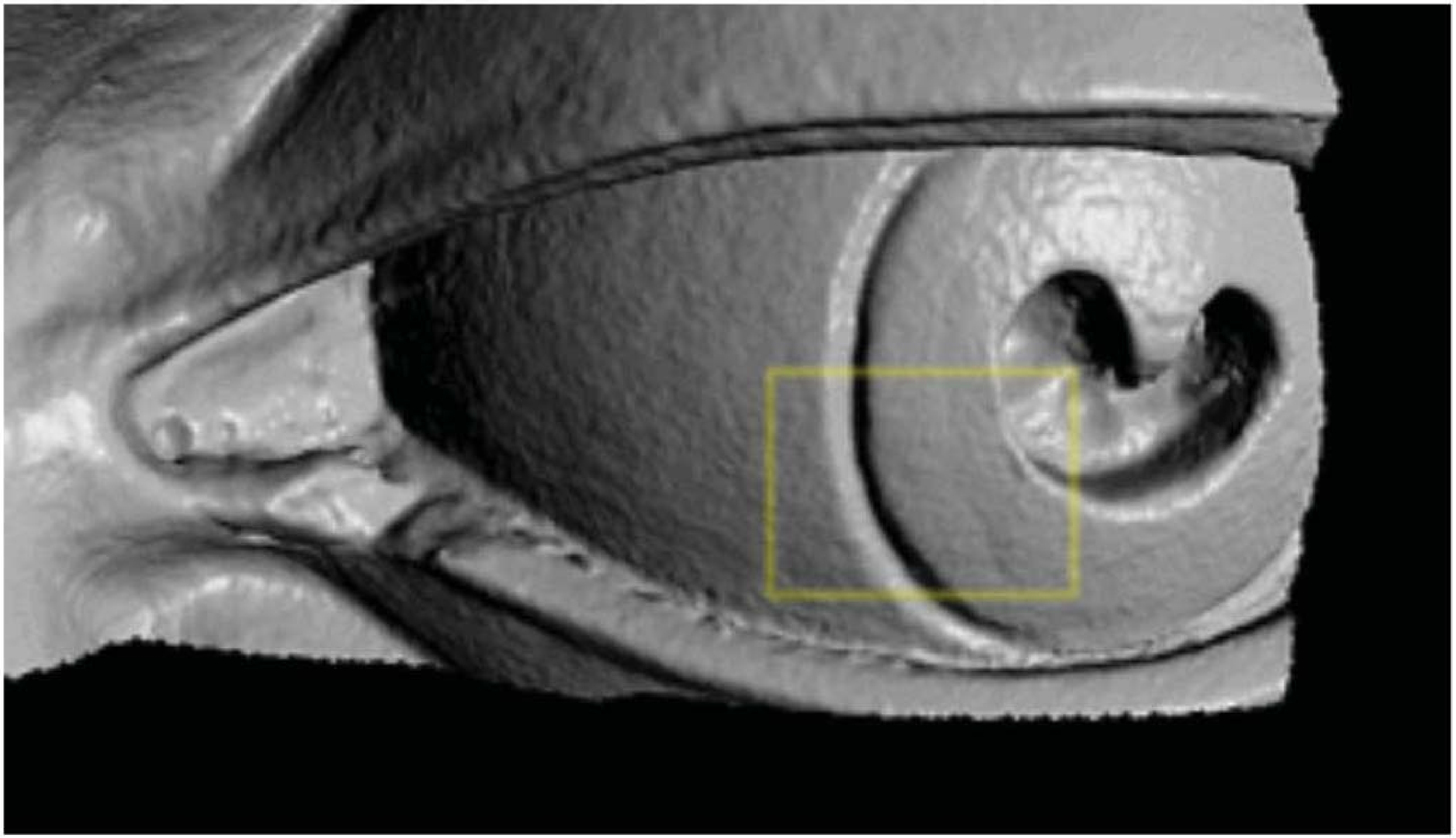
Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/

The Digital Michelangelo Project, Levoy et al.

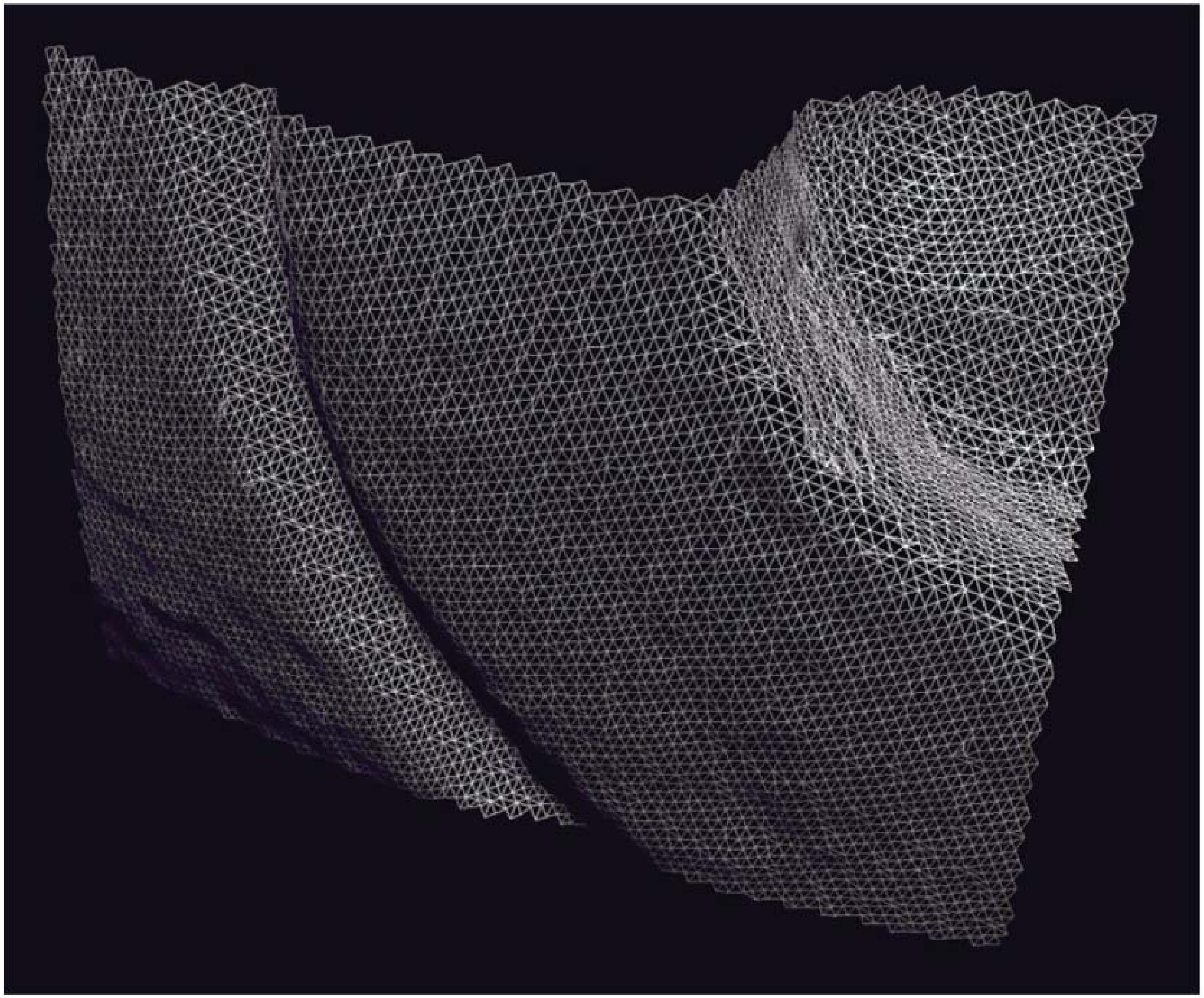
The Digital Michelangelo Project, Levoy et al.



The Digital Michelangelo Project, Levoy et al.



The Digital Michelangelo Project, Levoy et al.

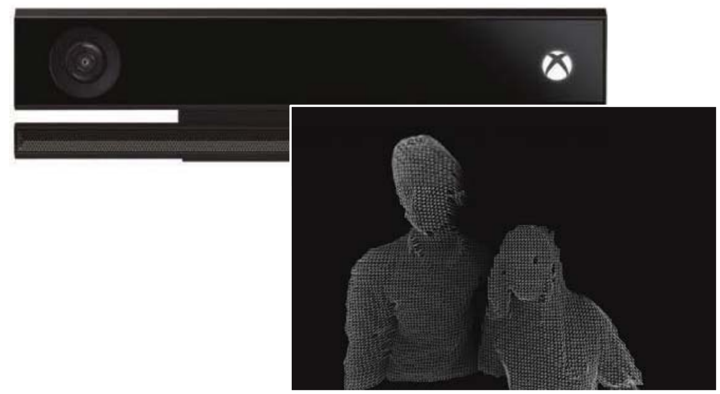


The Digital Michelangelo Project, Levoy et al.

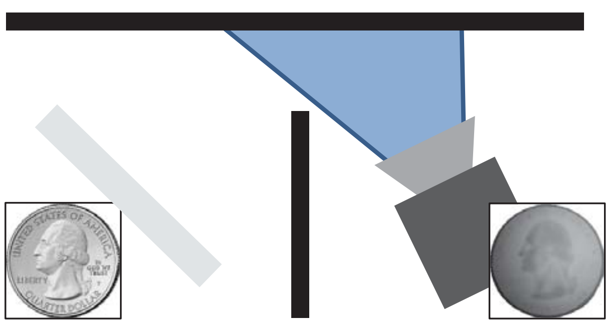
Computational Photography

Learn about structured light and other
(Come to my class: fall 2020)

cameras that take video at the speed of light



cameras that measure depth in real time



cameras that see around corners

cameras that capture entire focal stacks

How do we stitch images from different viewpoints?

Idea 1: Translate one image relative to another.

How do we stitch images from different viewpoints?

Idea 1: Translate one image relative to another.

left on top

right on top

Translation-only stitching is not enough to mosaic these images.

How do we stitch images from different viewpoints?

What else can we try?

How do we stitch images from different viewpoints?

Use image homographies.

We can use homographies when…

1. … the scene is planar; or

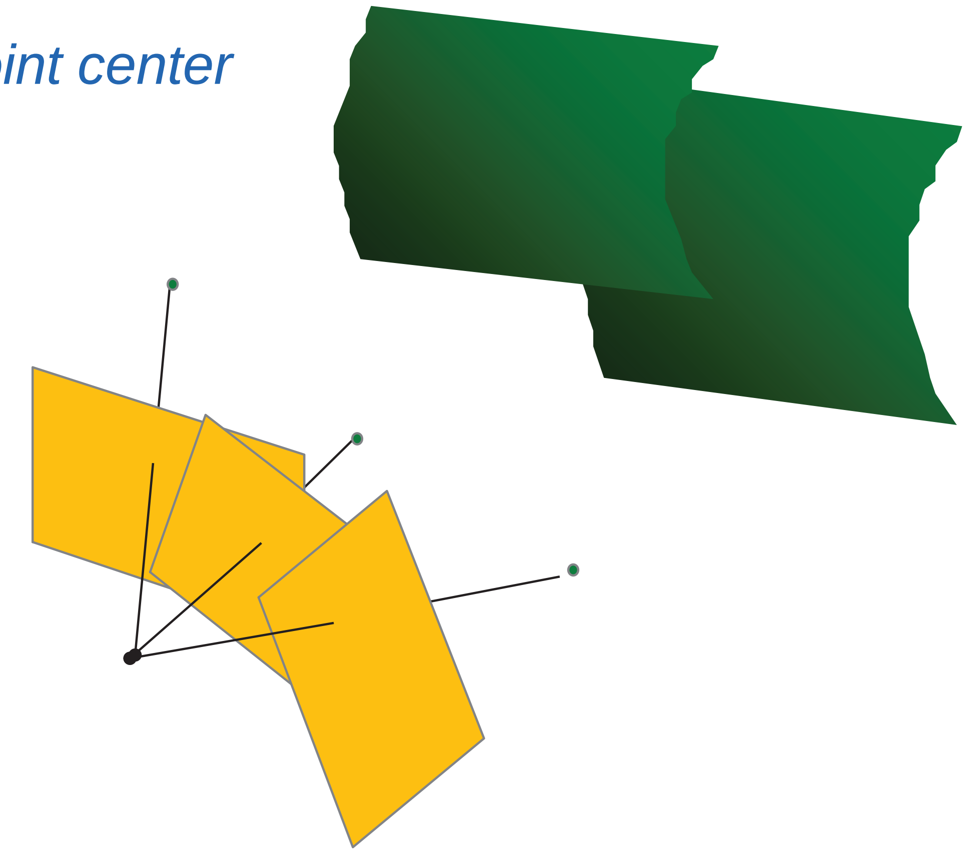
2. … the scene is very far or has small (relative) depth variation
→ scene is approximately planar

We can use homographies when…

3. … the scene is captured under camera rotation only (no translation or pose change)

More on why this is the case in a later lecture.

Homography can map images with a joint center



$$P_1 = [\mathrm{I}|0]$$

$$P_2 = [R \quad |0]$$

$$x = P_1 X$$

$$x' = P_2 \ X$$

Homography mapping $x' = R \quad x$

On the other hand: joint camera centers implies that 3D reconstruction is not possible!

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

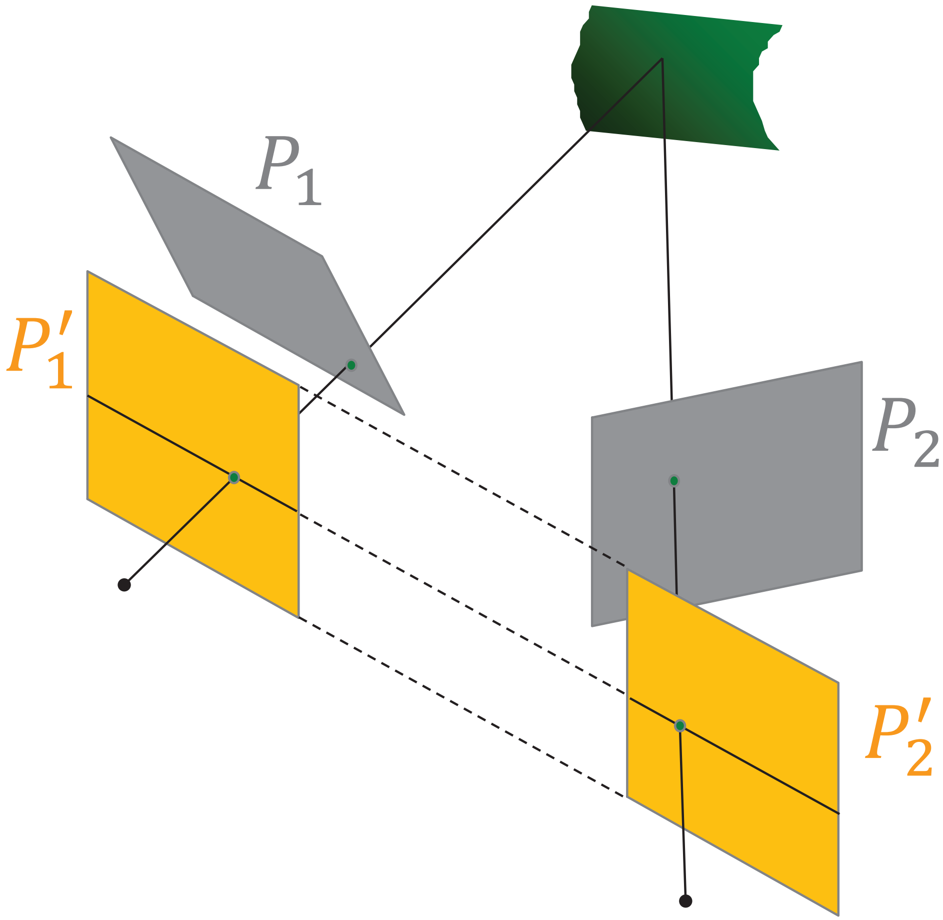
What is stereo rectification?

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

$$P_1 = [\mathrm{I}|0]$$
$$P_1' = [R'|0]$$

$$x = P_1 X$$
$$x' = P_1' X$$

Homography mapping $x' = R'x$



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

References

Basic reading:

- Szeliski textbook, Section 8.1 (not 8.1.1-8.1.3), Chapter 11, Section 12.2.
- Hartley and Zisserman, Section 11.12.