AI70601/CSE61101

3D Vision and Machine Perception

Prof. Kyungdon Joo

3D Vision & Robotics Lab.

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

Some materials, figures, and slides (used for this course) are from textbooks, published papers, and other open lectures

Epipolar constraints

- These constraints can be used in RANSAC like homography

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

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

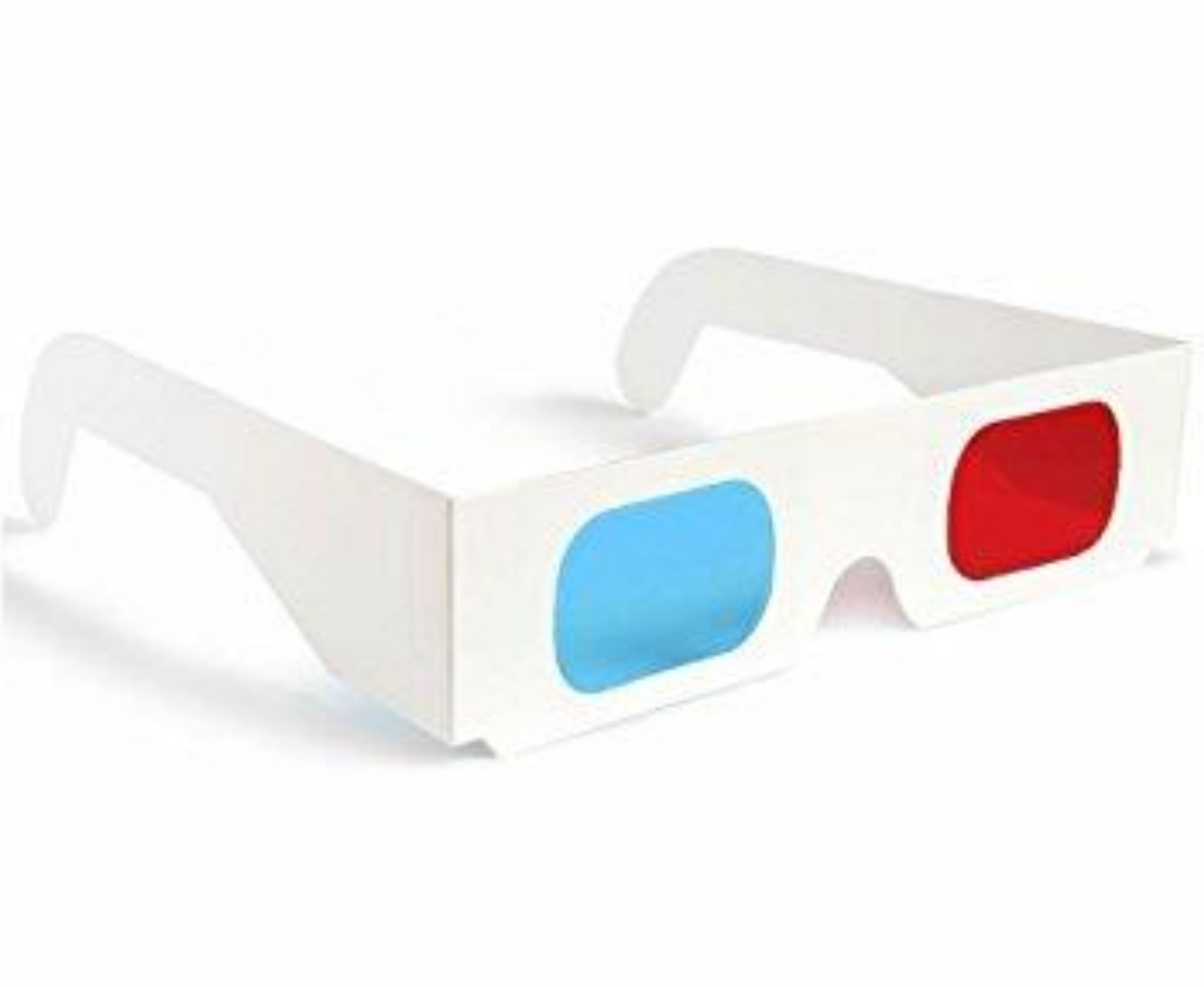
Epipolar constraints

- These constraints can be used in RANSAC like homography
- What is the benefits over using homography?

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

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

Stereo



Revisiting triangulation

How would you reconstruct 3D points?

Left image

Right image

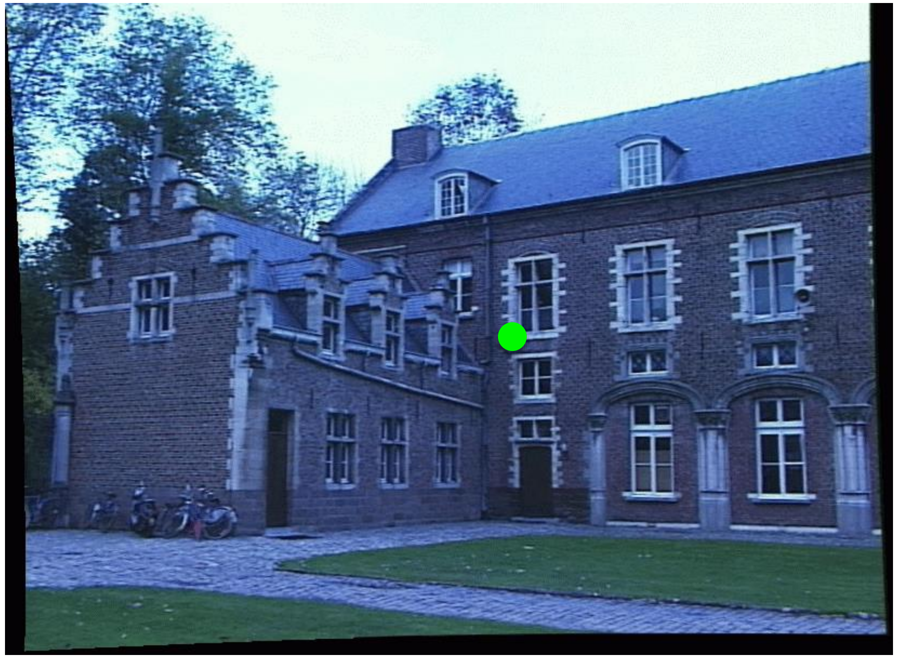
How would you reconstruct 3D points?

Left image

Right image

- Select point in one image (how?)

How would you reconstruct 3D points?

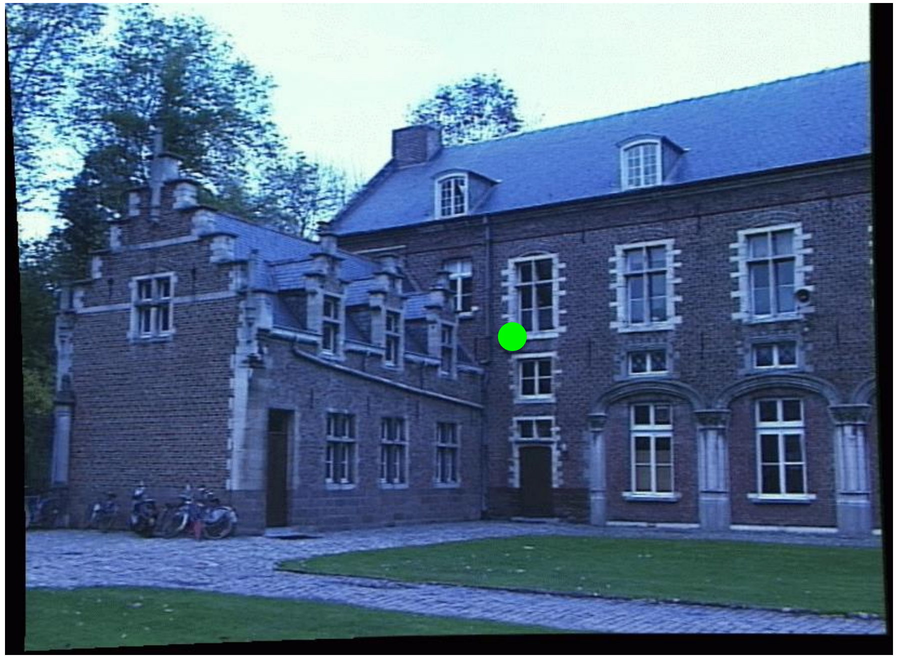


Left image

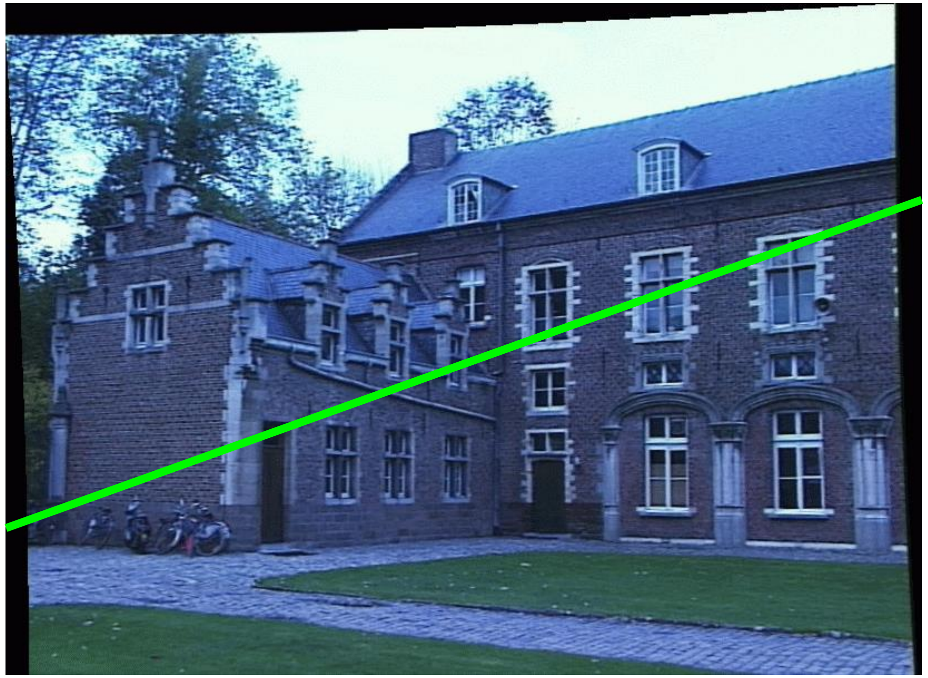
Right image

- Select point in one image (how?)
- Form epipolar line for that point in second image (how?)

How would you reconstruct 3D points?



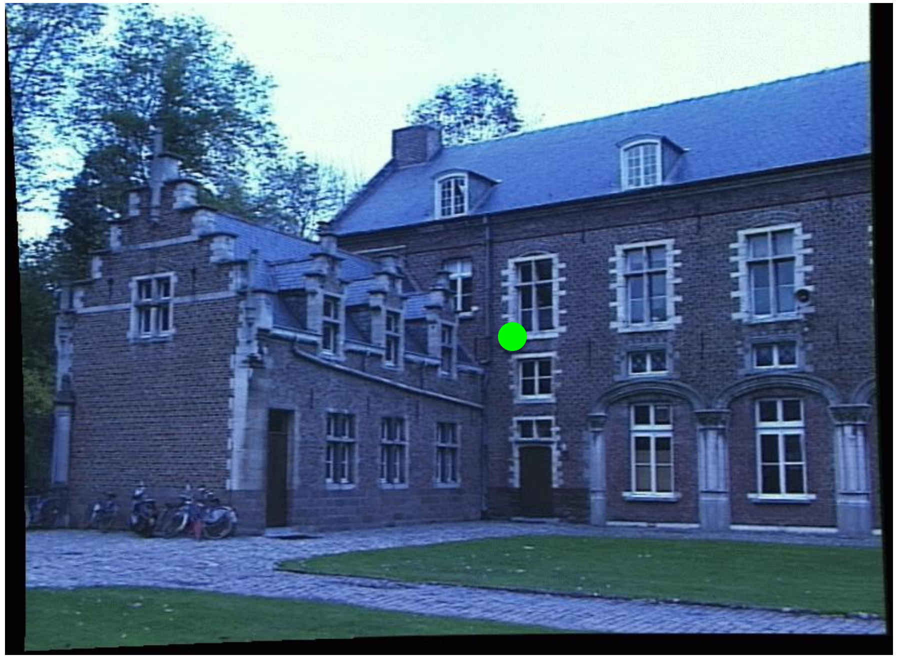
Left image



Right image

- Select point in one image (how?)
- Form epipolar line for that point in second image (how?)
- Find matching point along line (how?)

How would you reconstruct 3D points?

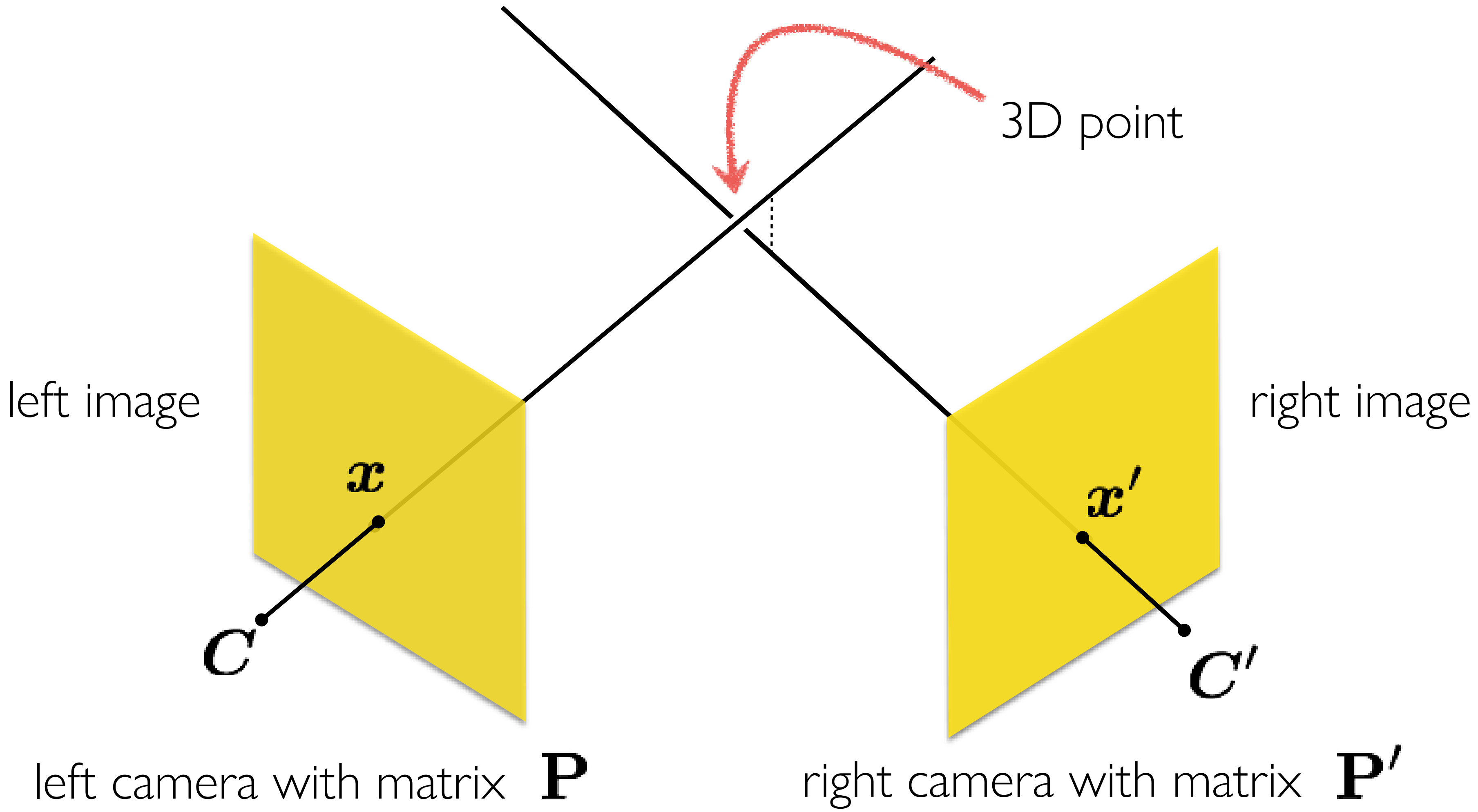


Left image

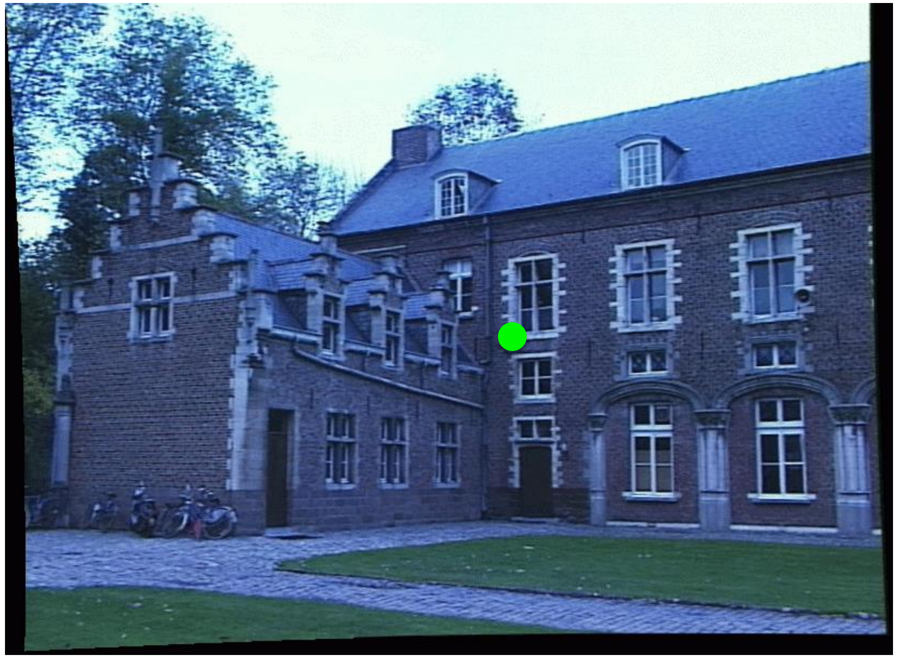
Right image

- Select point in one image (how?)
- Form epipolar line for that point in second image (how?)
- Find matching point along line (how?)
- Perform triangulation (how?)

Triangulation



How would you reconstruct 3D points?



Left image

Right image

- Select point in one image (how?)
- Form epipolar line for that point in second image (how?)
- Find matching point along line (how?)
- Perform triangulation (how?)
- What are the disadvantages of this procedure?

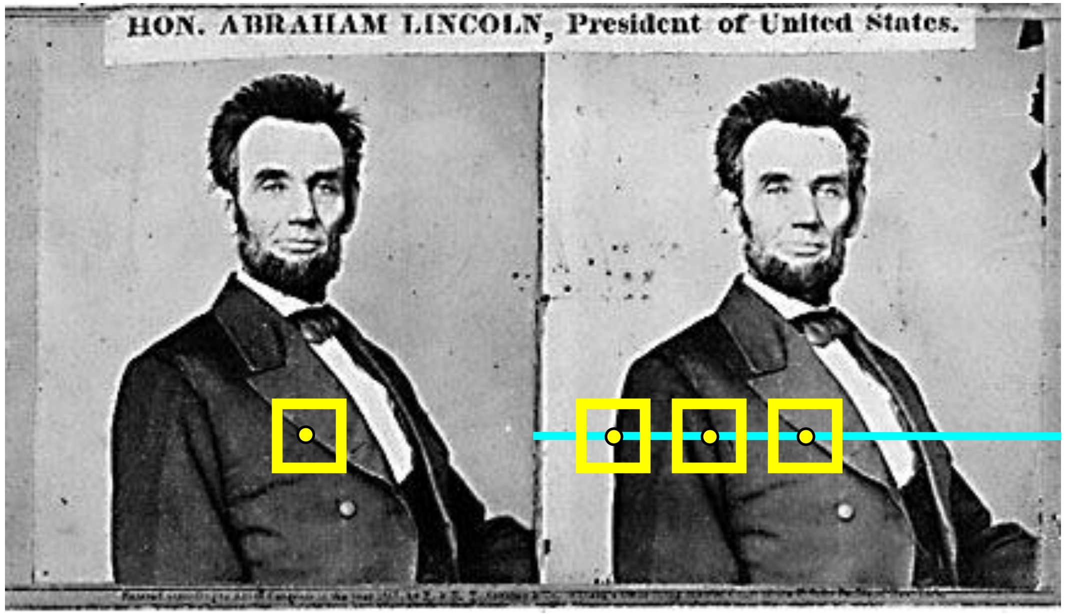
Stereo matching

- What’s different between these two images?

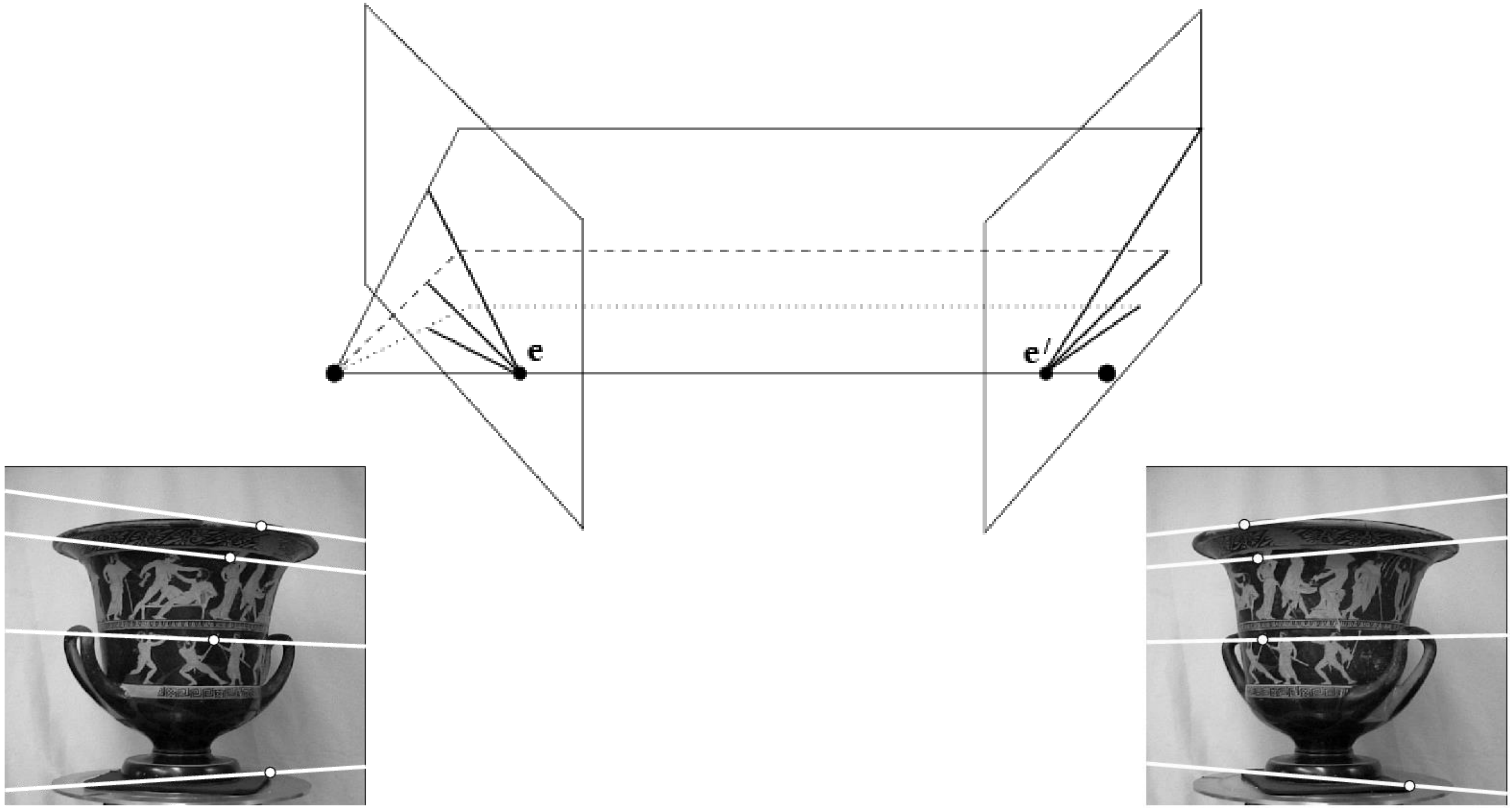
- Objects that are close move more or less?


Depth Estimation via Stereo Matching

Overview of depth estimation in stereo setup

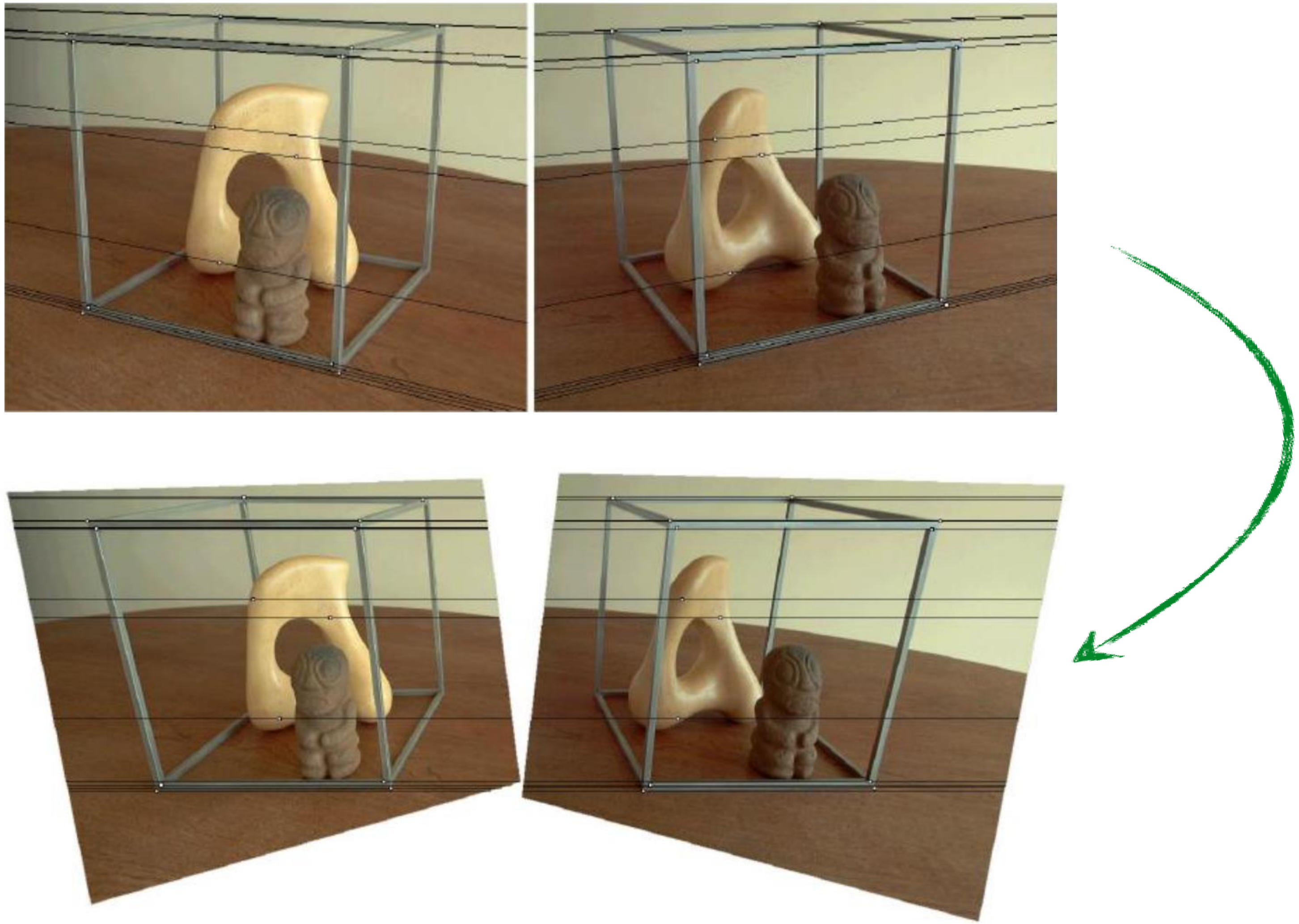


1. Rectify images
   (make epipolar lines horizontal)
2. For each pixel
   - Find epipolar line
   - Scan line for best match
   - Compute depth from disparity ($Z = \frac{bf}{d}$)



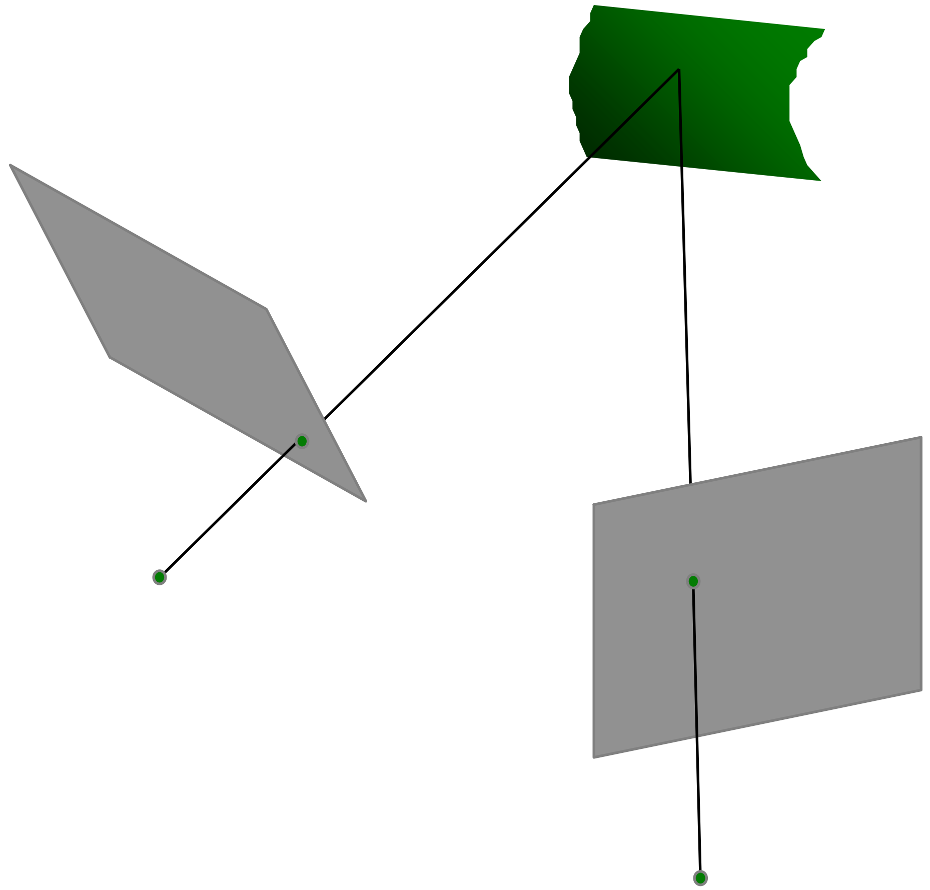
- It’s hard to make the image planes exactly parallel

- How can you make the epipolar lines horizontal?
- Use stereo rectification?



Stereo rectification

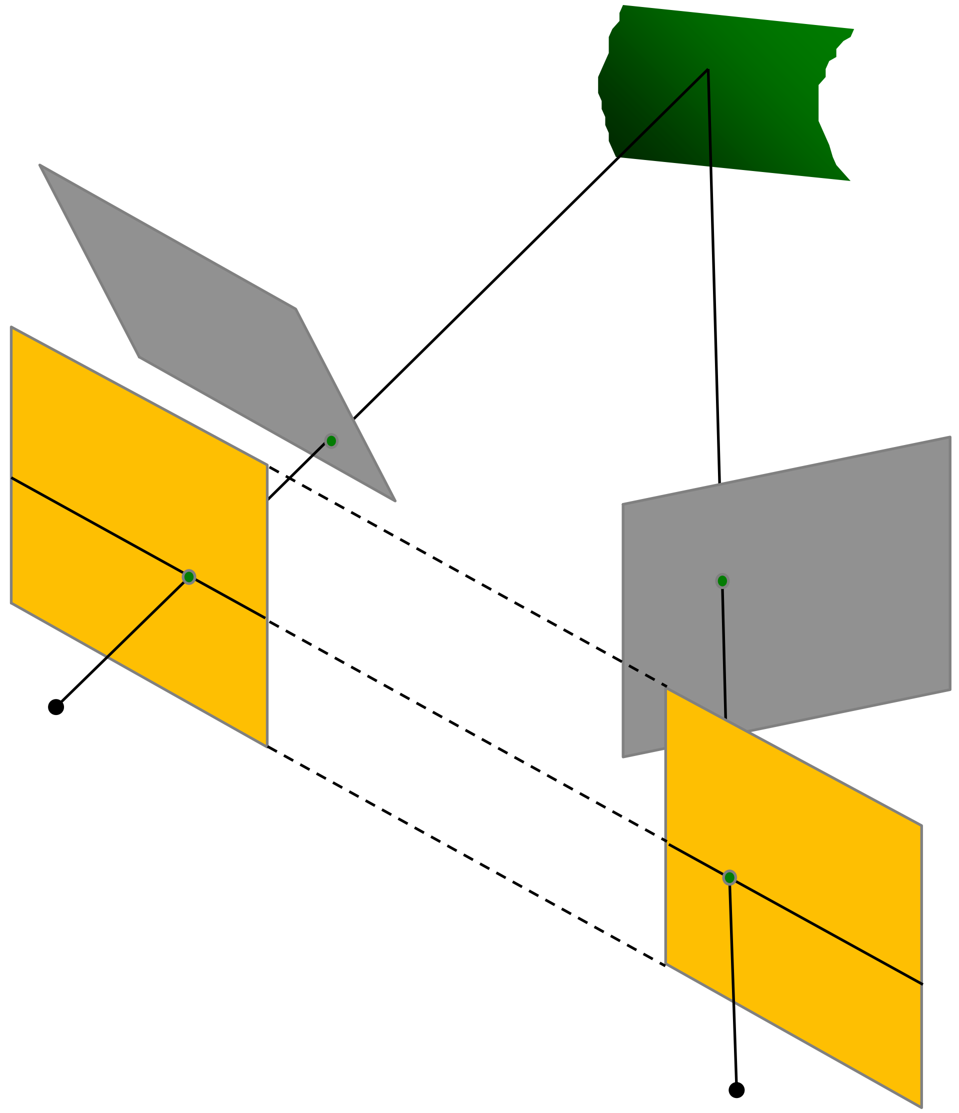
- What is stereo rectification?



- What is stereo rectification?

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

- How can you do this?

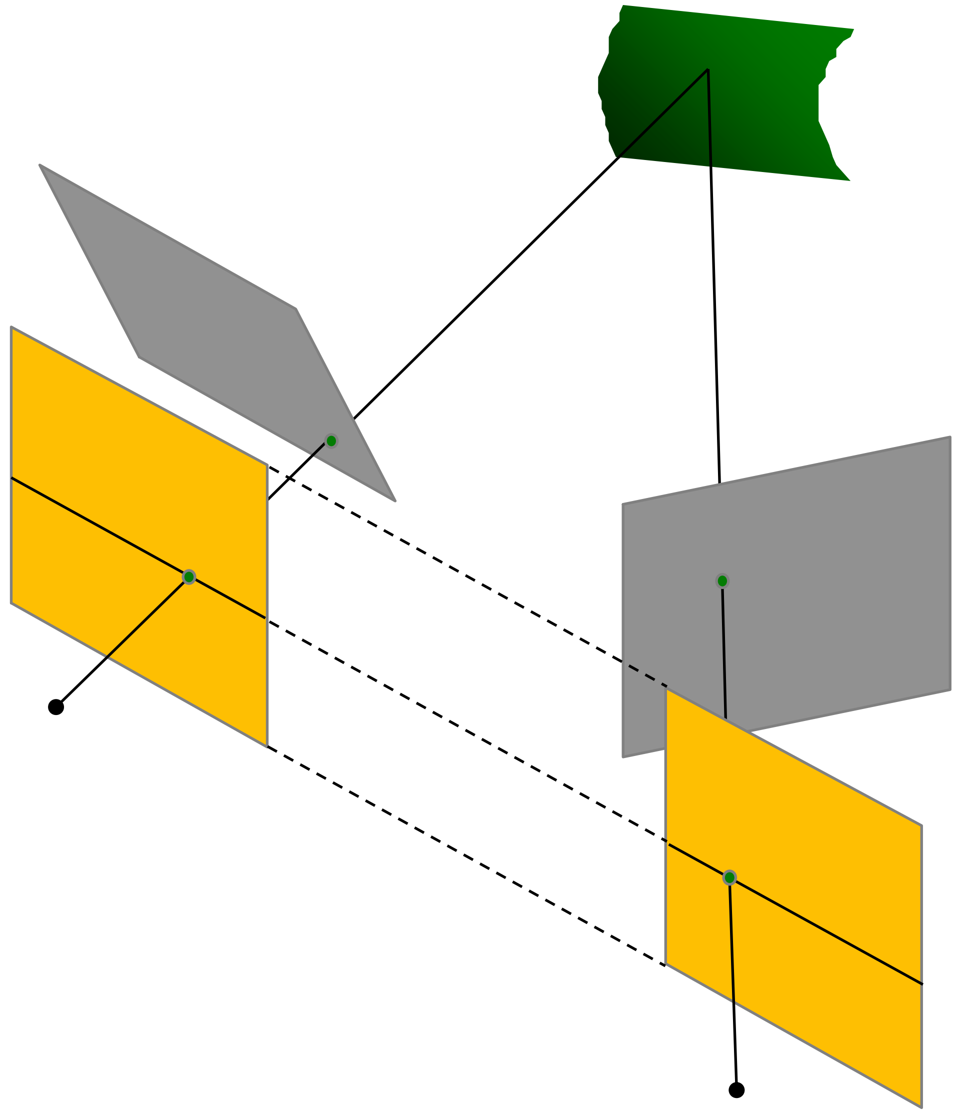


- What is stereo rectification?

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

- How can you do this?

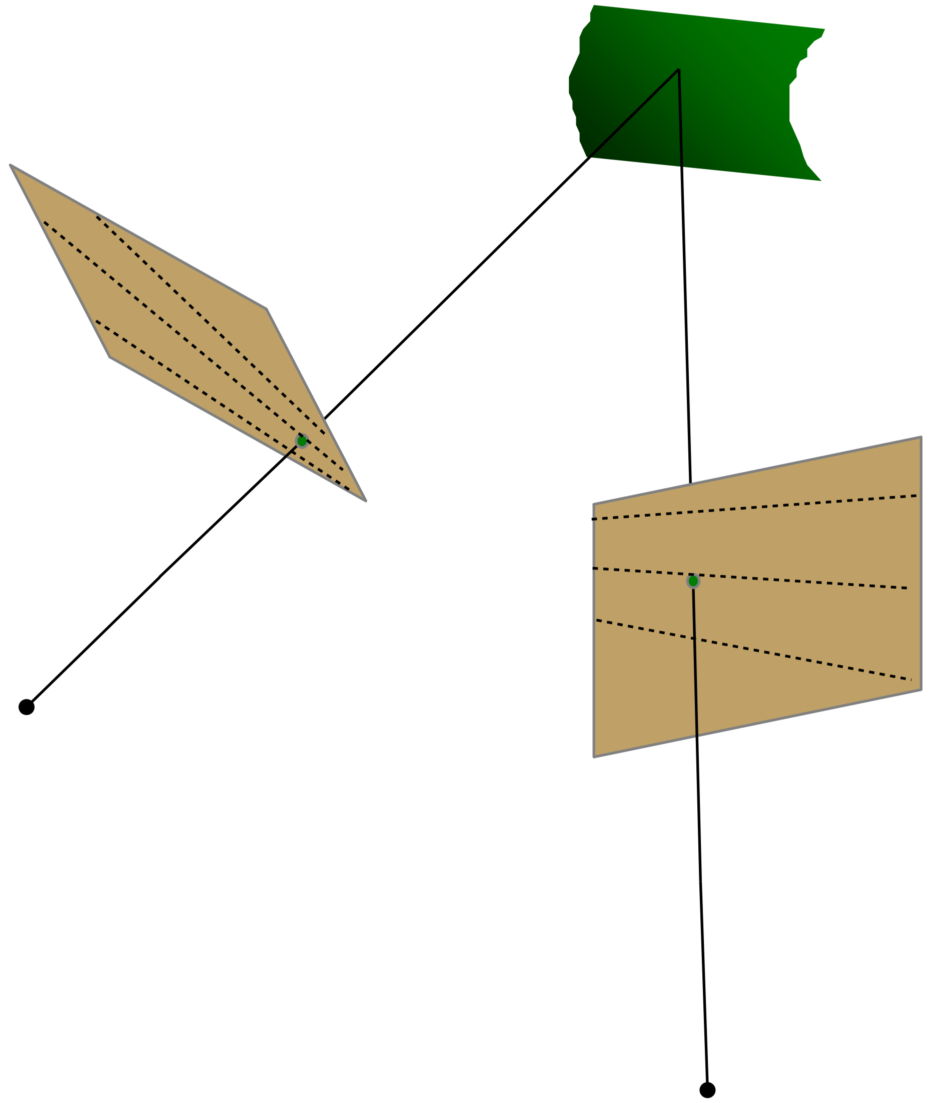
- Need two homographies (3x3 transform), one for each input image reprojection



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

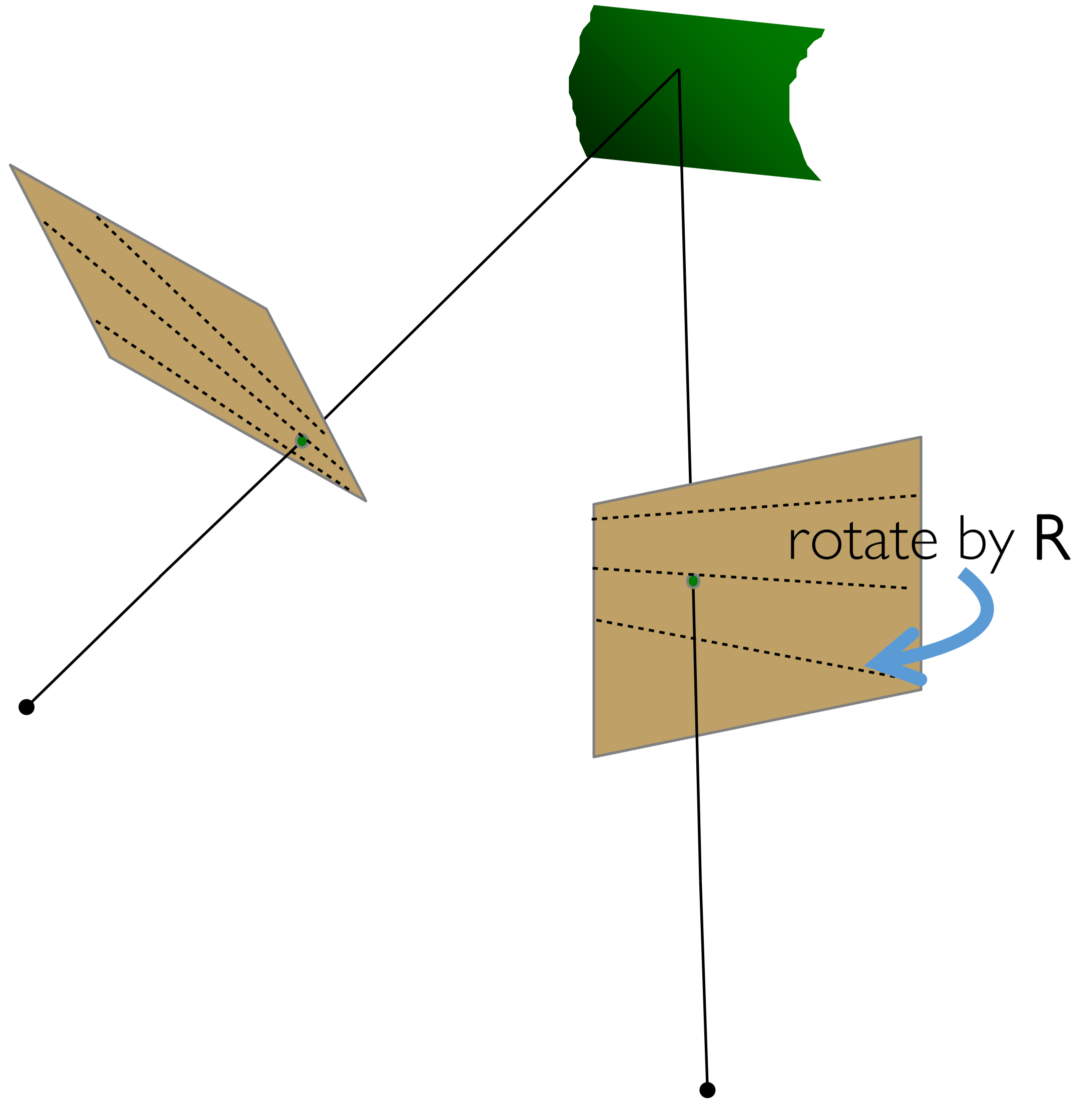
- Stereo Rectification:

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**



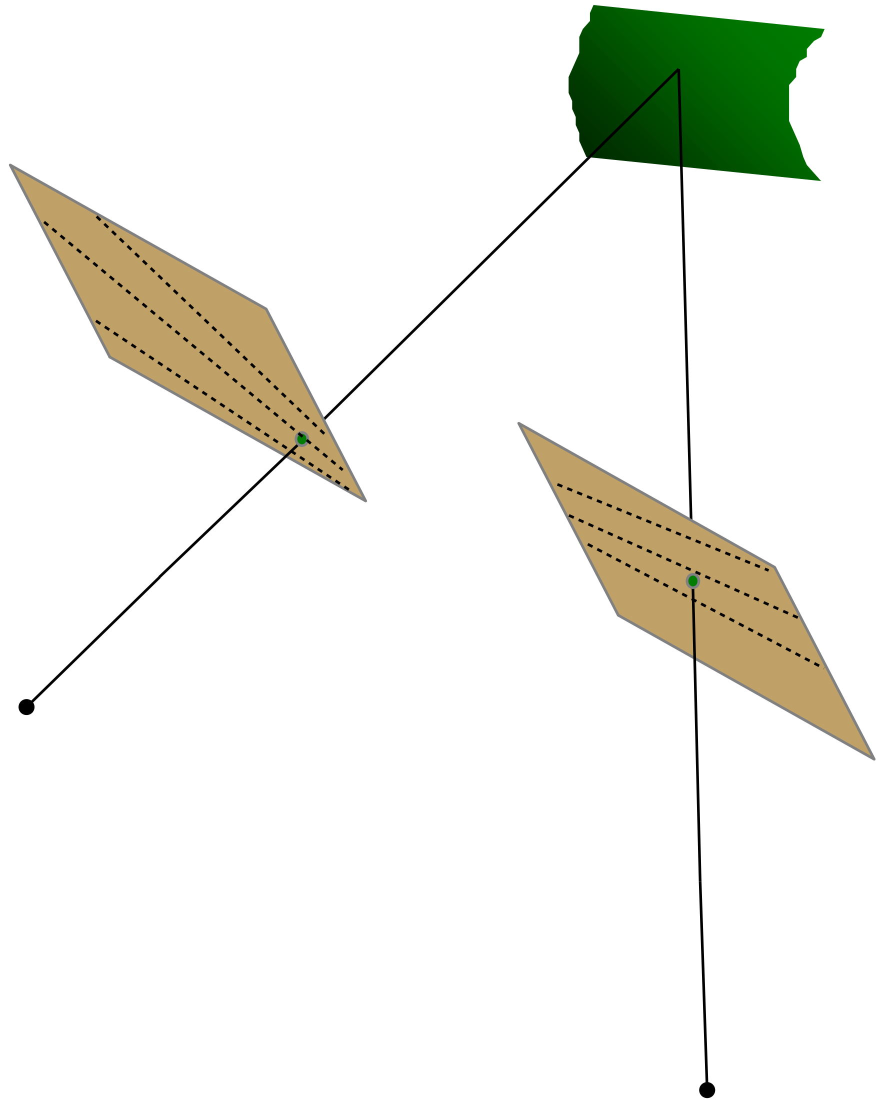
- Stereo Rectification:

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**



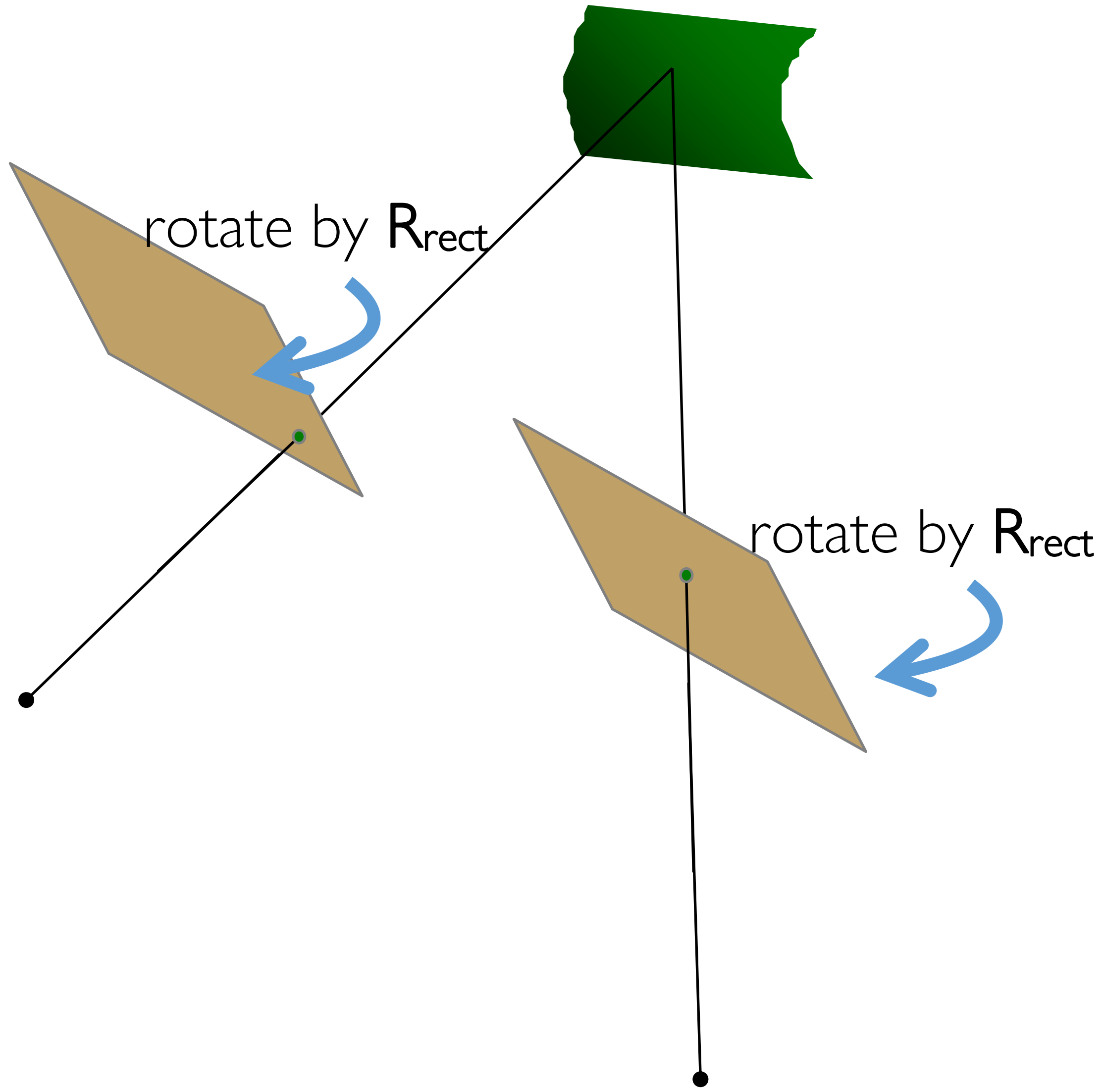
- Stereo Rectification:

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**



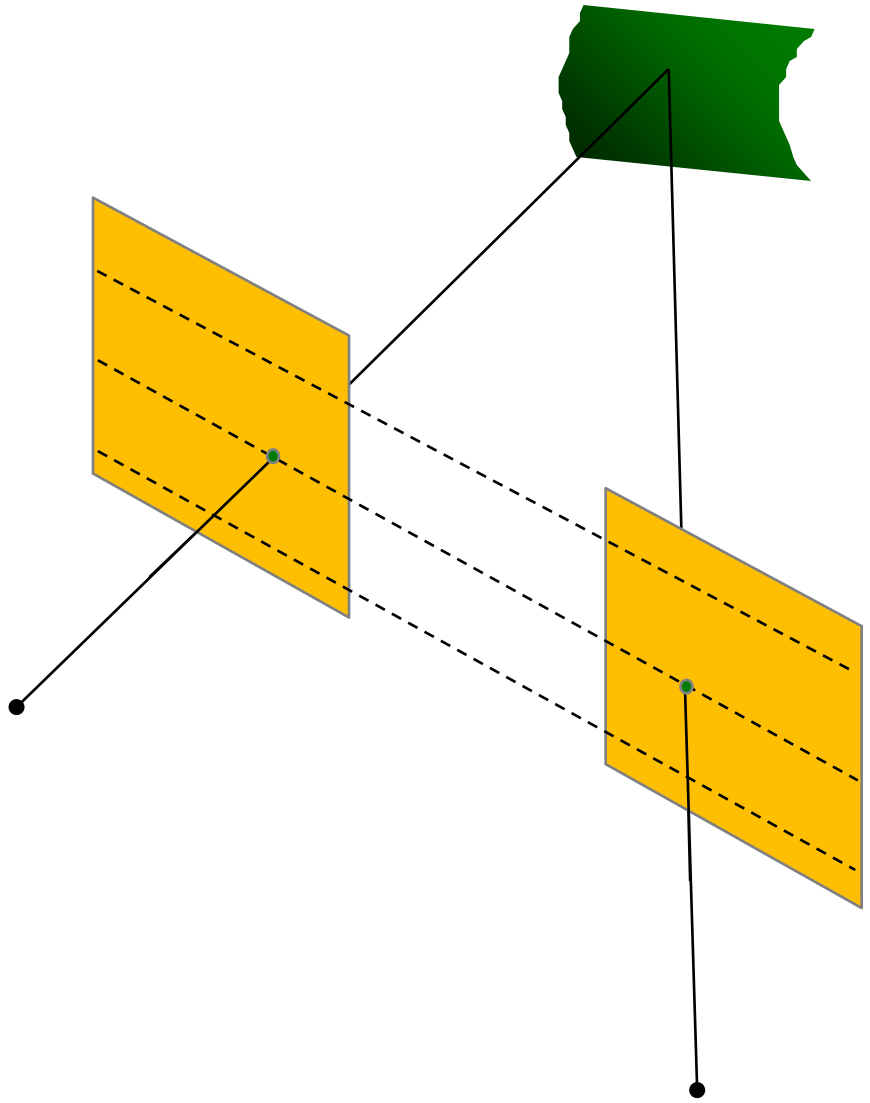
- Stereo Rectification:

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**



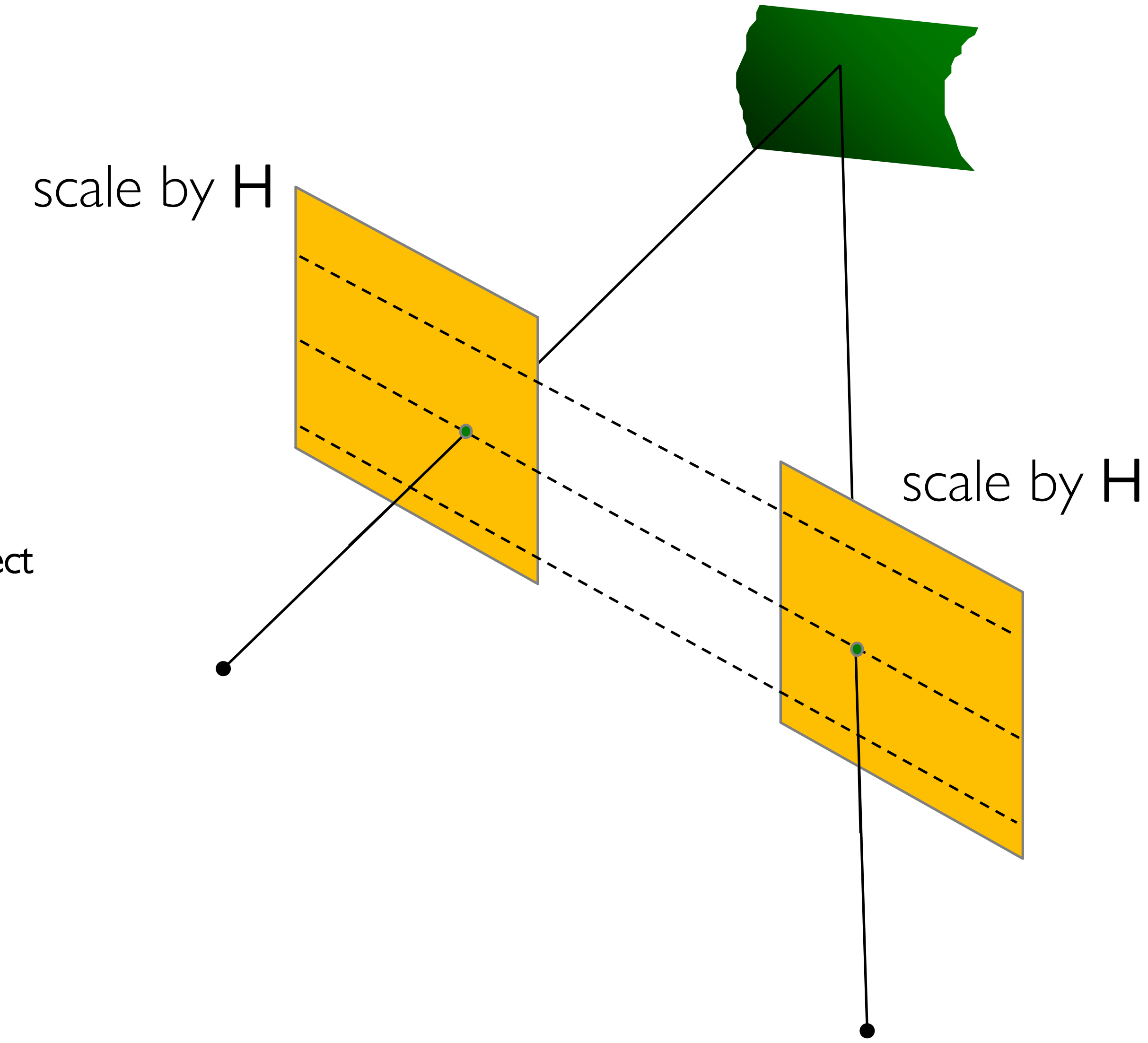
- Stereo Rectification:

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**



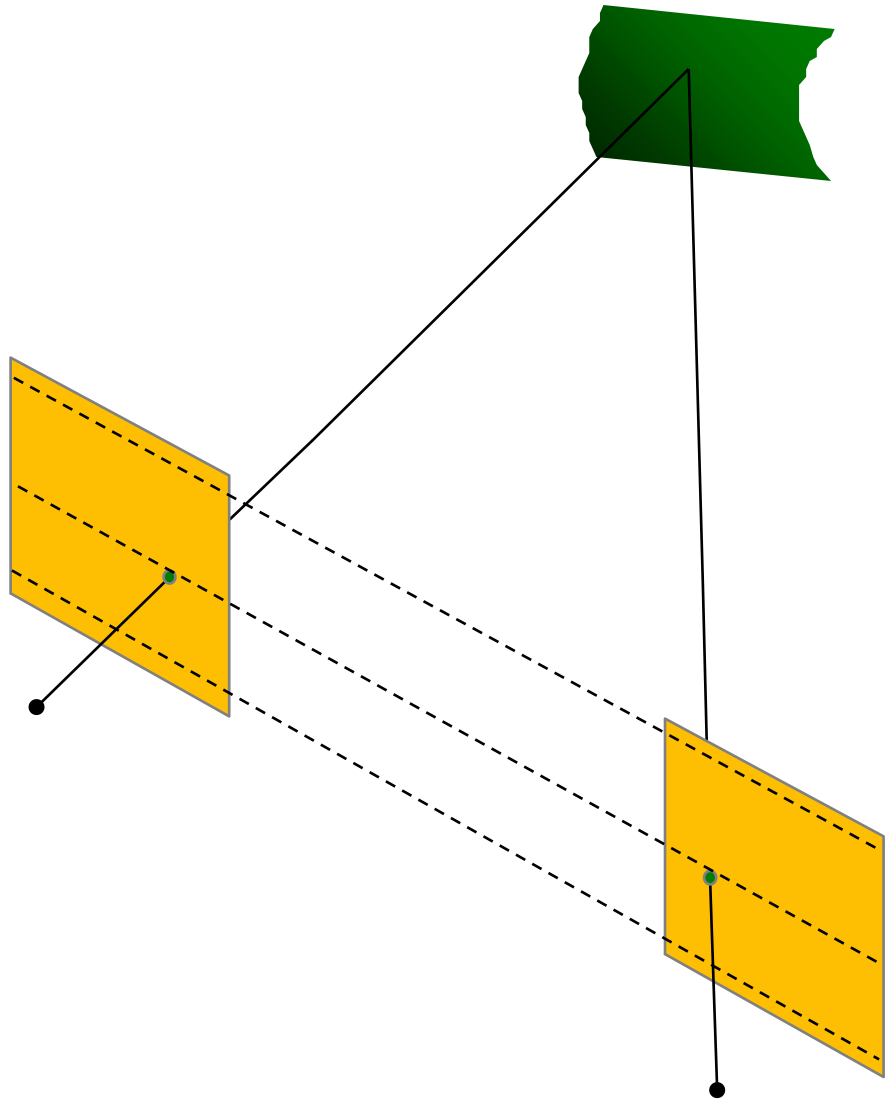
- Stereo Rectification:

1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R}_{\mathbf{rect}}$
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. **Rotate** the right camera by R
   (aligns camera coordinate system orientation only)

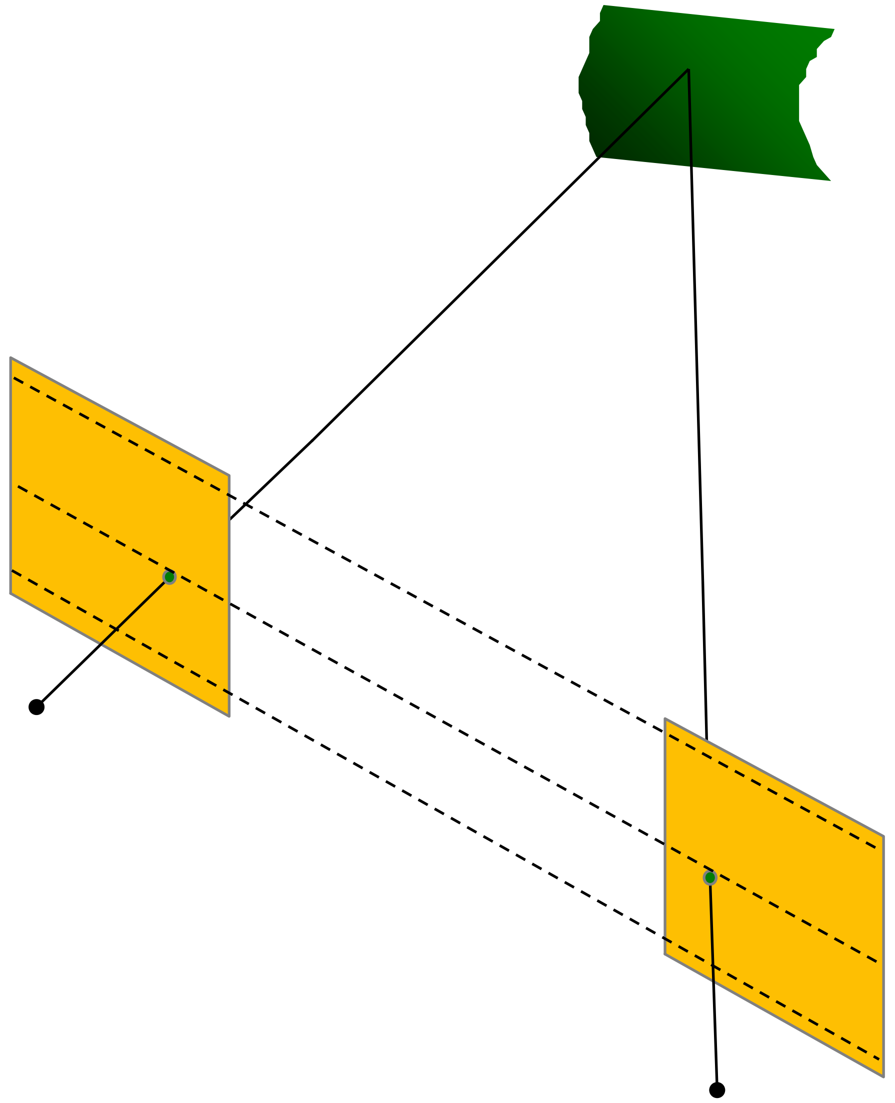
2. Rotate (**rectify**) the left camera so that the epipole is at infinity

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**



- **Step 1:** compute E to get R

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

- We get **four** solutions:

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

$$\mathbf{R}_1 = \mathbf{U}\mathbf{W}\mathbf{V}^\top \quad \mathbf{R}_2 = \mathbf{U}\mathbf{W}^\top\mathbf{V}^\top \qquad \mathbf{T}_1 = U_3 \quad \mathbf{T}_2 = -U_3$$

two possible rotations

two possible translations

Note that this is a general method to decompose R and T from E.

- We get **four** solutions:

$$\mathbf{R}_1 = \mathbf{U}\mathbf{W}\mathbf{V}^\top \qquad \mathbf{R}_1 = \mathbf{U}\mathbf{W}\mathbf{V}^\top$$
$$\mathbf{T}_1 = U_3 \qquad \mathbf{T}_2 = -U_3$$

$$\mathbf{R}_2 = \mathbf{U}\mathbf{W}^\top\mathbf{V}^\top \qquad \mathbf{R}_2 = \mathbf{U}\mathbf{W}^\top\mathbf{V}^\top$$
$$\mathbf{T}_2 = -U_3 \qquad \mathbf{T}_1 = U_3$$

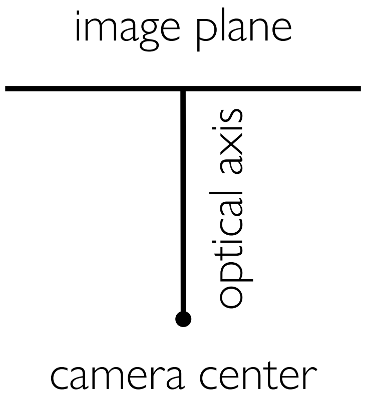
- Which one do we choose?

- Compute determinant of R, valid solution must be equal to 1
(note: det(R) = -1 means rotation and reflection)

- Compute 3D point using triangulation, valid solution has positive Z value
(note: negative Z means point is behind the camera)

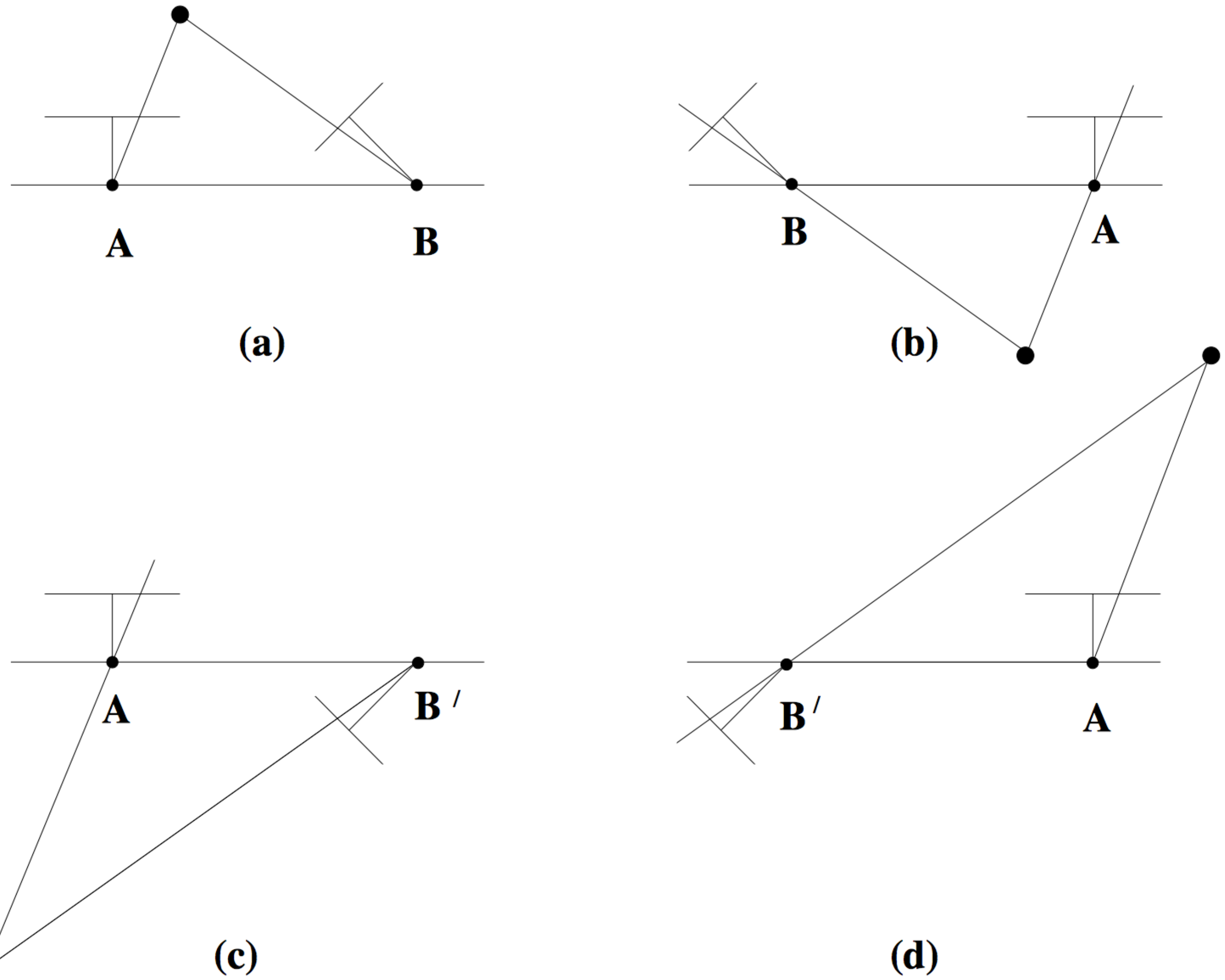
- Let’s visualize the four configurations…

camera Icon

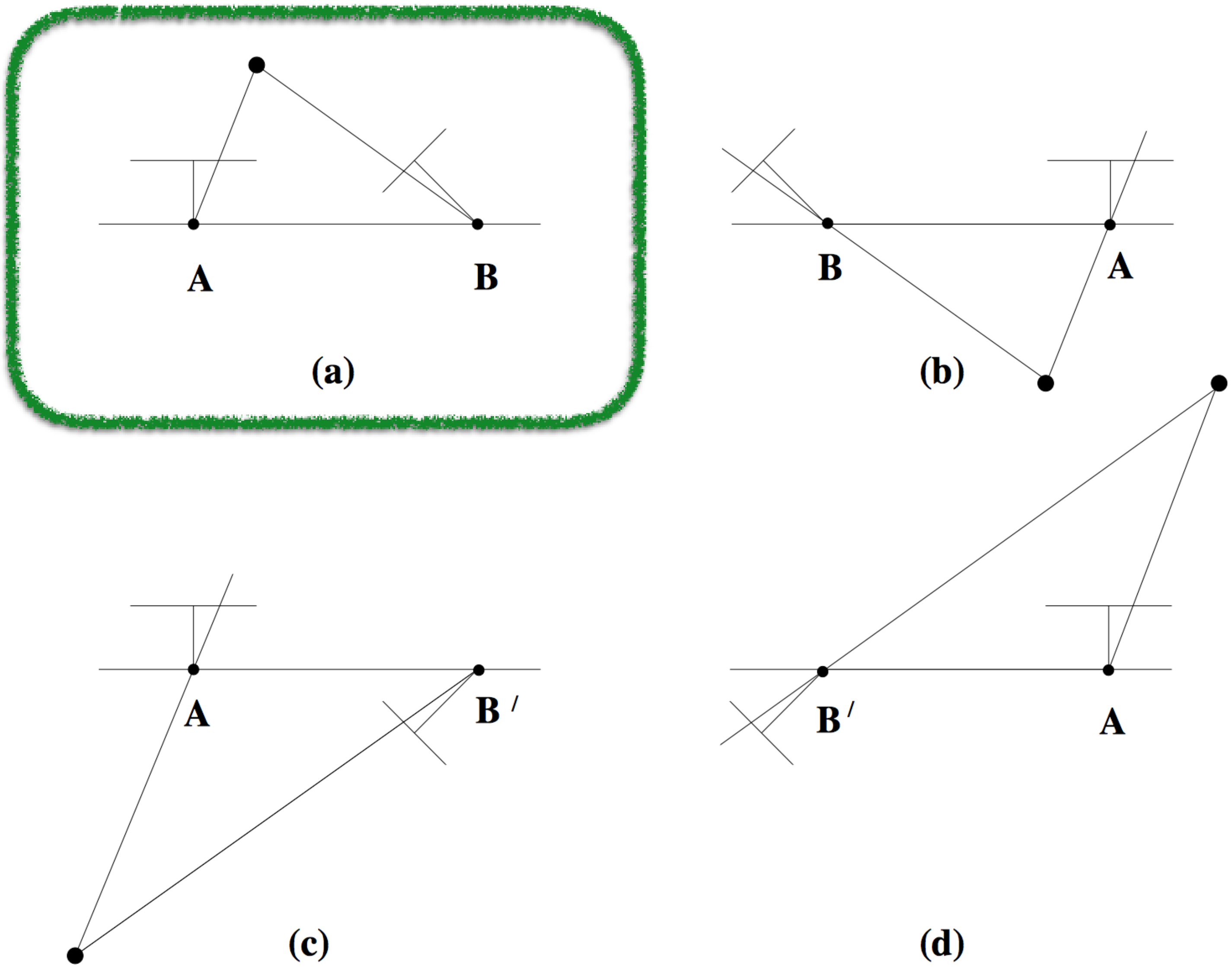


- Find the configuration where the point is in front of both cameras

- Find the configuration where the points is in front of both cameras

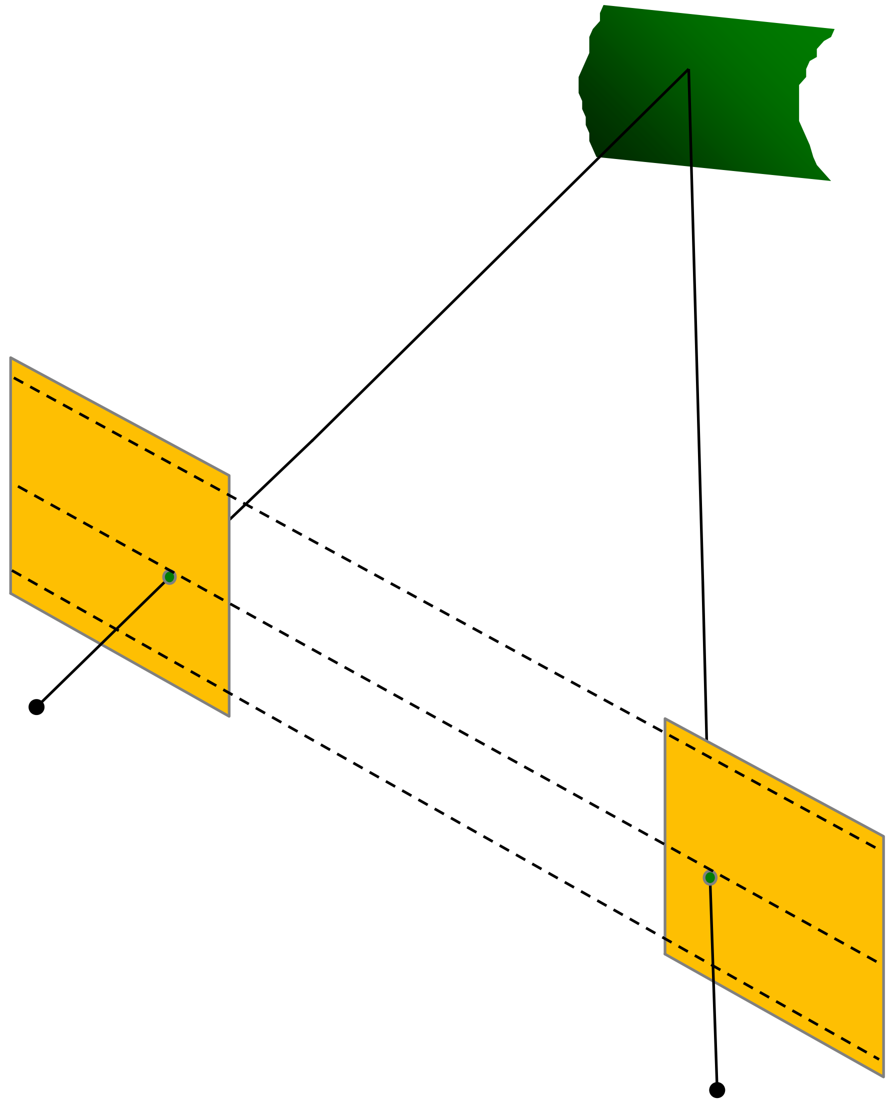


- Find the configuration where the points is in front of both cameras

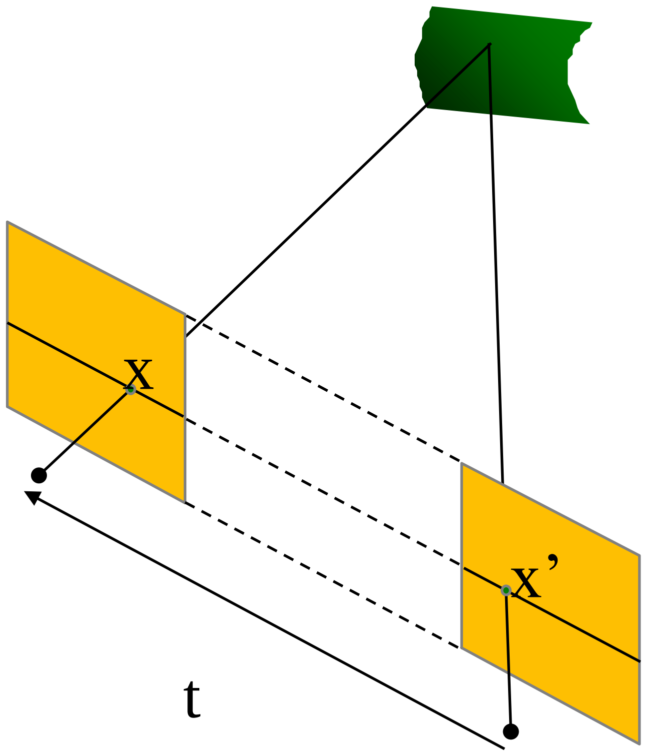


- Stereo Rectification:

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**



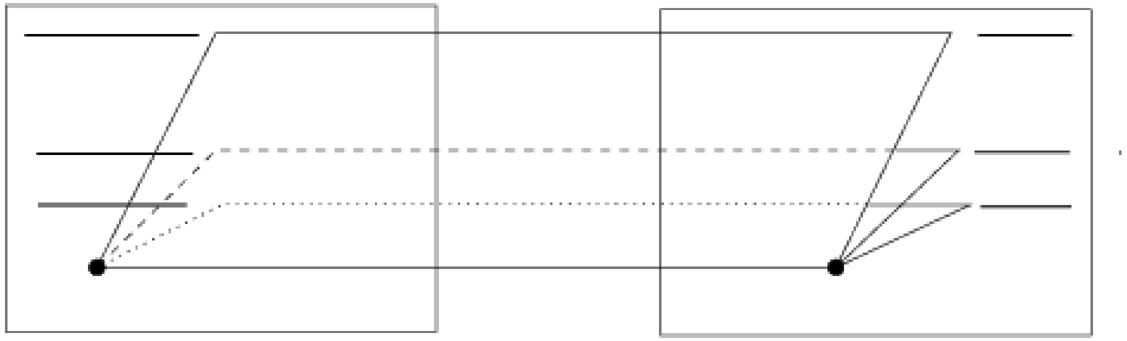
- When are epipolar lines horizontal?

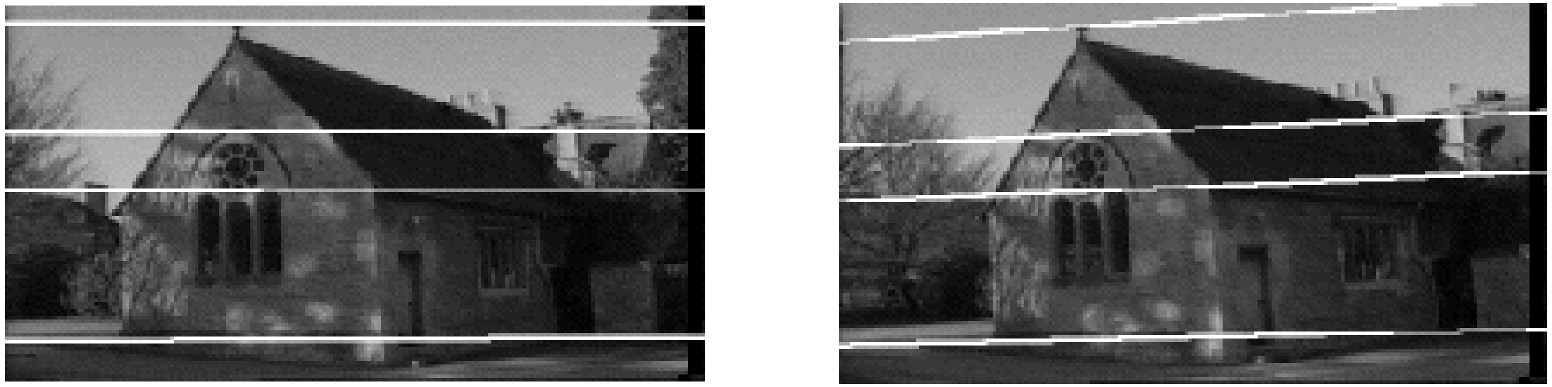


- When this relationship holds:

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

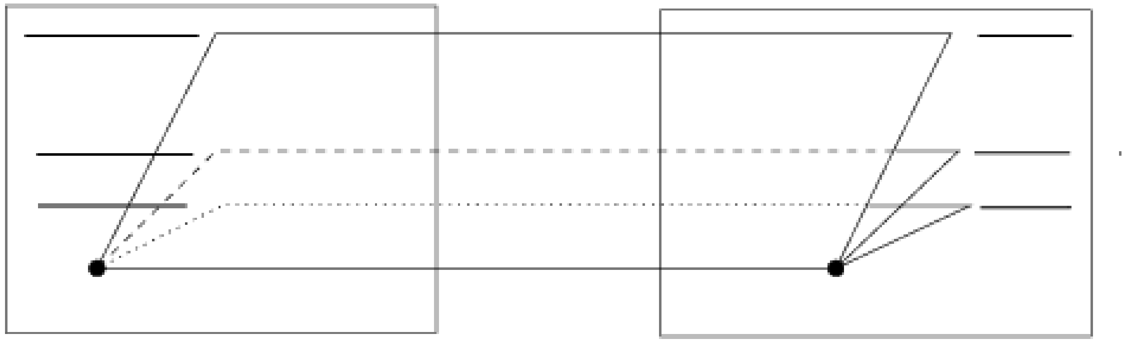
- Parallel cameras

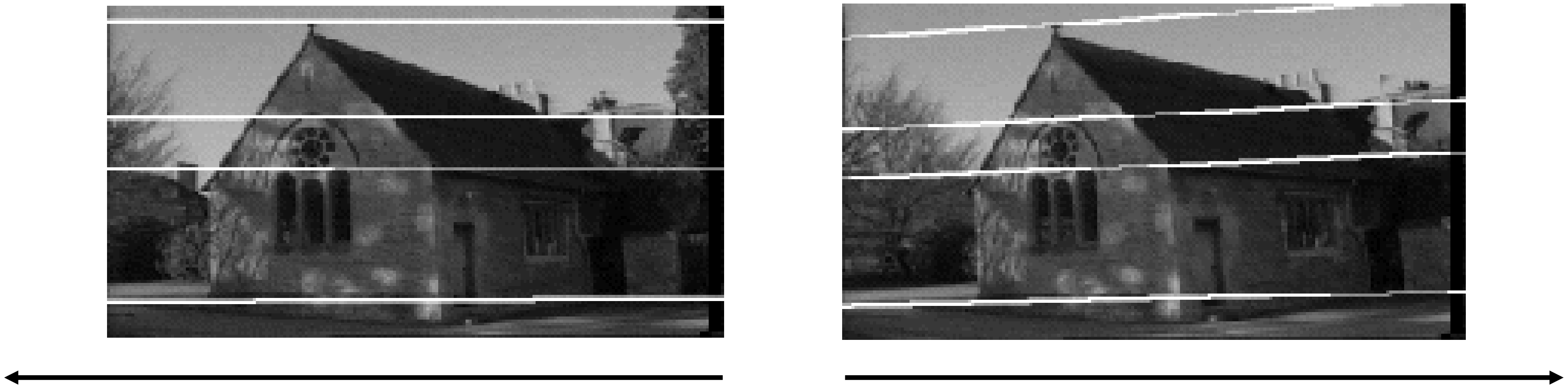




- Where is the epipole?

- Parallel cameras





- Epipole at infinity

- Setting the epipole to infinity (building R_{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** (using SVD on 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 $\boldsymbol{r}_1 = \boldsymbol{e}_1 = \frac{T}{||T||}$ and $\boldsymbol{r}_2$ $\boldsymbol{r}_3$ orthogonal

- Then $R_{\text{rect}}\boldsymbol{e}_1 = \left[\begin{array}{c} \boldsymbol{r}_1^\top \boldsymbol{e}_1 \\ \boldsymbol{r}_2^\top \boldsymbol{e}_1 \\ \boldsymbol{r}_3^\top \boldsymbol{e}_1 \end{array}\right] = \left[\begin{array}{c} ? \\ ? \\ ? \end{array}\right]$

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

- Where is this point located on the image plane? At x-infinity

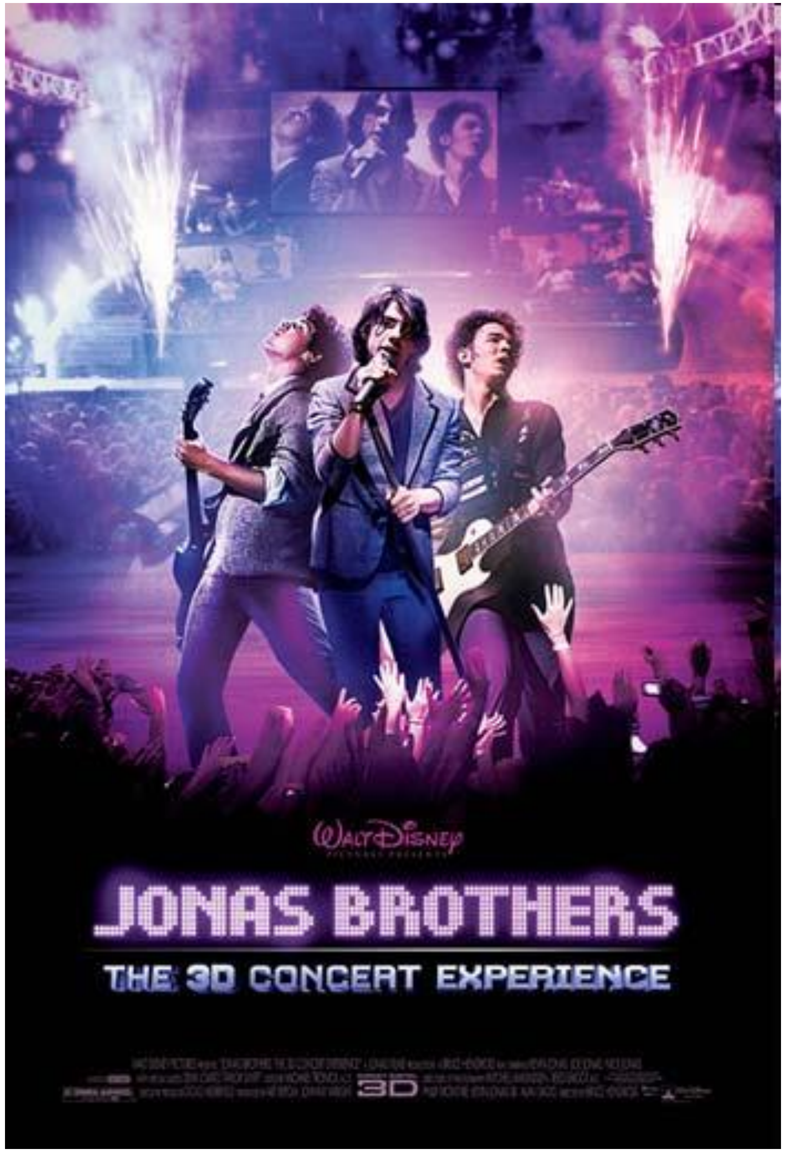
- Stereo Rectification Algorithm

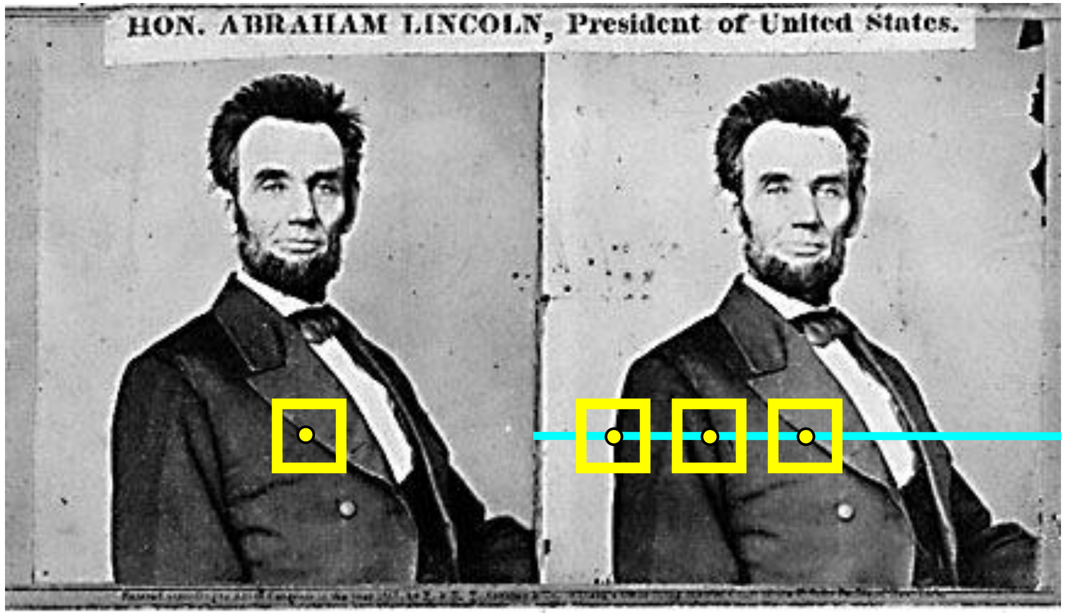
1. Estimate $\mathbf{E}$ using the 8 point algorithm (SVD)
2. Estimate the epipole $\mathbf{e}$ (SVD of $\mathbf{E}$)
3. Build R_{rect} from $\mathbf{e}$
4. Decompose $\mathbf{E}$ into R and T
5. Set $R_1 = R_{rect}$ and $R_2 = RR_{rect}$
6. Rotate each left camera point (warp image) $[x'\ y'\ z'] = R_1 [x\ y\ z]$
7. Rectified points as $\mathbf{p} = f/z'[x'\ y'\ z']$
8. Repeat 6 and 7 for right camera points using R_2

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

Teesta suspension bridge-Darjeeling, India

This is how 3D movies work

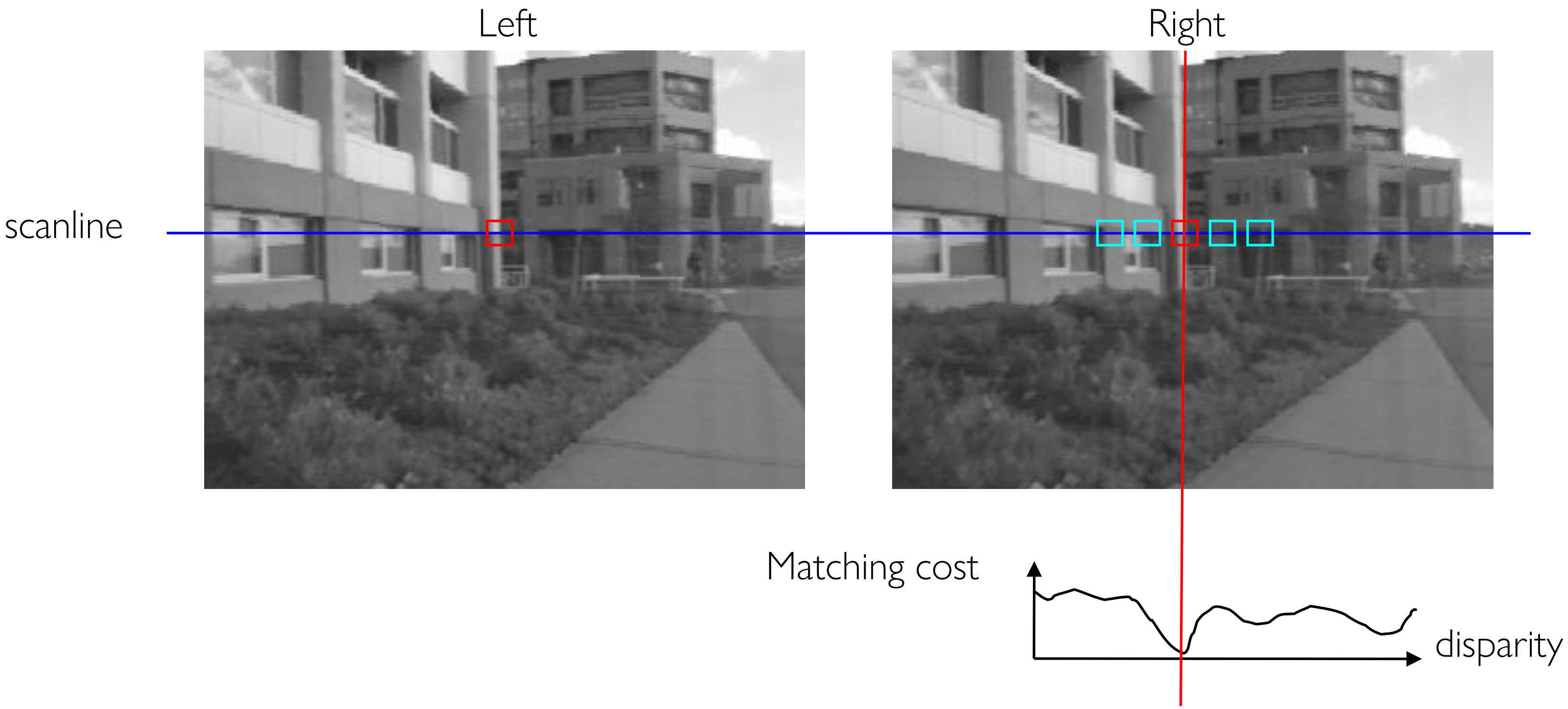




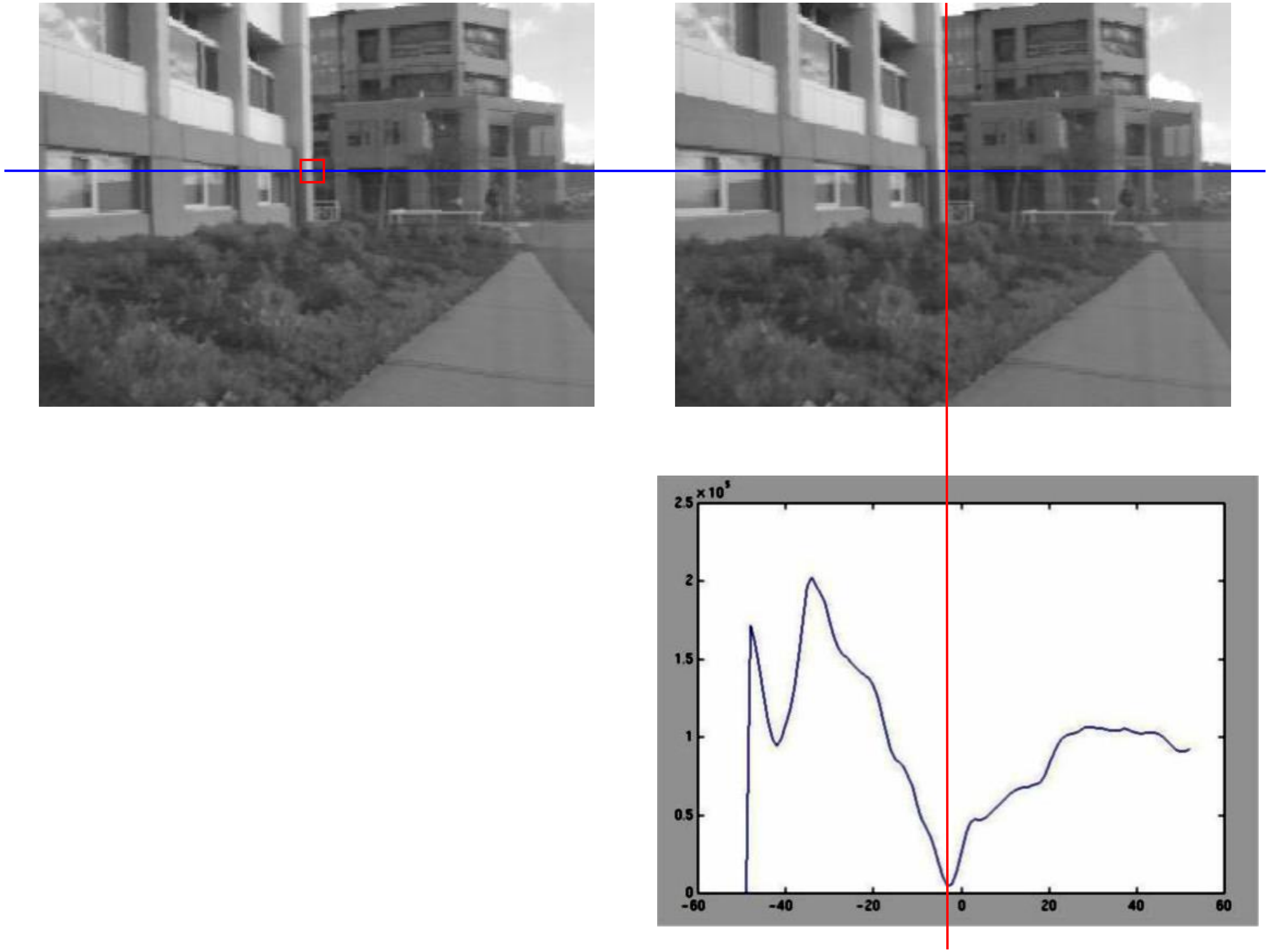
1. Rectify images
   (make epipolar lines horizontal)
2. For each pixel
   - Find epipolar line
   - **Scan line for best match**
   - Compute depth from disparity ($Z = \frac{bf}{d}$)

how would you do this?

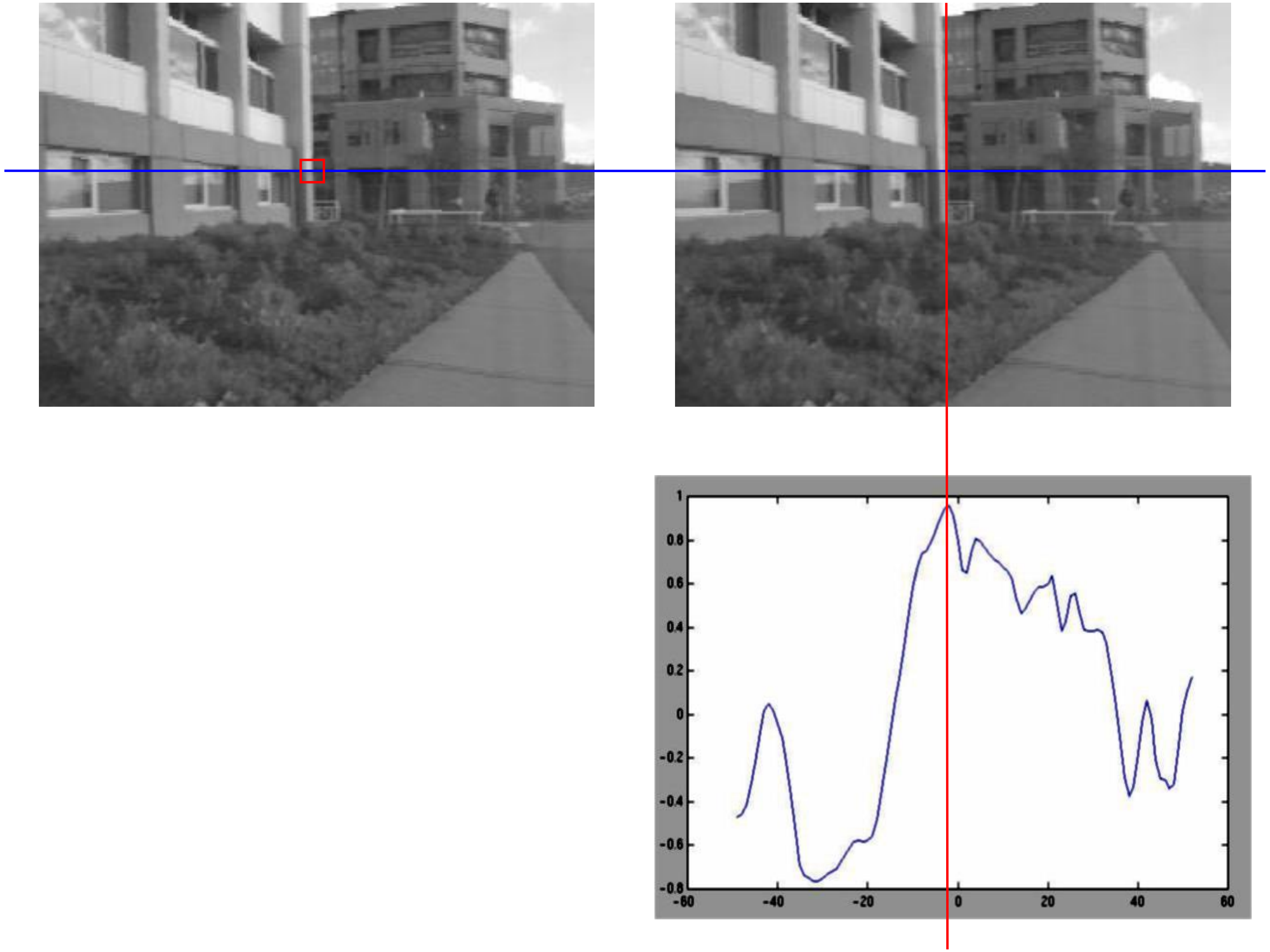
Stereo Block Matching



- 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



SSD



Normalized cross-correlation

What is the best method?

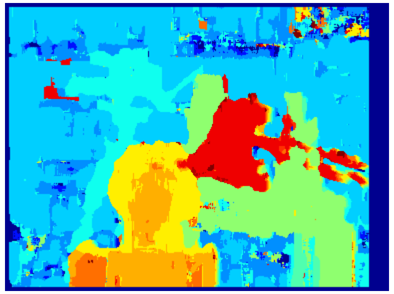
- It depends on whether you care about speed or invariance.

- Zero-mean: fastest, very sensitive to local intensity.

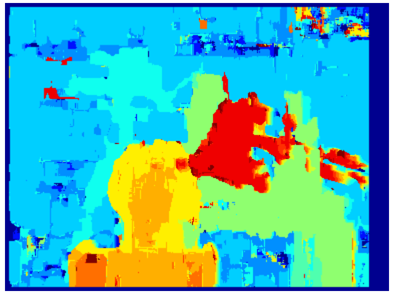
- Sum of squared differences: medium speed, sensitive to intensity offsets.

- Normalized cross-correlation: slowest, invariant to contrast and brightness.

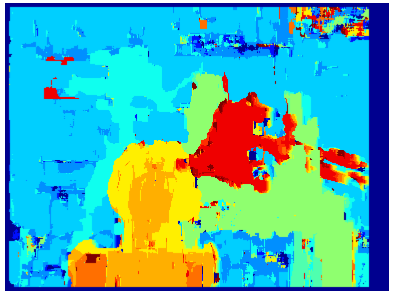
Similarity Measure	Formula
Sum of Absolute Differences (SAD)	$\sum_{(i,j)\in W} \lvert I_1(i,j) - I_2(x+i, y+j) \rvert$
Sum of Squared Differences (SSD)	$\sum_{(i,j)\in W} \left(I_1(i,j) - I_2(x+i, y+j)\right)^2$
Zero-mean SAD	$\sum_{(i,j)\in W} \lvert I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i,y+j) + \bar{I}_2(x+i,y+j)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

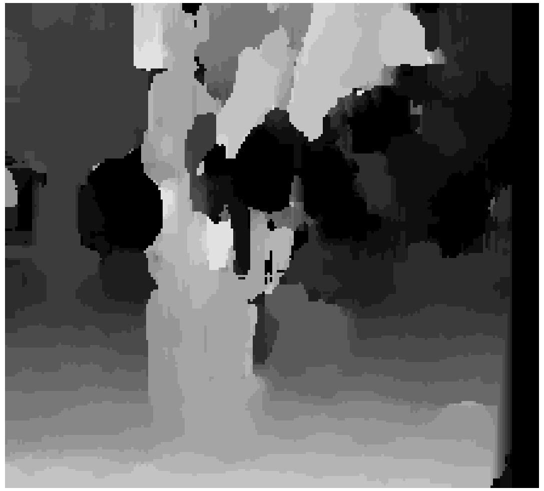


NCC

Ground truth

Effect of window size

W = 3



W = 20

Smaller window

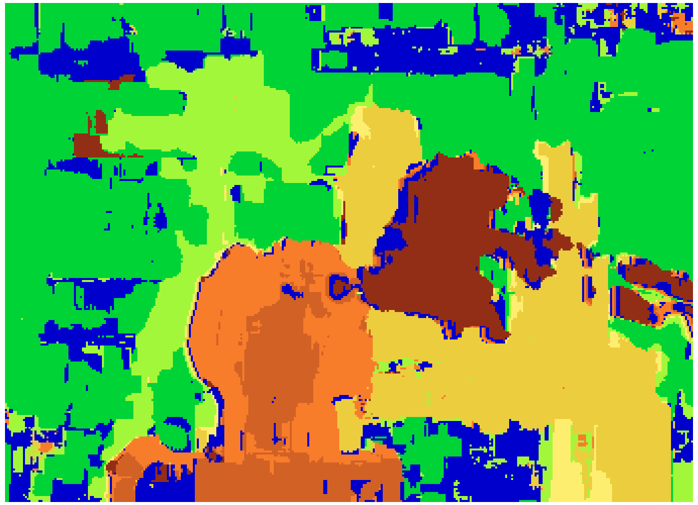
+ More detail
- More noise

Larger window

+ Smoother disparity maps
- Less detail
- Fails near boundaries

Improving stereo matching

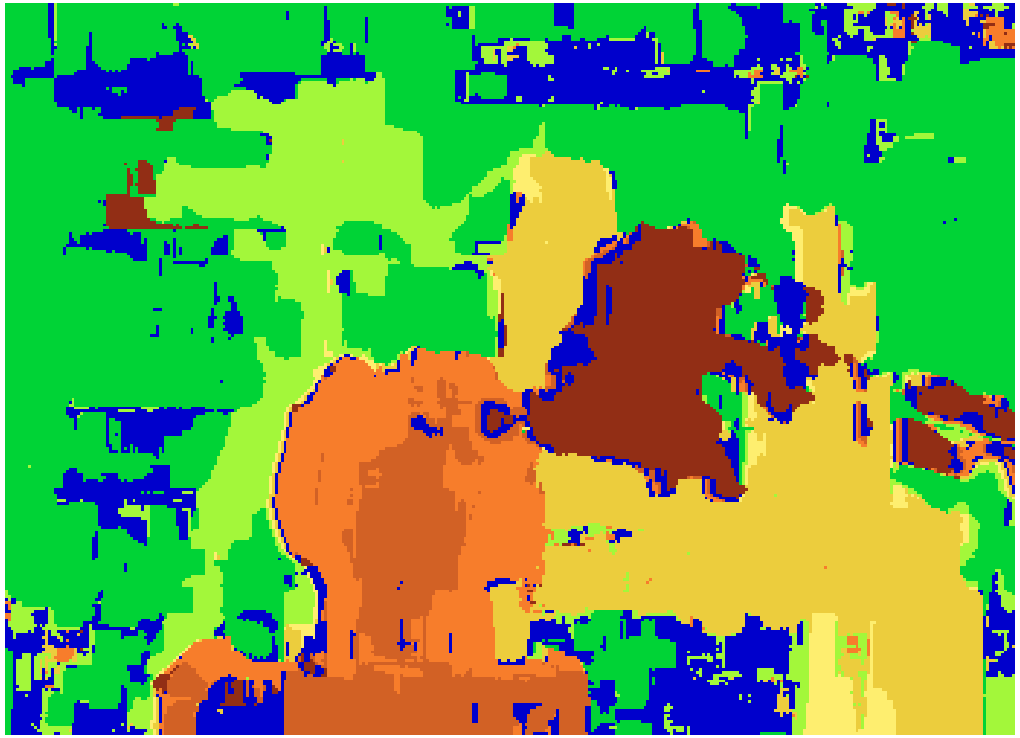
Block matching



Ground truth

- What are some problems with the result?

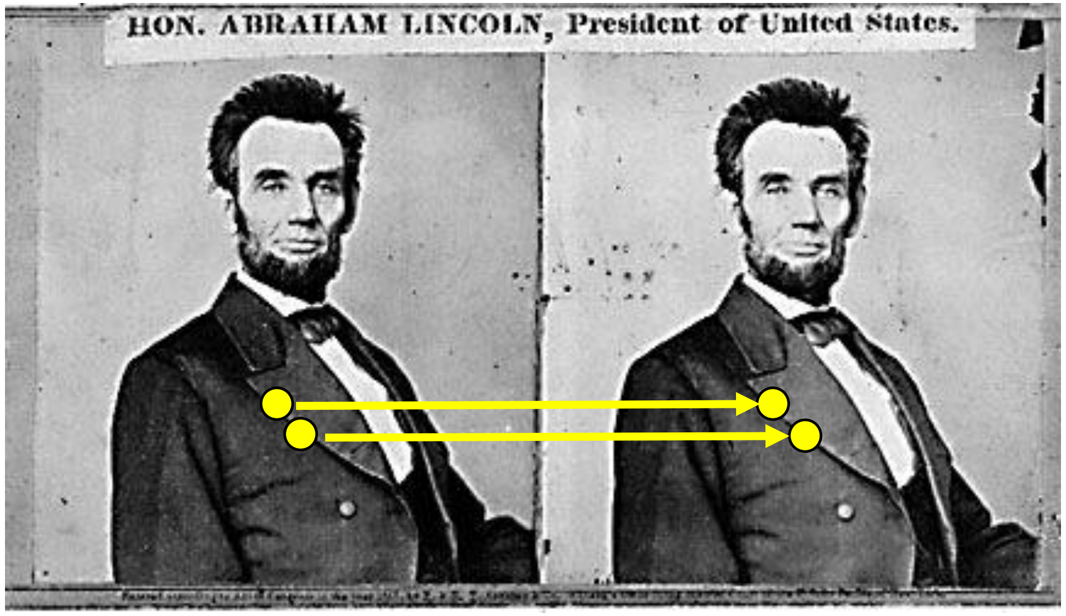
- How can we improve depth estimation?



- 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?
    - Match quality
        - Want each pixel to find a good match in the other image
    - Smoothness
        - If two pixels are adjacent, they should (usually) move about the same amount

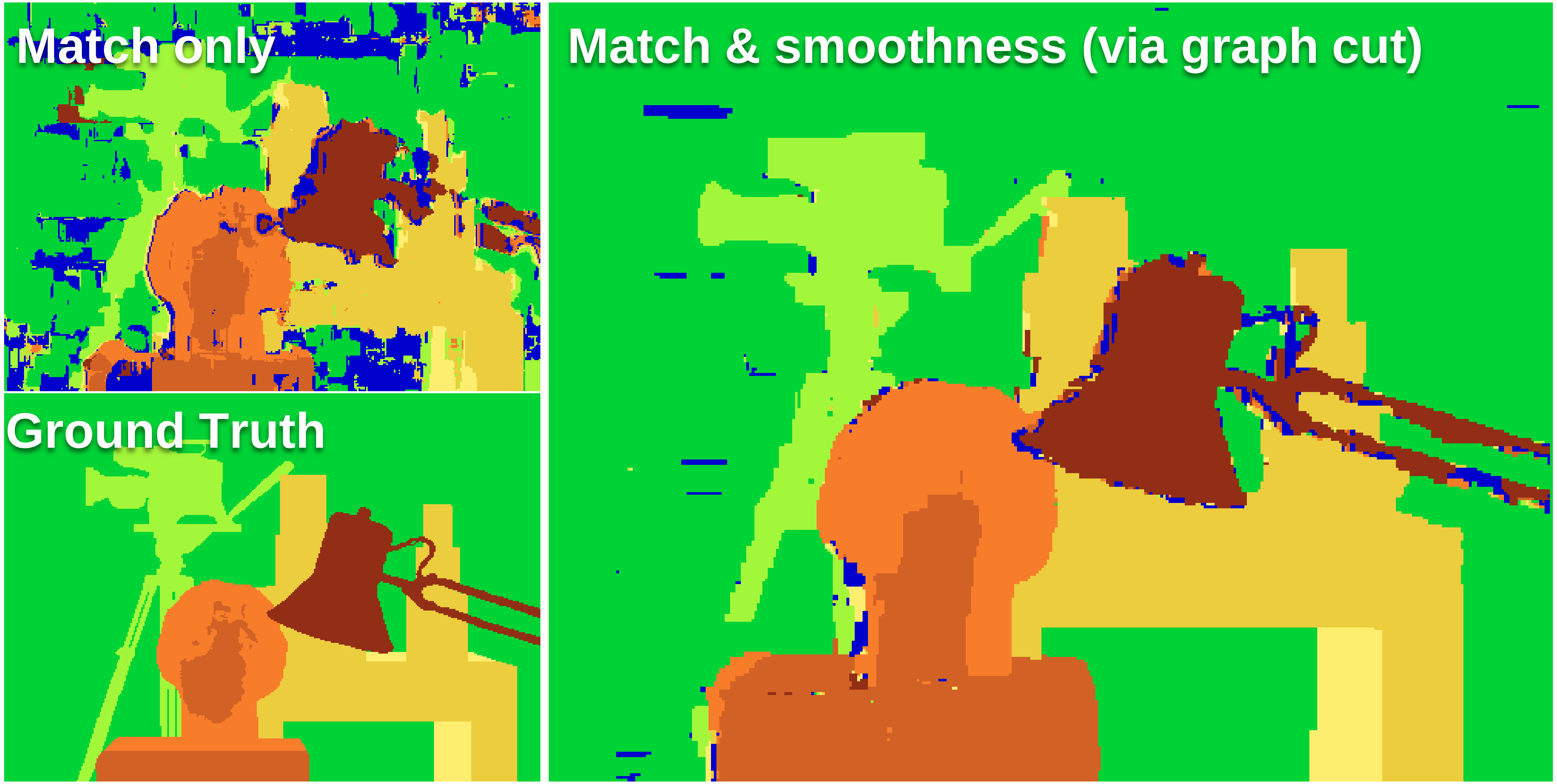


Energy function
(for one pixel)

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

Want each pixel to find a good match in the other image (block matching result)

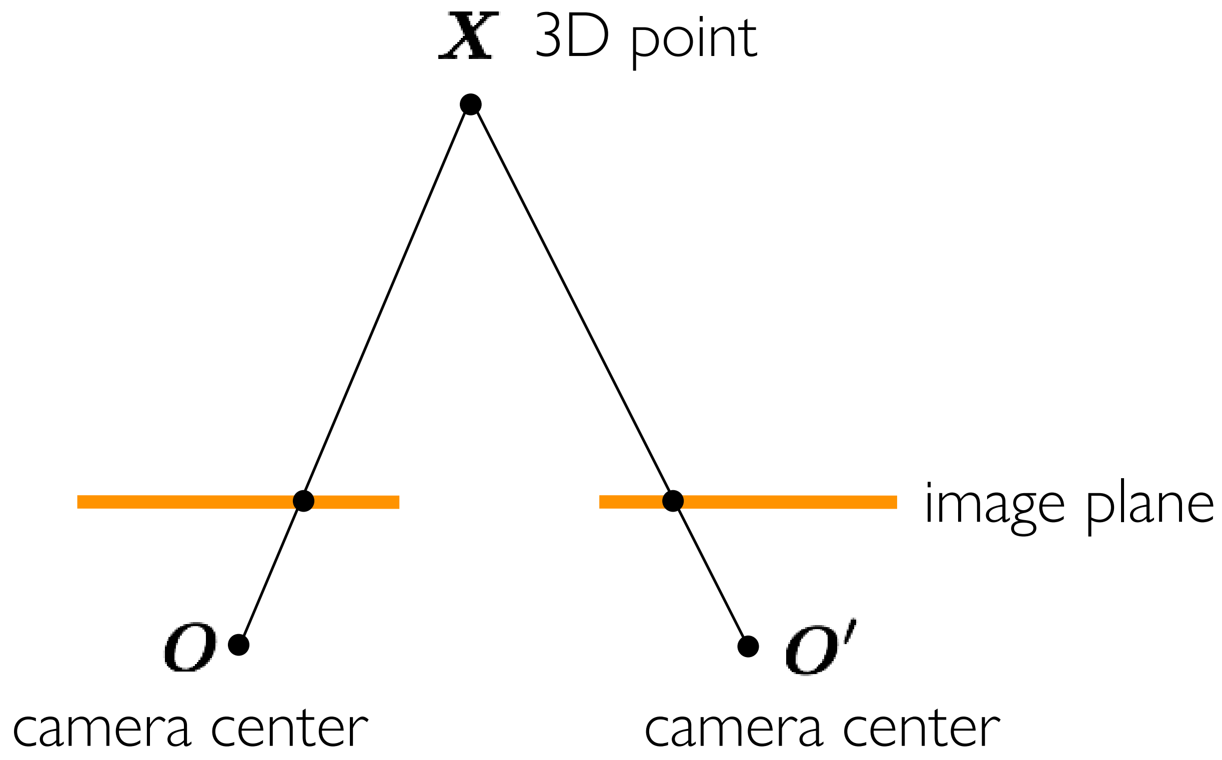
Adjacent pixels should (usually) move about the same amount (smoothness function)



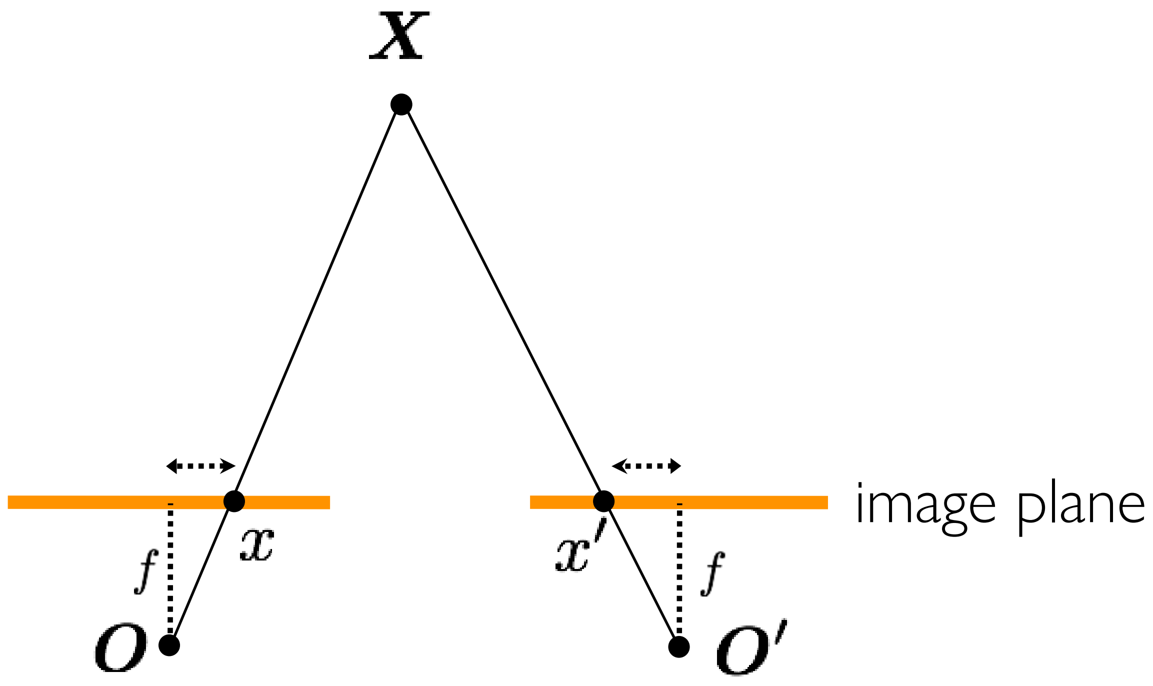
Y. Boykov, O. Veksler, and R. Zabih, Fast Approximate Energy Minimization via Graph Cuts, PAMI 2001

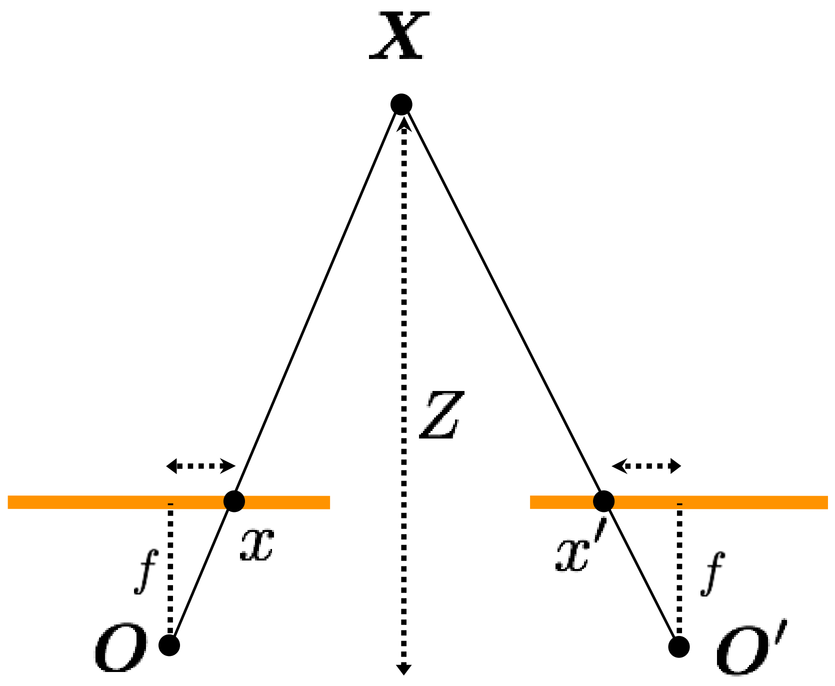
- All of these cases remain difficult, what can we do?

Depth estimation
(Triangulation with rectified images)

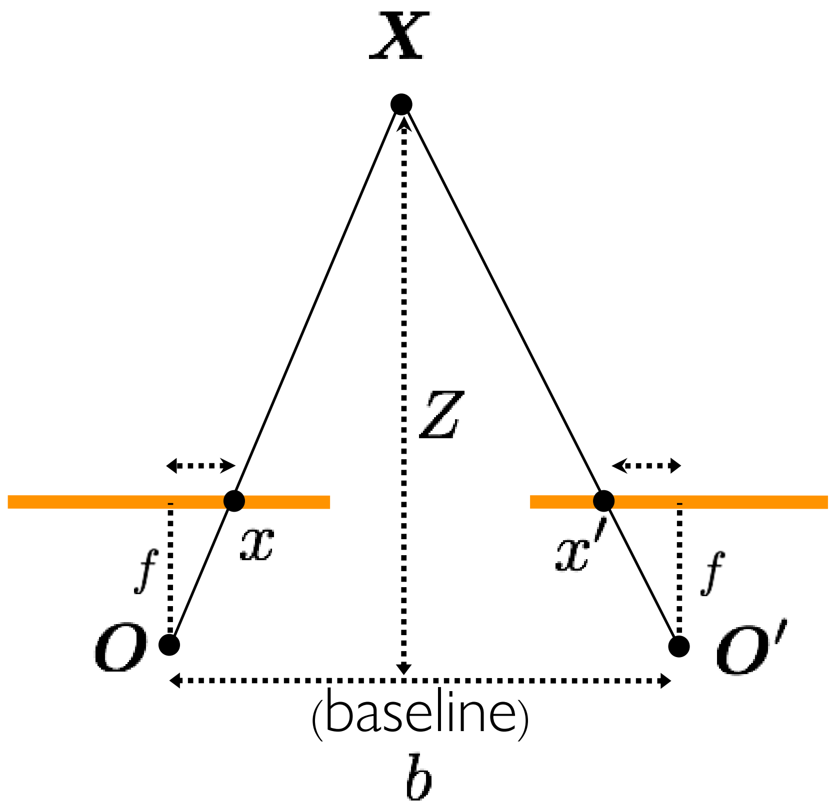


X 3D point
image plane
O
camera center
O'
camera center


X
x
x'
f
f
O
O'
image plane

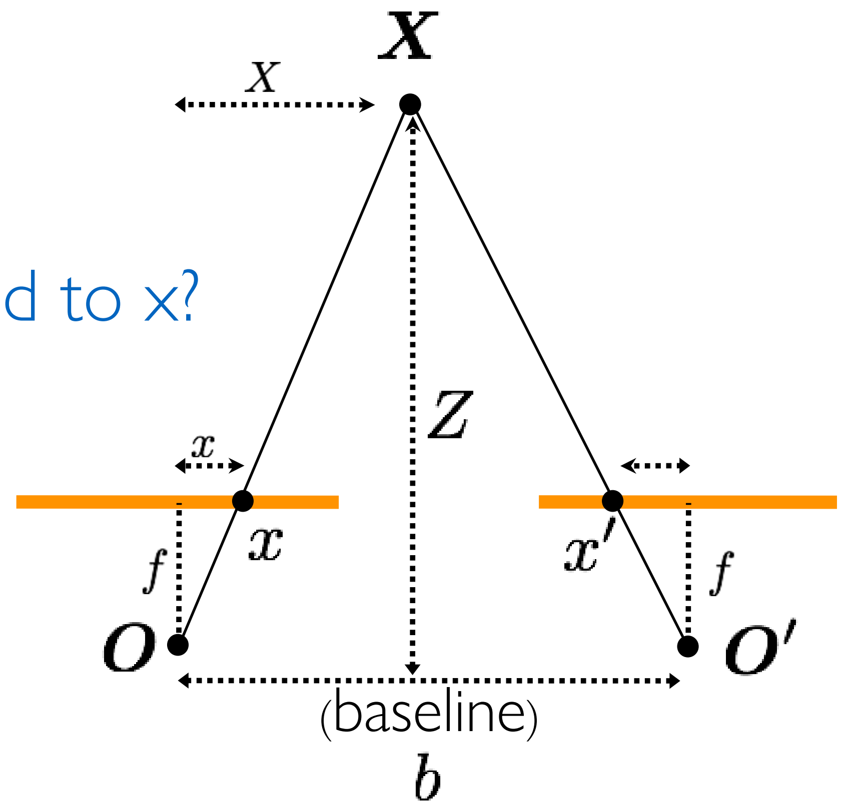


X
Z
x
x'
f
f
O
O'

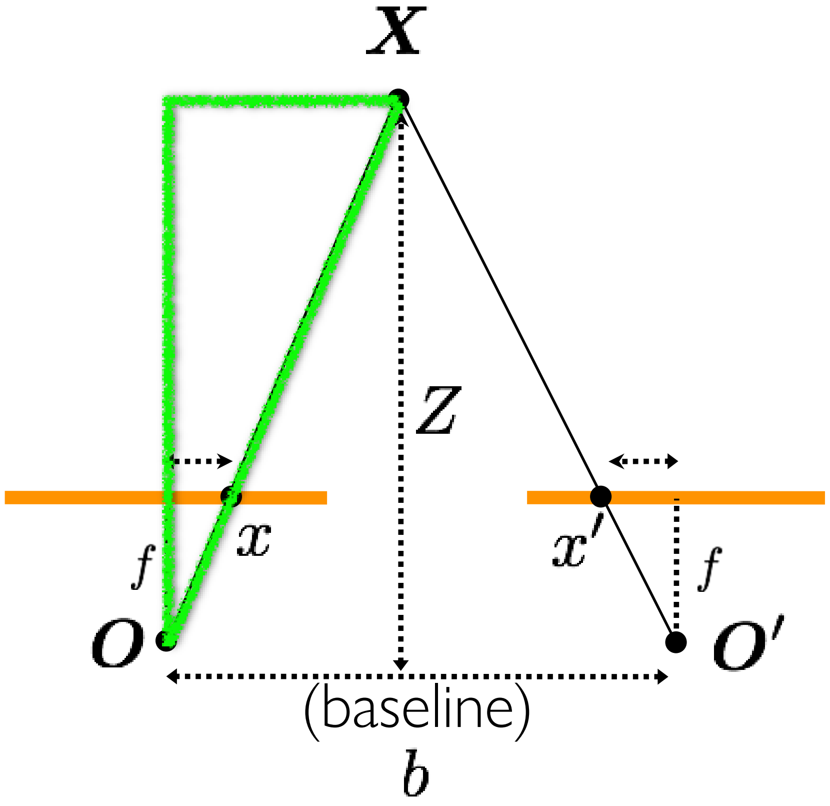


$\boldsymbol{X}$
Z
x
x'
f
f
$\boldsymbol{O}$
$\boldsymbol{O'}$
(baseline)
b

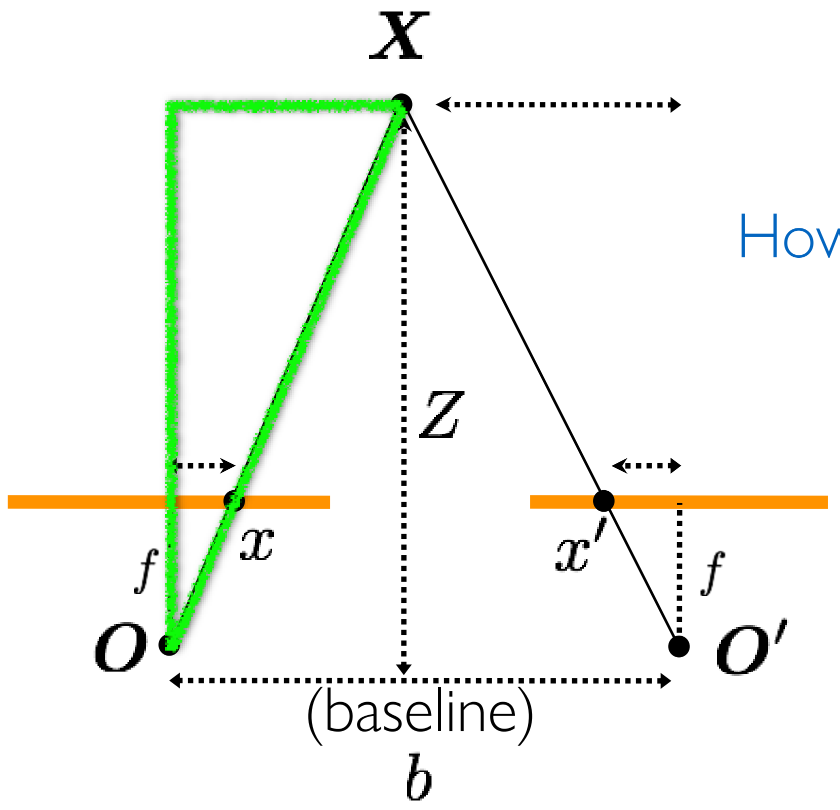
How is X related to x?



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

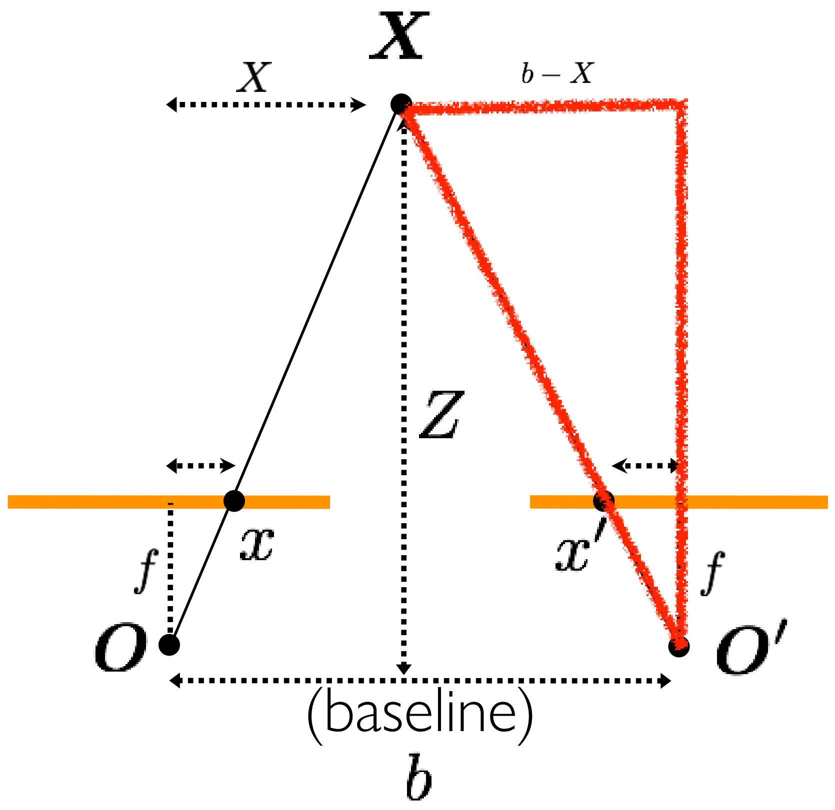


$$\frac{X}{Z} = \frac{x}{f}$$



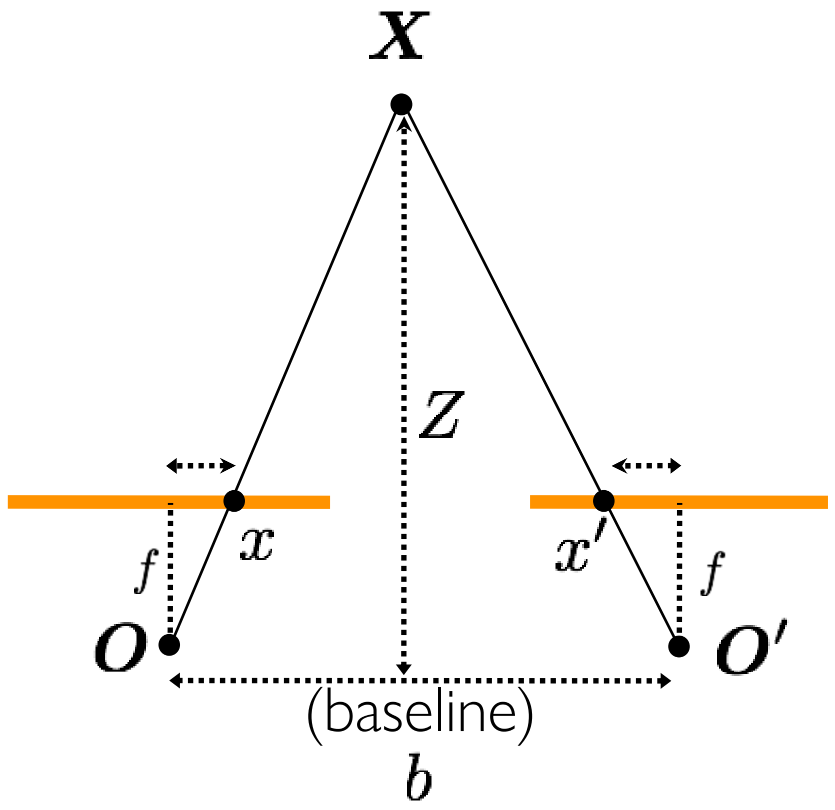
How is X related to x'?

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



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

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



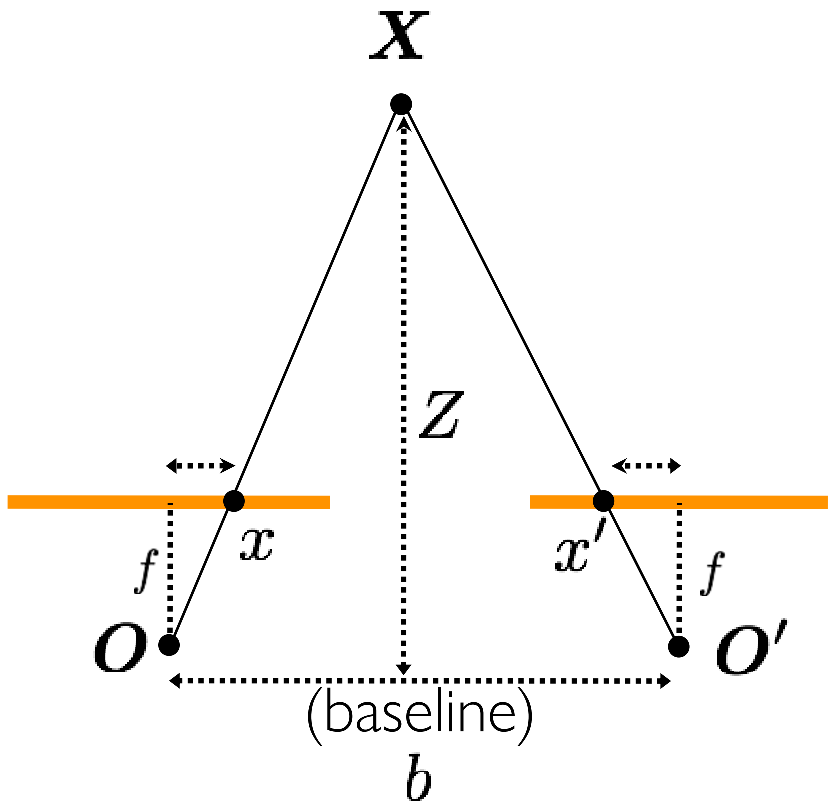
$$\frac{b - X}{Z} = \frac{x'}{f}$$

Disparity

$$d = x - x'$$ (w.r.t 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