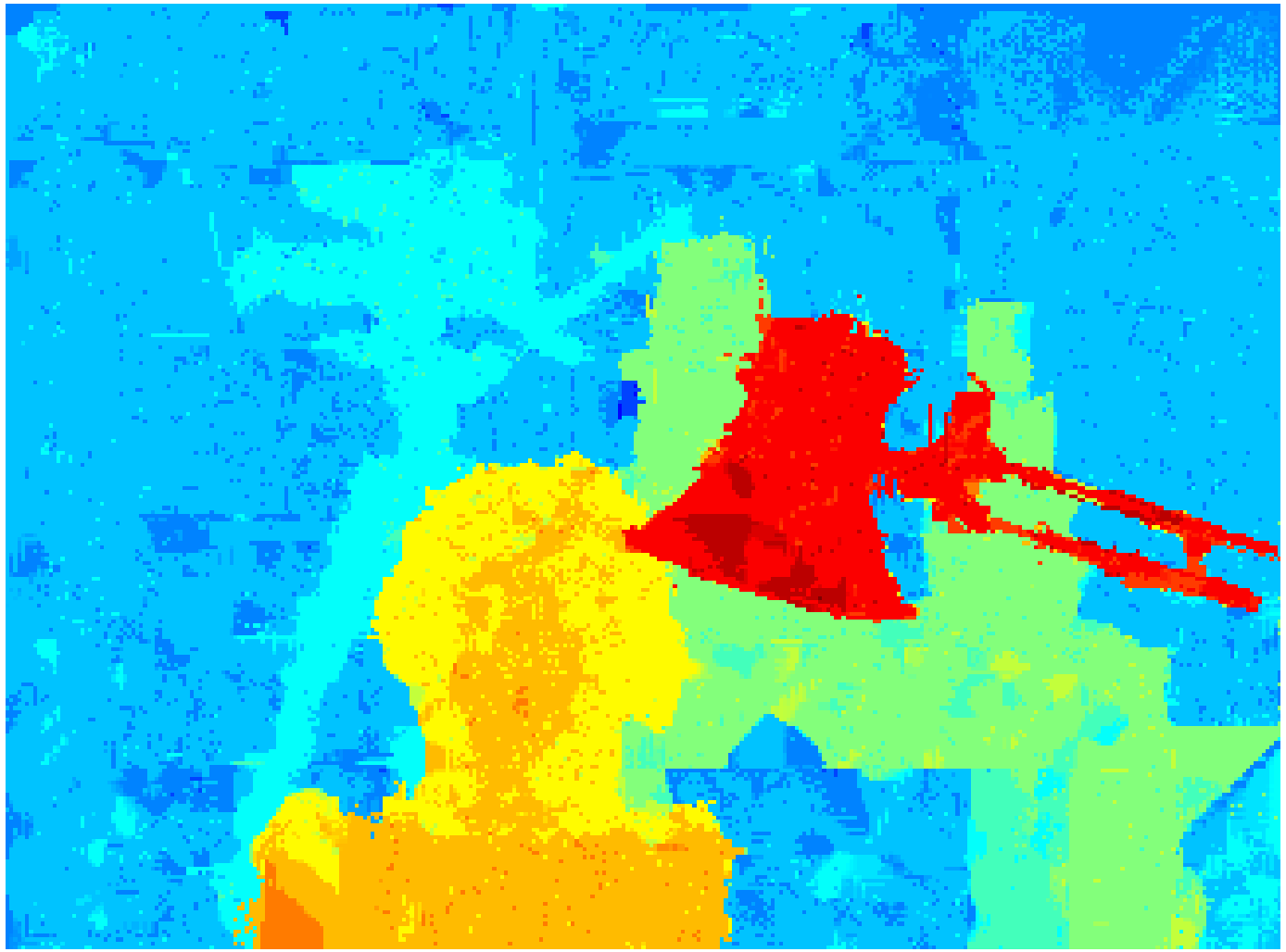
Disparity and passive stereo



15-463, 15-663, 15-862
Computational Photography
Fall 2020, Lecture 22

http://graphics.cs.cmu.edu/courses/15-463

Course announcements

- Homework assignment 6 posted.
  - Due on December 11.

- Office hours this week:
  - Monday, 3:30 – 5:30 pm, Yannis.
  - Tuesday, 3:30 – 5:30 pm, Byeongjoo.
  - Wednesday, 3:30 – 5:30 pm, Jenny.

- Materials for last Friday's reading group posted on Piazza/Canvas.

- No reading group this week (Thanksgiving).

Overview of today's lecture

- Leftover from previous lecture.
- Revisiting triangulation.
- Disparity.
- Stereo rectification.
- Stereo matching.
- Epipolar imaging.

Slide credits

Many of these slides were adapted directly from:

- Kris Kitani (16-385, Spring 2017).
- Srinivasa Narasimhan (16-820, Spring 2017).
- Mohit Gupta (Wisconsin).
- James Tompkin (Brown).

Revisiting triangulation

How would you reconstruct 3D points?

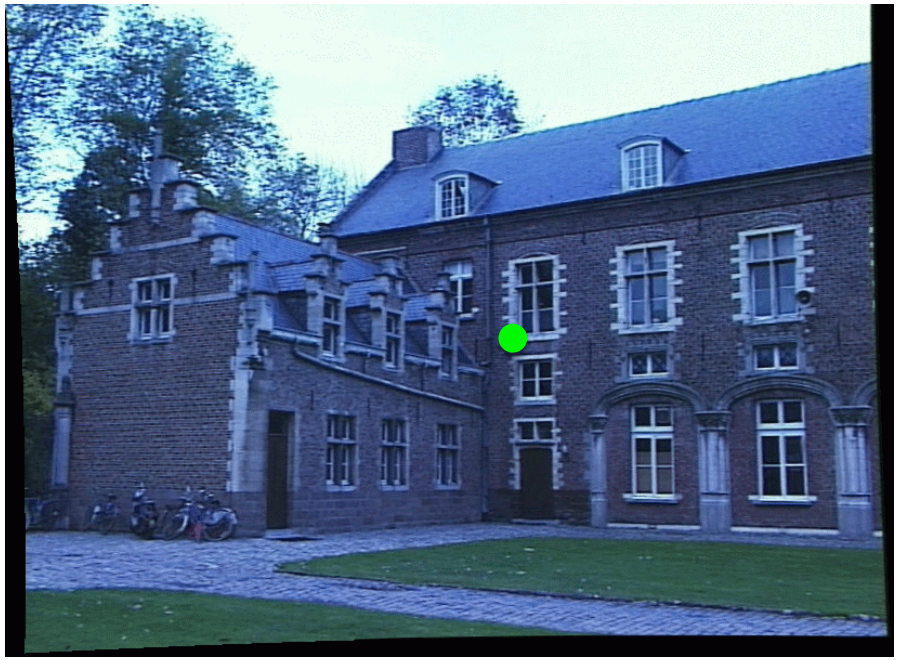


Left image



Right image

How would you reconstruct 3D points?



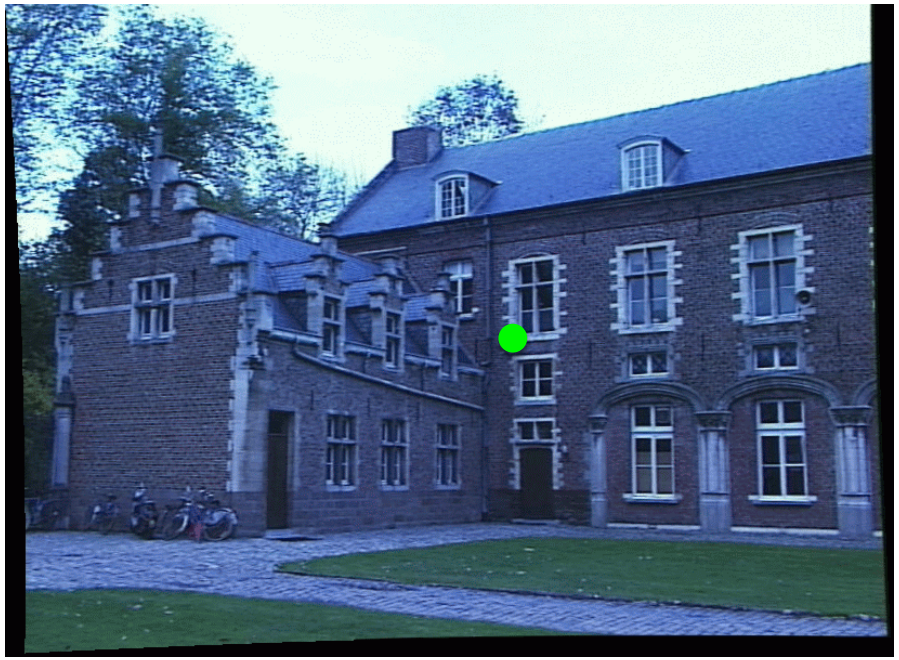
Left image



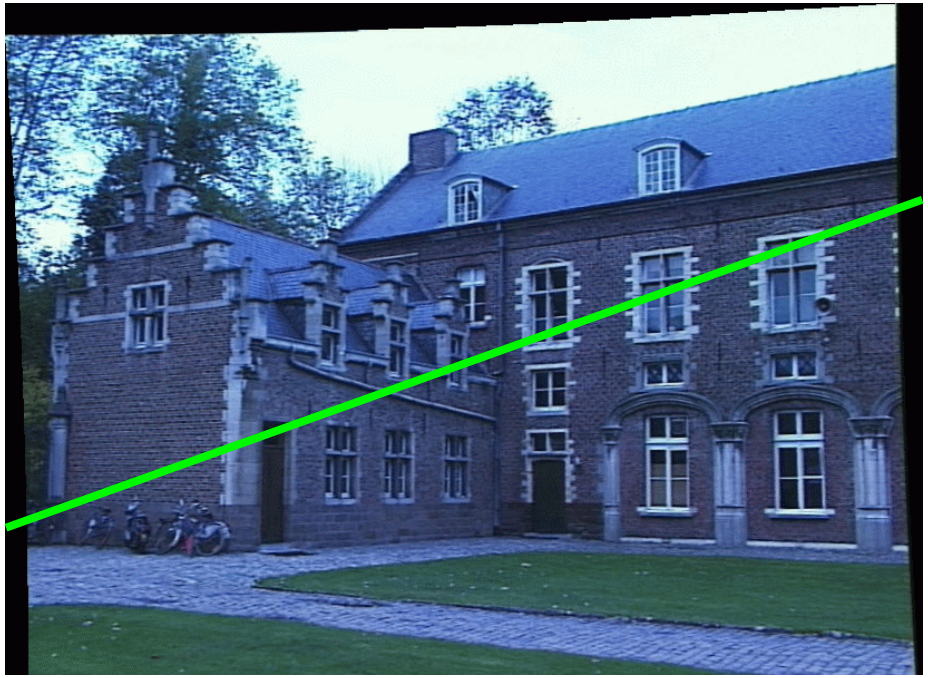
Right image

1. Select point in one image

How would you reconstruct 3D points?



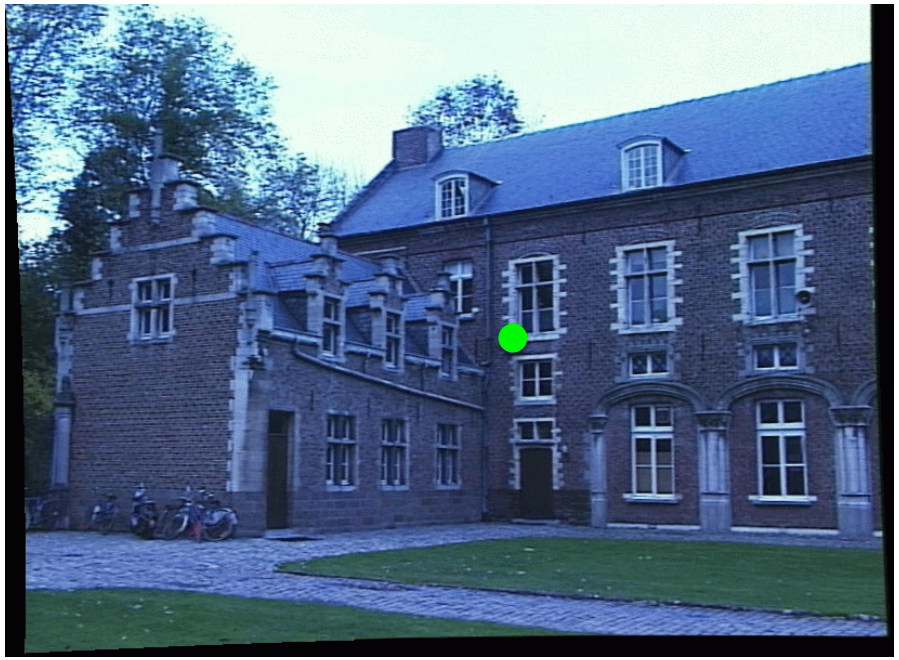
Left image



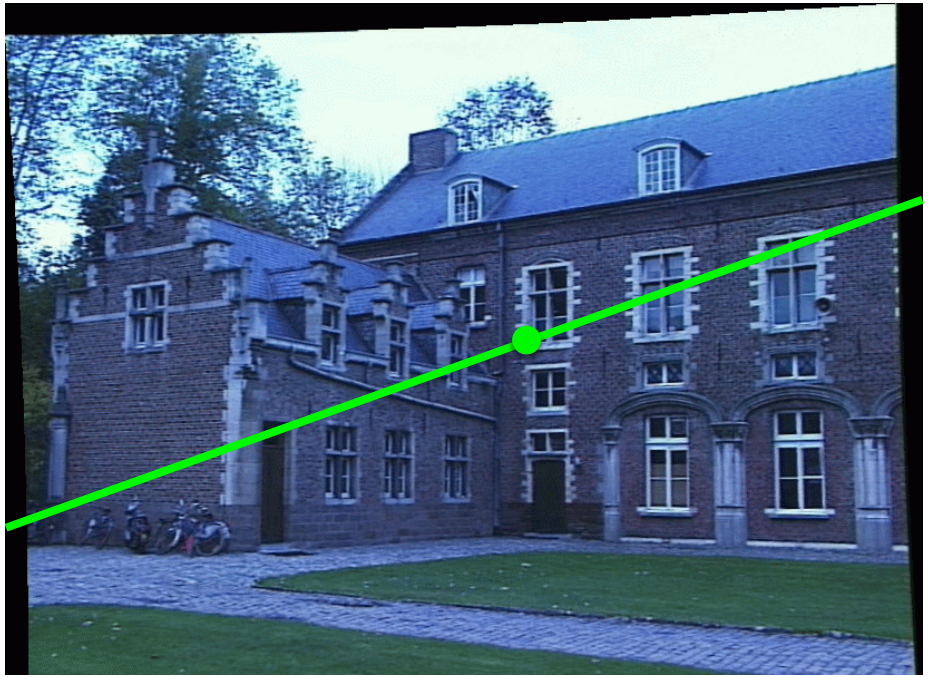
Right image

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

How would you reconstruct 3D points?



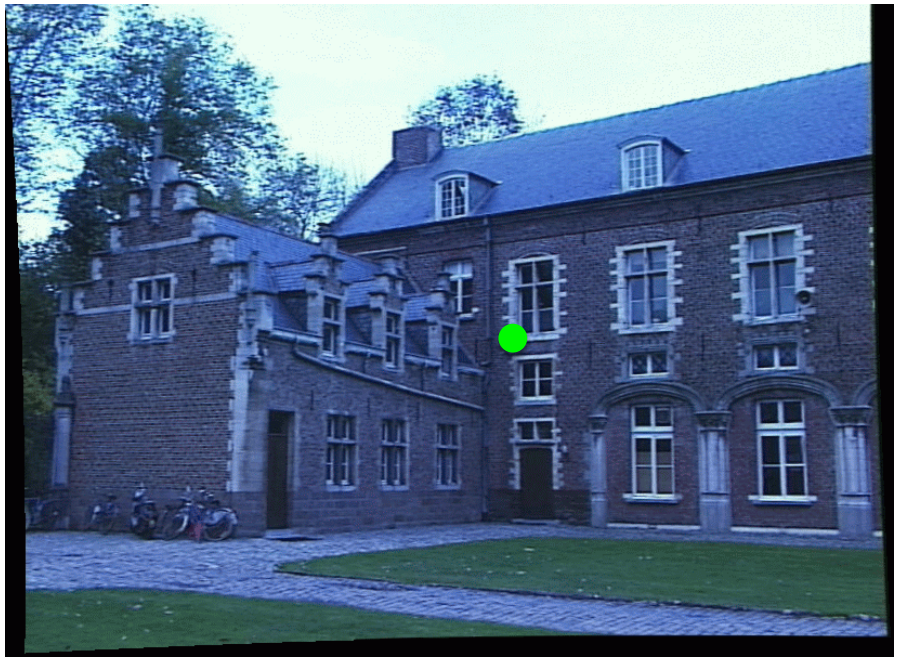
Left image



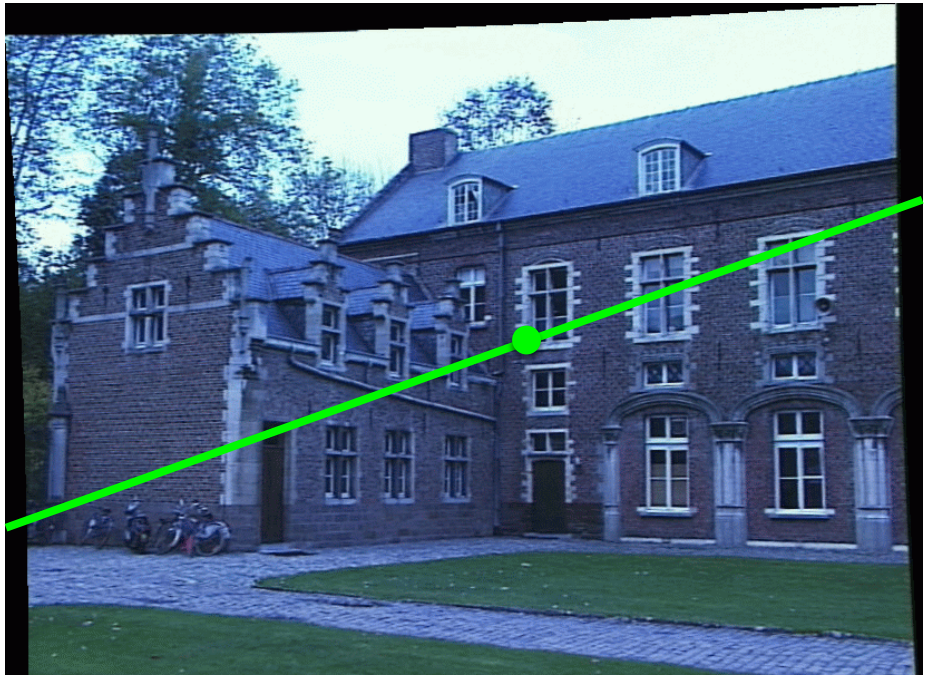
Right image

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

How would you reconstruct 3D points?



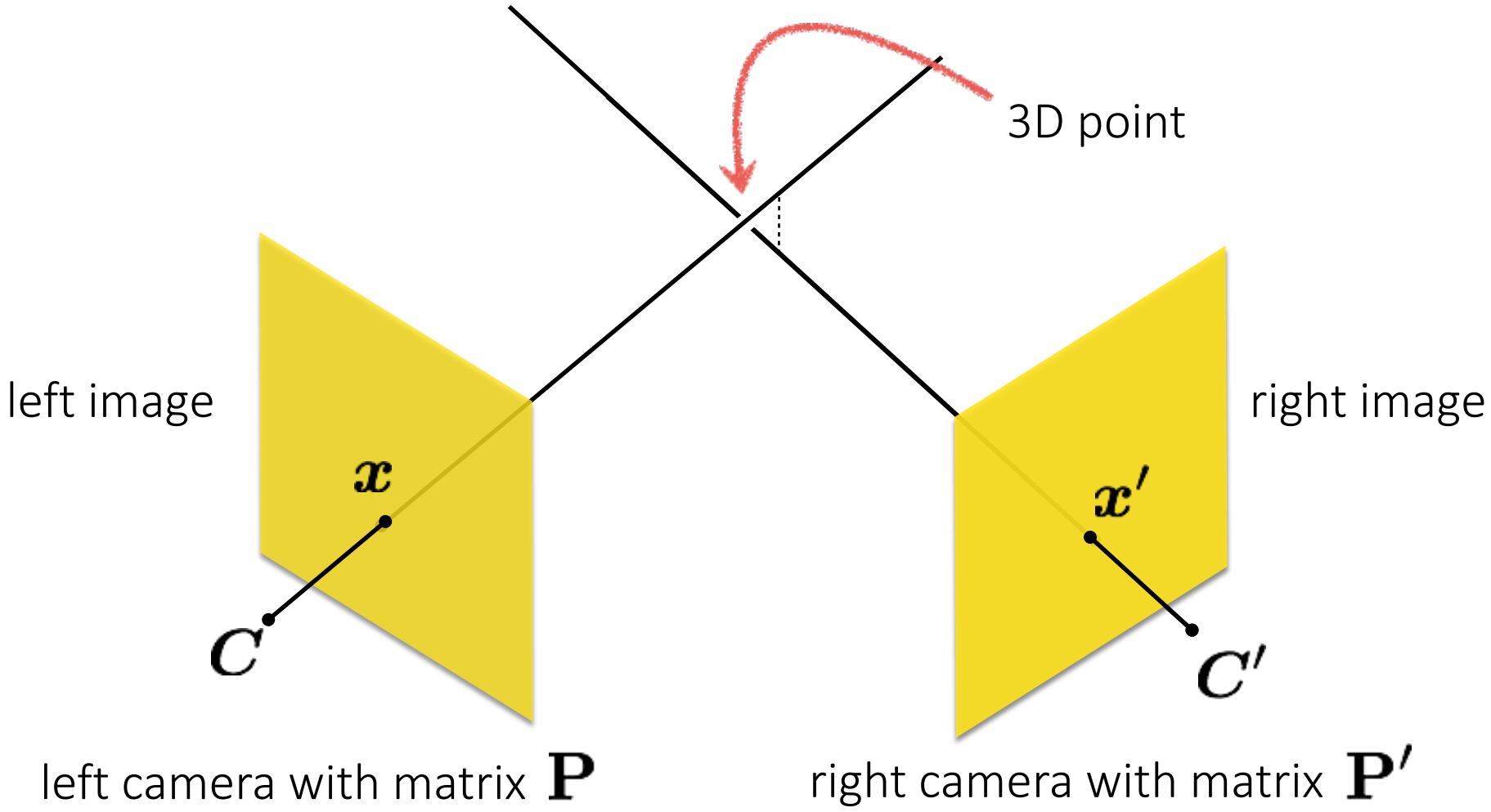
Left image



Right image

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

Triangulation



Stereo rectification





What's different between these two images?





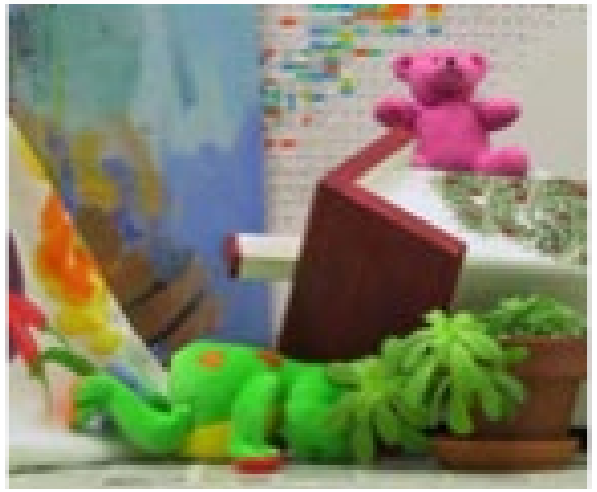


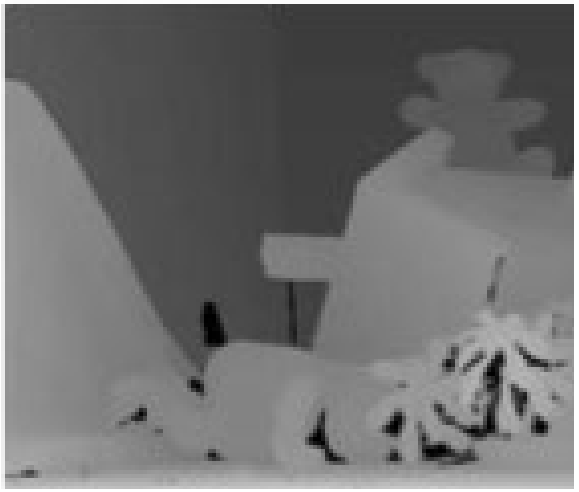


Objects that are close move more or less?

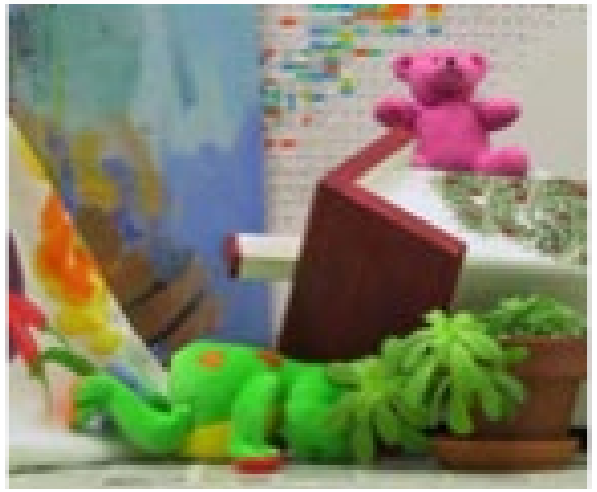
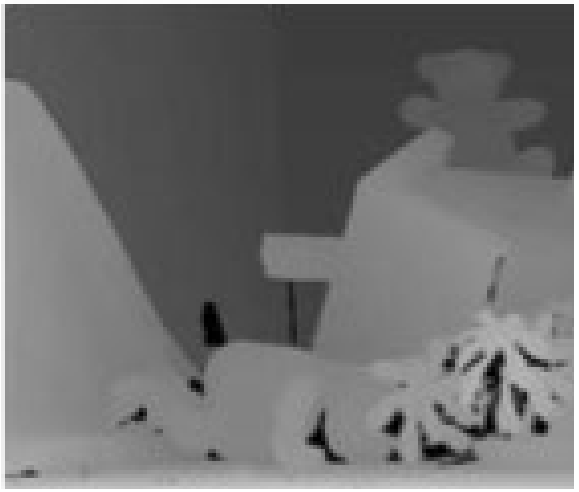
The amount of horizontal movement is inversely proportional to …





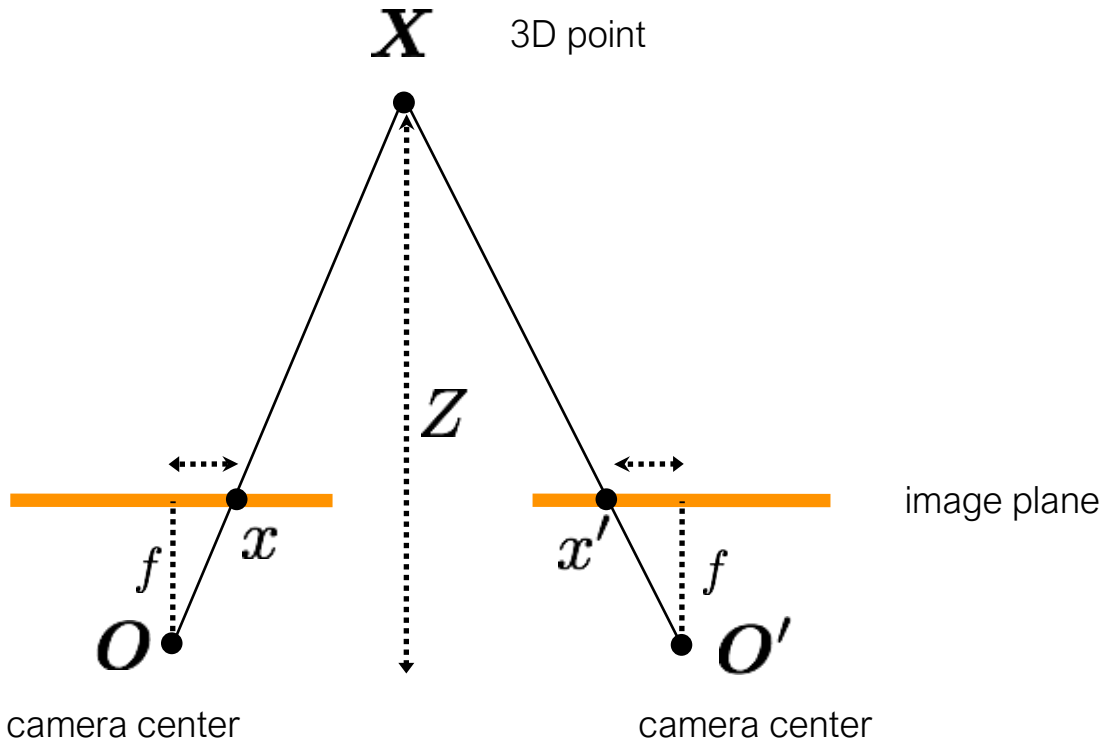


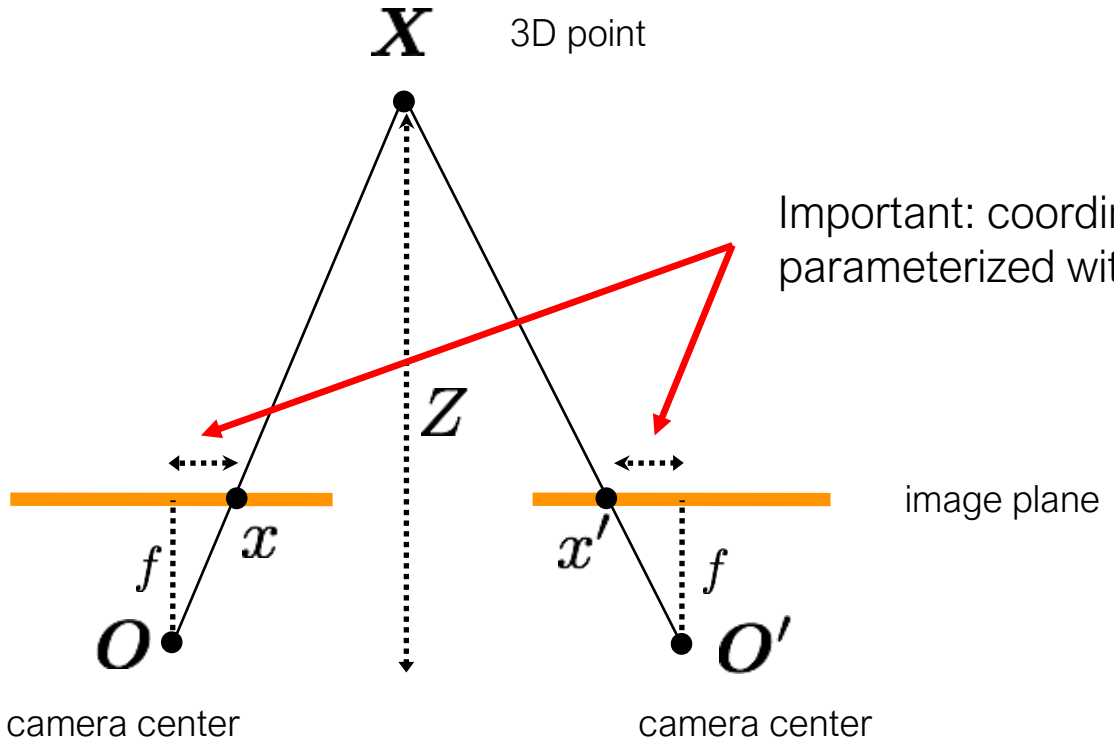
The amount of horizontal movement is inversely proportional to …

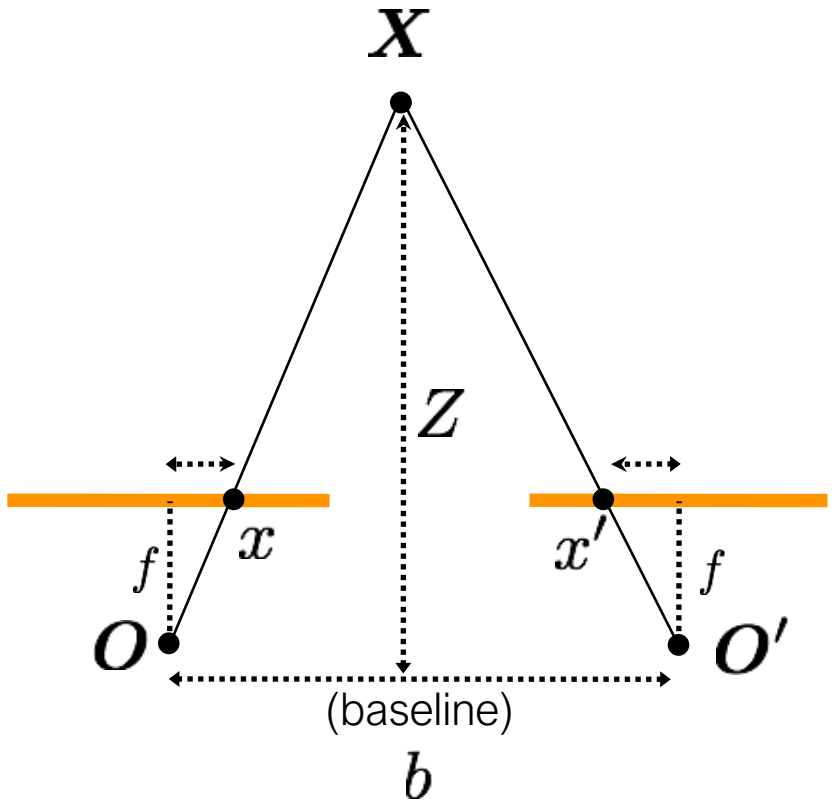
… the distance from the camera.

More formally…

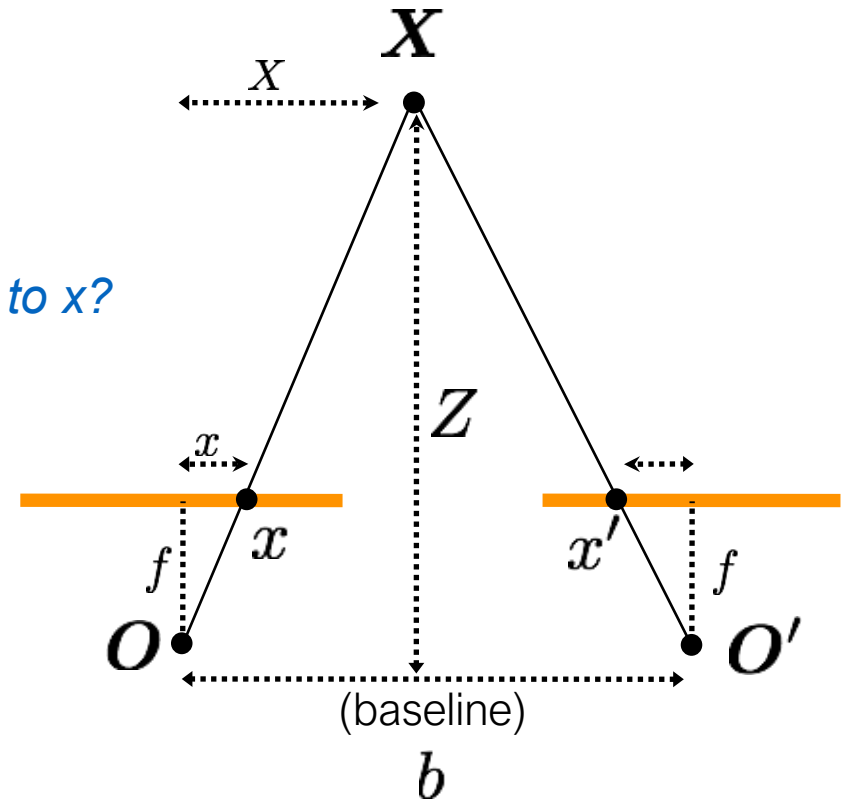

X
3D point
Z
image plane
f
x
x′
f
O
O′
camera center
camera center



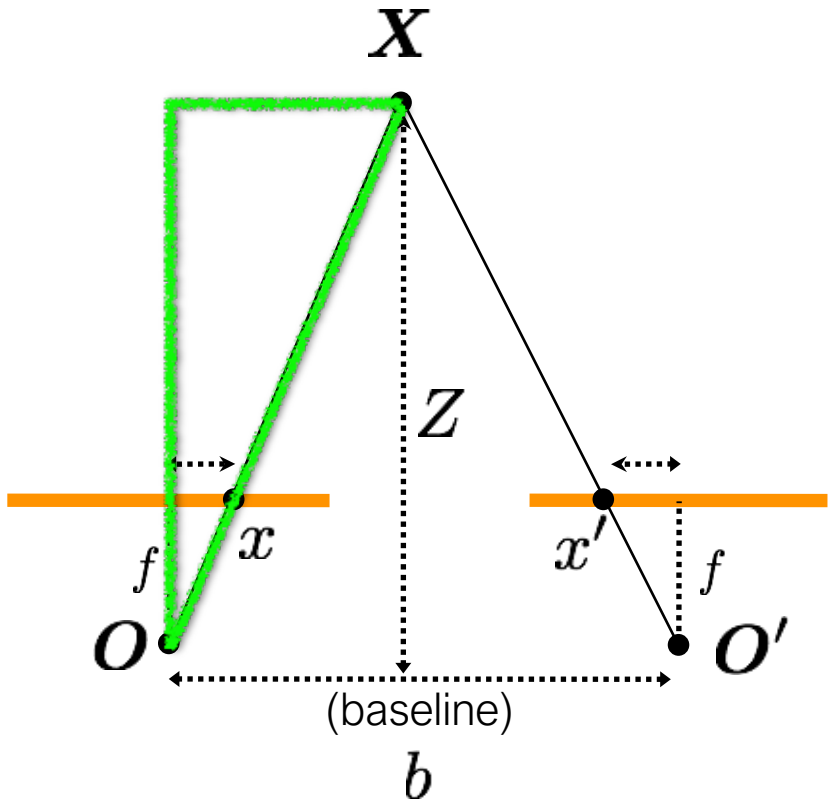
Important: coordinates **x** and **x'** are parameterized with respect to image center.


X
Z
x
x′
f
f
O
O′
(baseline)
b

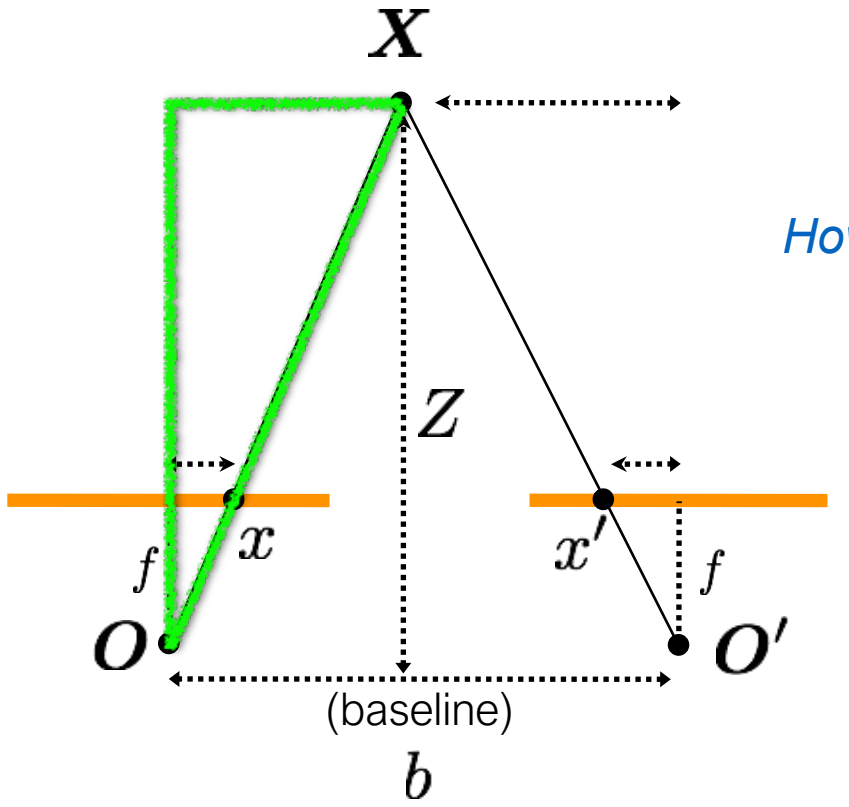
How is X related to x?



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

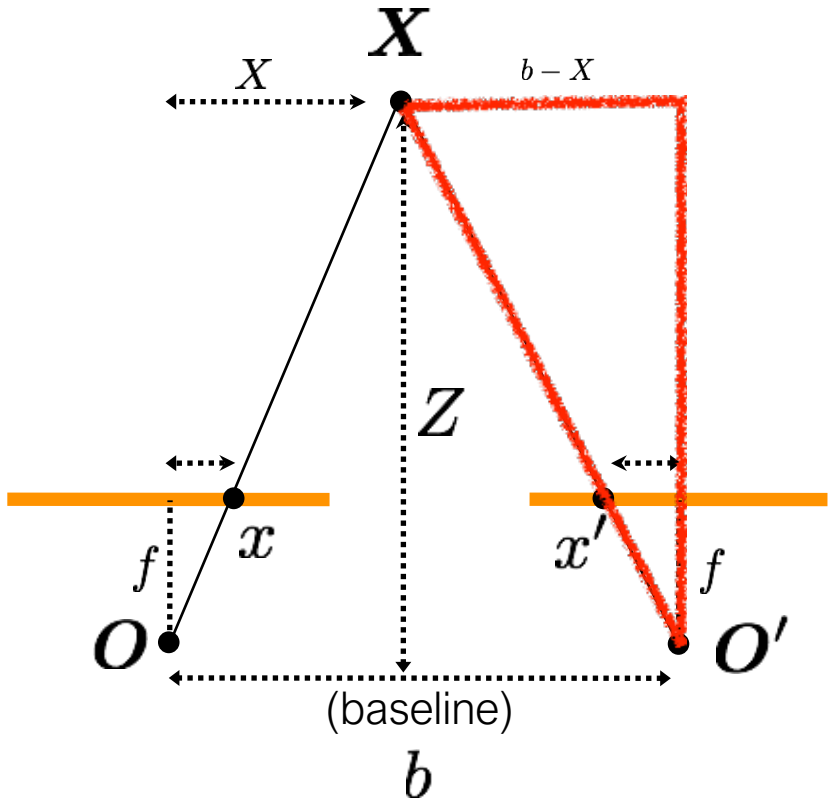


$$\frac{X}{Z} = \frac{x}{f}$$



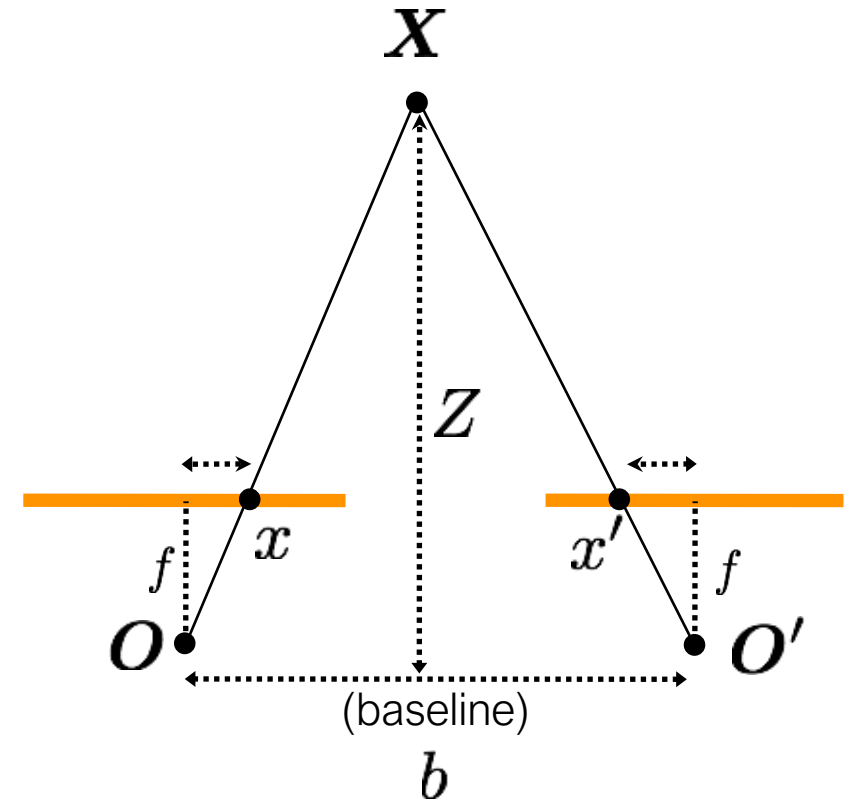
How is X related to x'?

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



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

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



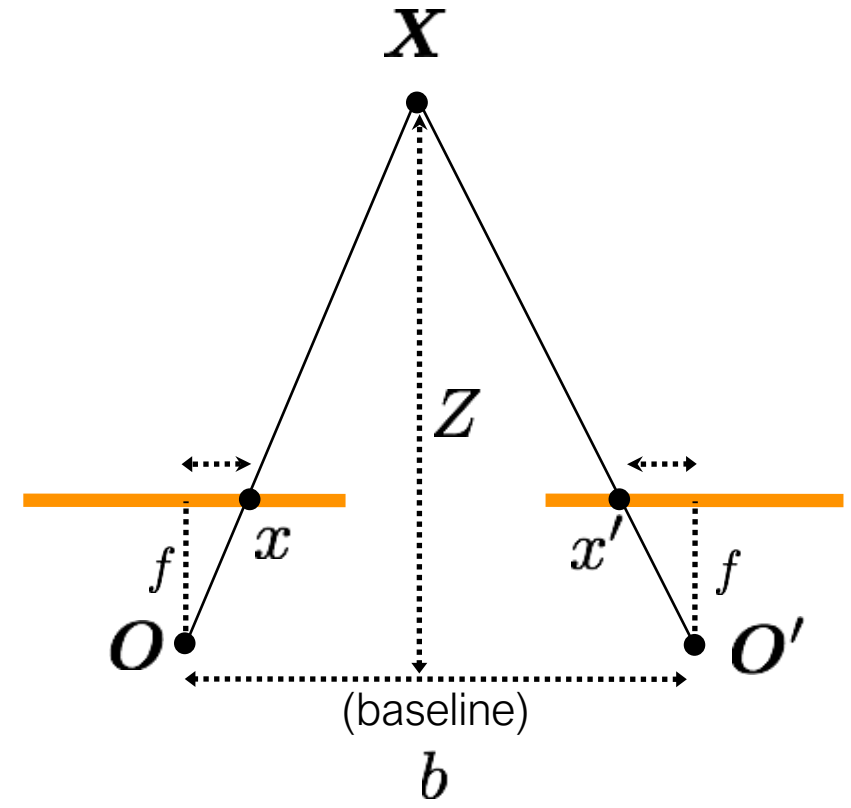
$$\frac{b - X}{Z} = \frac{-x'}{f}$$

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

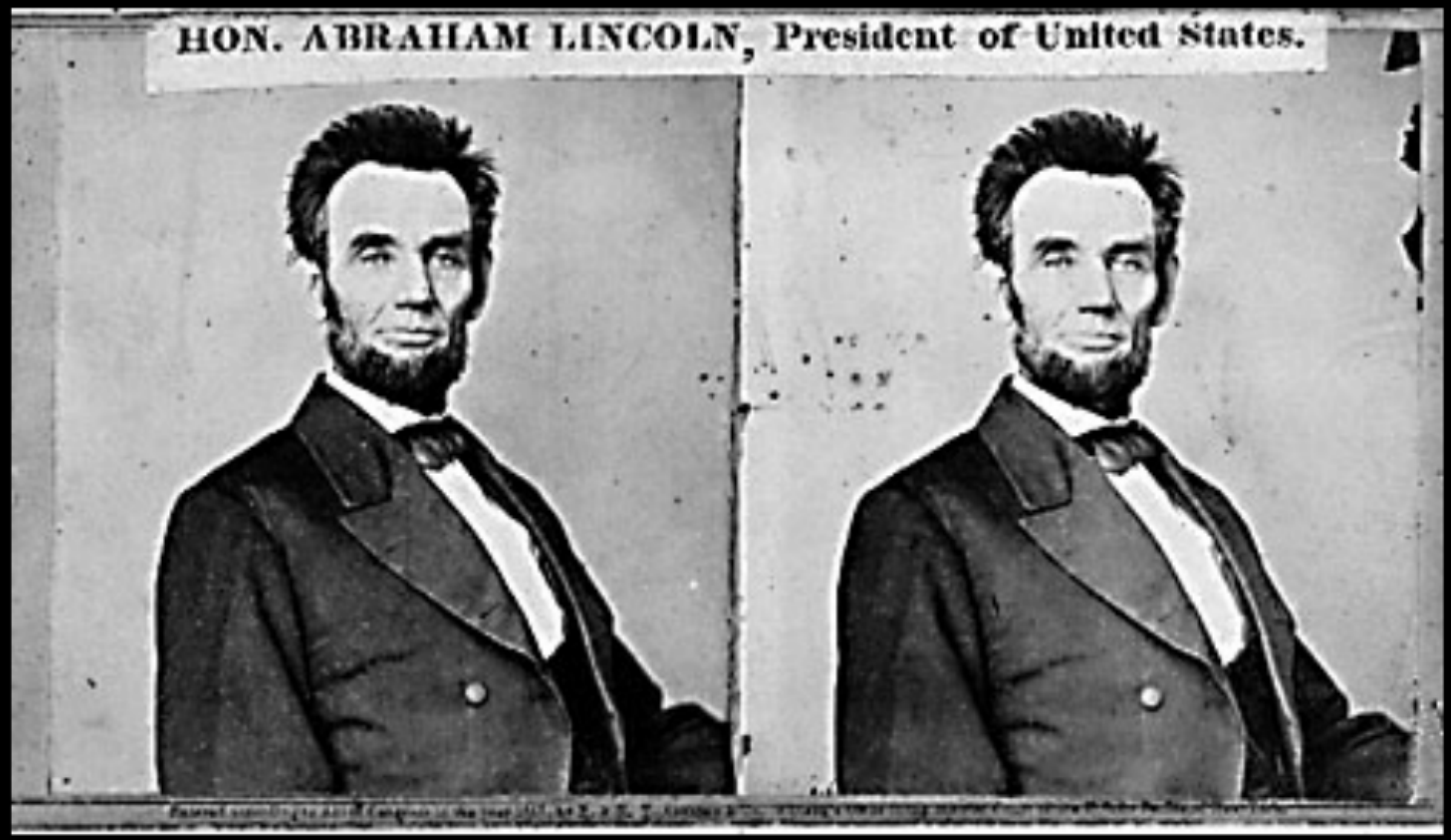


What other vision system uses disparity for depth sensing?

Stereoscopes: A 19th Century Pastime






HON. ABRAHAM LINCOLN, President of United States.



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





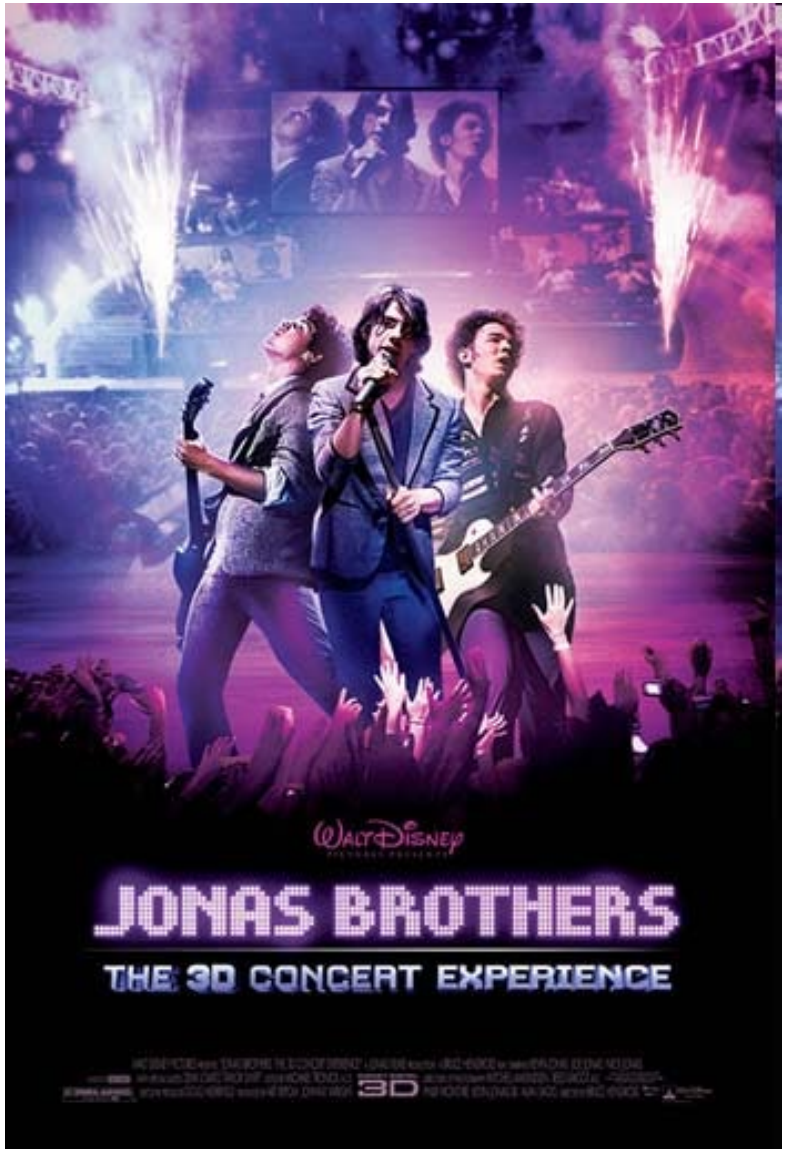
Teesta suspension bridge-Darjeeling, India





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

This is how 3D movies work

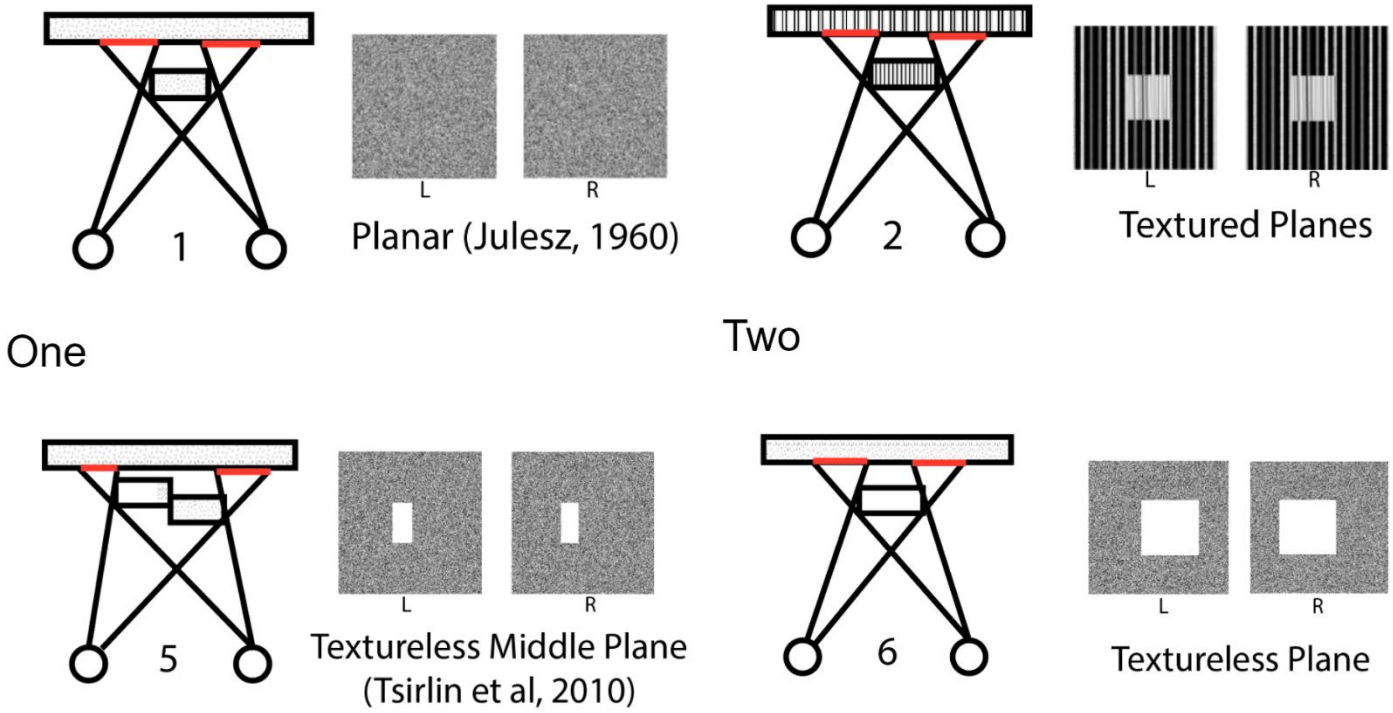




Simple stereoscope



Google cardboard

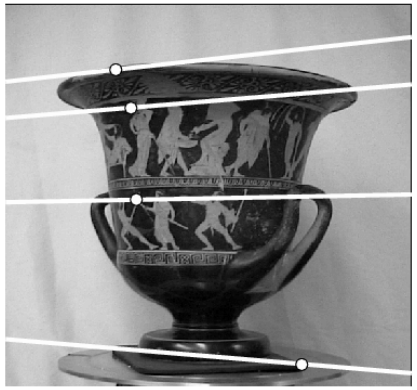


Fun patterns: random dot stereograms

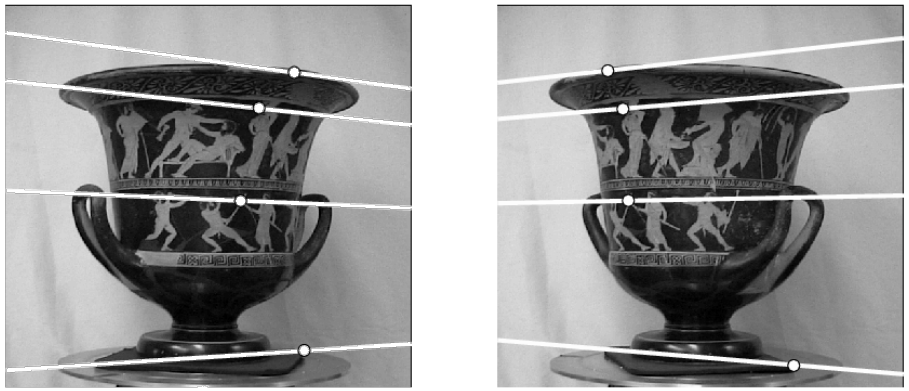
http://vision.seas.harvard.edu/stereo/

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



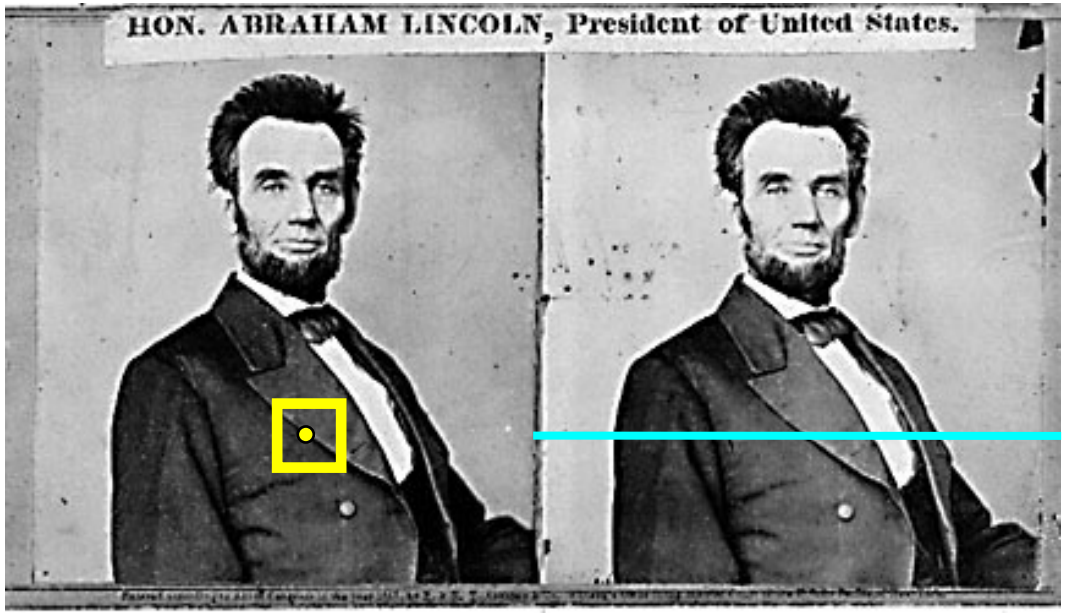


So can I compute depth using disparity from any two images of the same object?

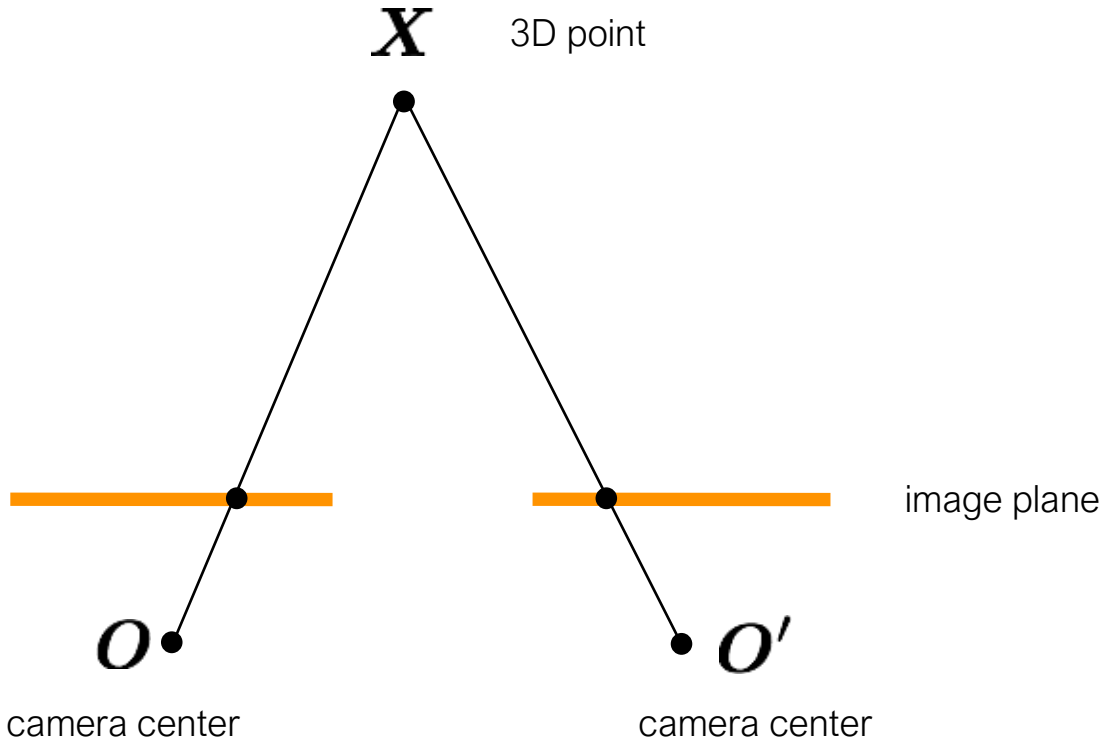


1. Need sufficient baseline

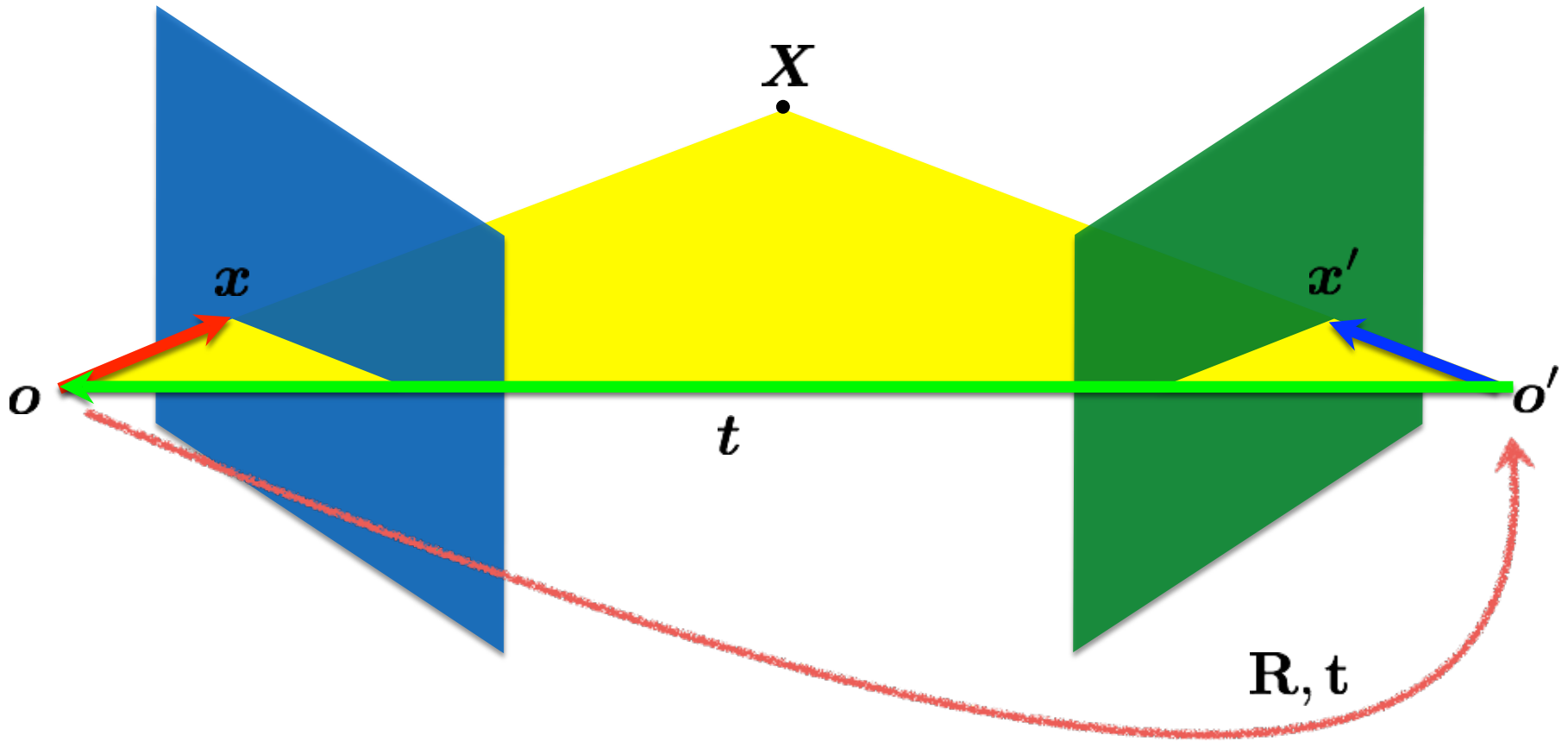
2. Images need to be ‘rectified’ first (make epipolar lines horizontal)



How can you make the epipolar lines horizontal?

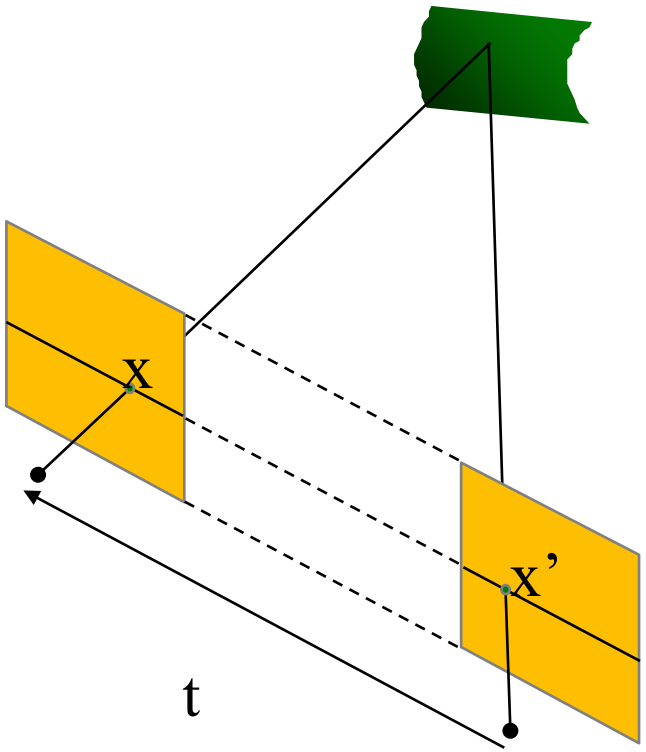


What's special about these two cameras?

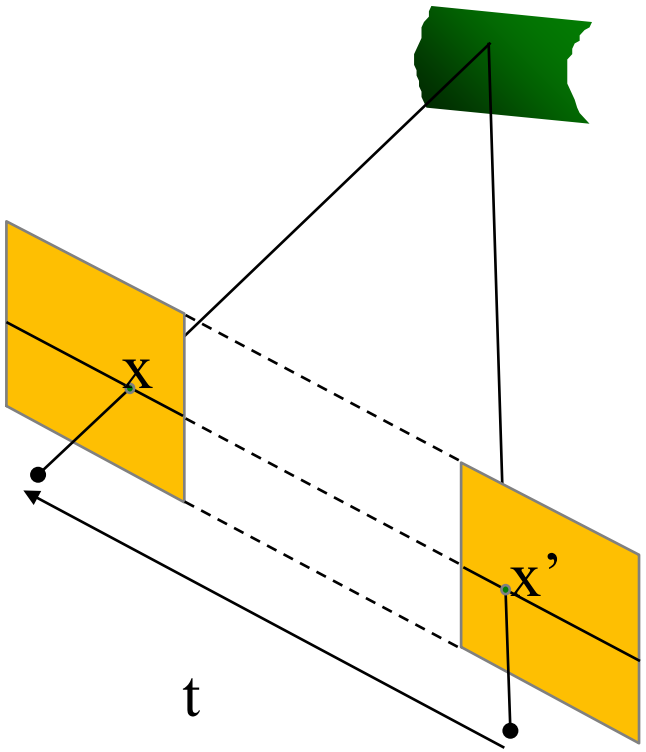


$$x' = \mathbf{R}(x - t)$$

When are epipolar lines horizontal?



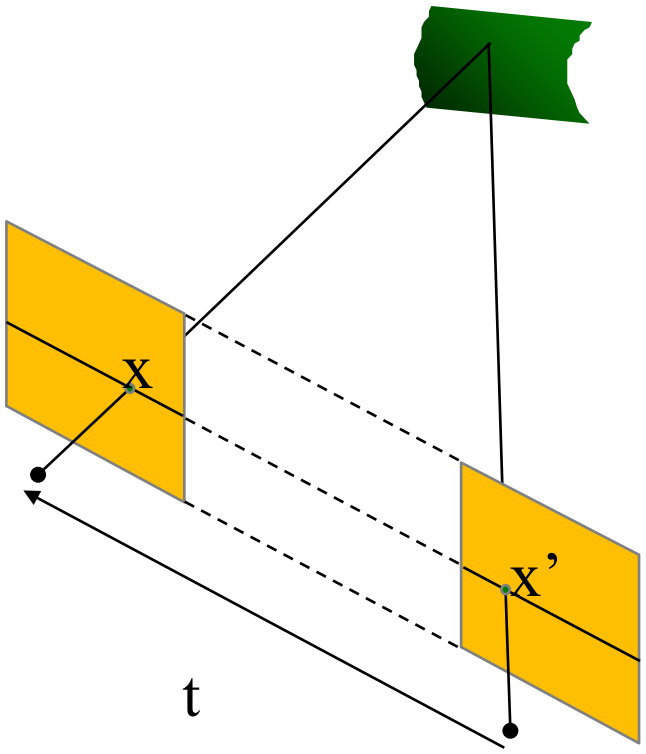
When are epipolar lines horizontal?



When this relationship holds:

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

When are epipolar lines horizontal?



When this relationship holds:

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

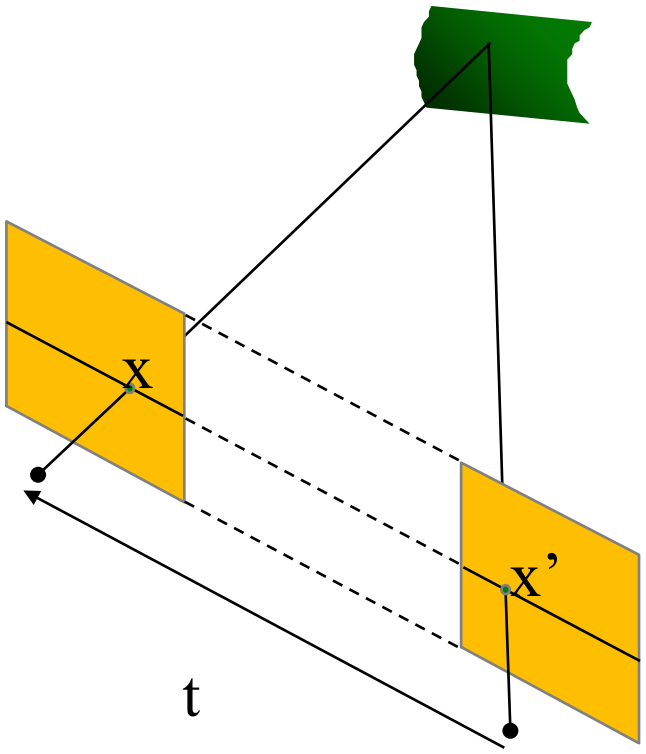
Let's try this out…

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

This always has to hold:

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

When are epipolar lines horizontal?



When this relationship holds:

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

Let's try this out…

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

This always has to hold:

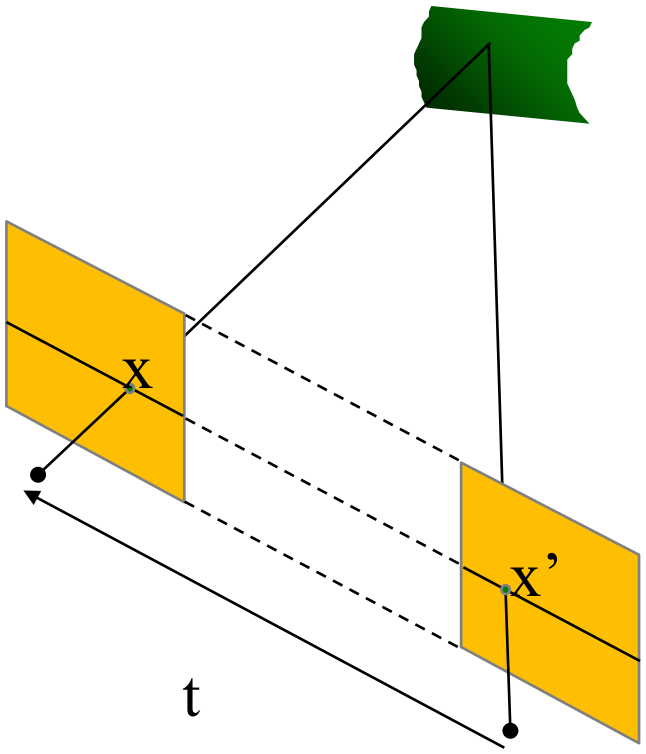
$$x^T E x' = 0$$

Write out the constraint

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

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

When are epipolar lines horizontal?



When this relationship holds:

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

Let's try this out…

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

This always has to hold:

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

Write out the constraint

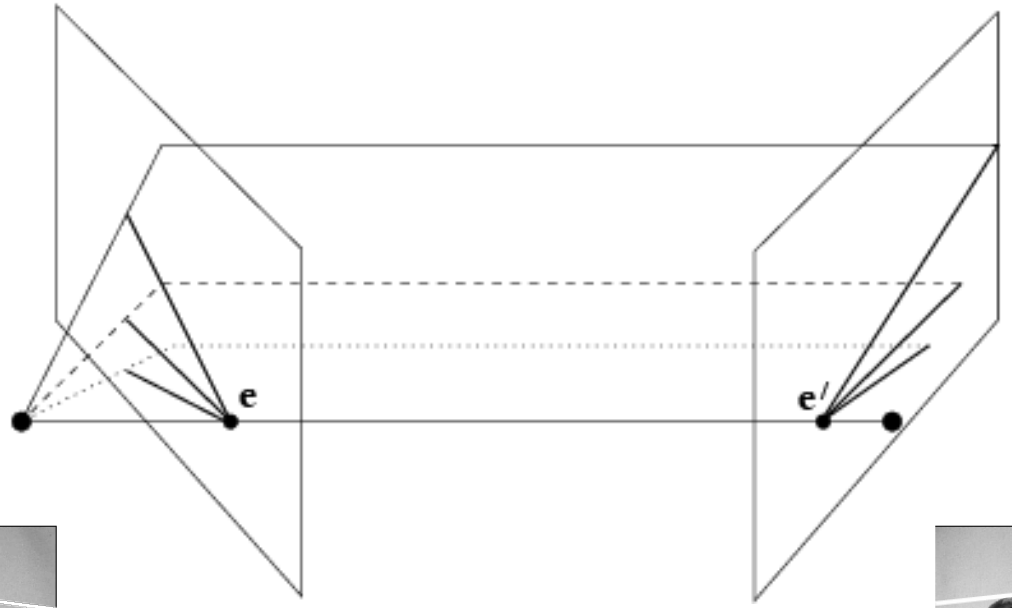
$$\begin{pmatrix} u & v & 1 \end{pmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T \\ 0 & T & 0 \end{bmatrix} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0$$

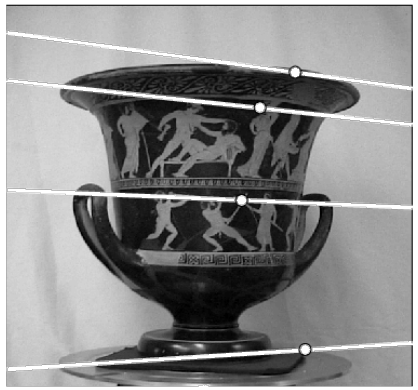
$$\begin{pmatrix} u & v & 1 \end{pmatrix} \begin{pmatrix} 0 \\ -T \\ Tv' \end{pmatrix} = 0$$

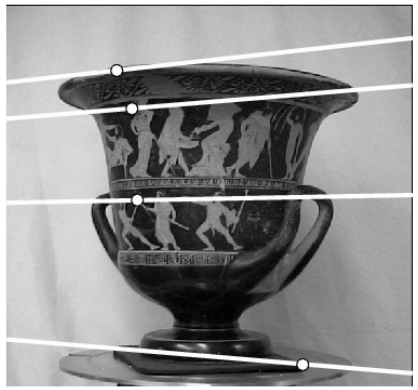
The image of a 3D point will always be on the same horizontal line

$$Tv = Tv'$$

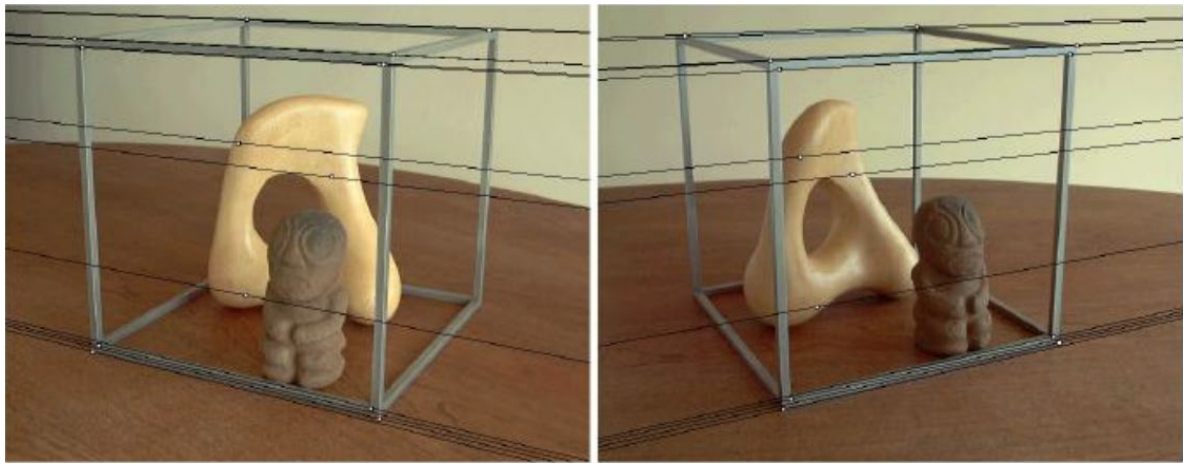
y coordinate is always the same!





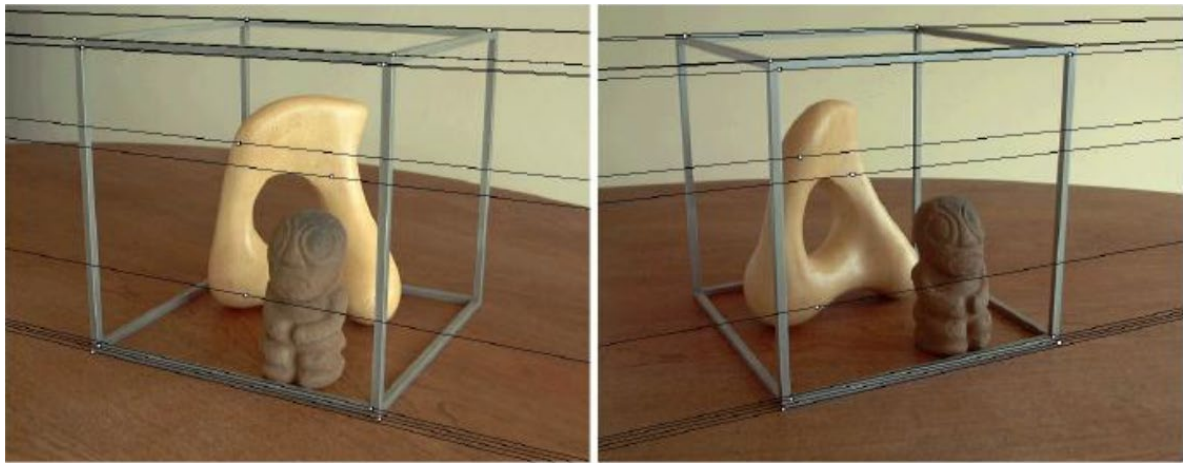


It’s hard to make the image planes exactly parallel

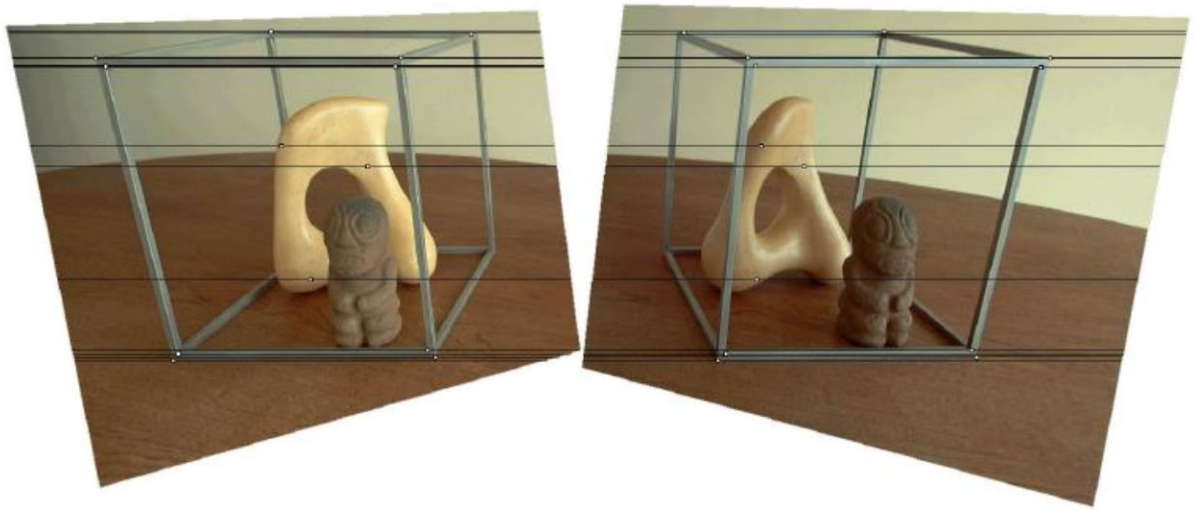


How can you make the epipolar lines horizontal?



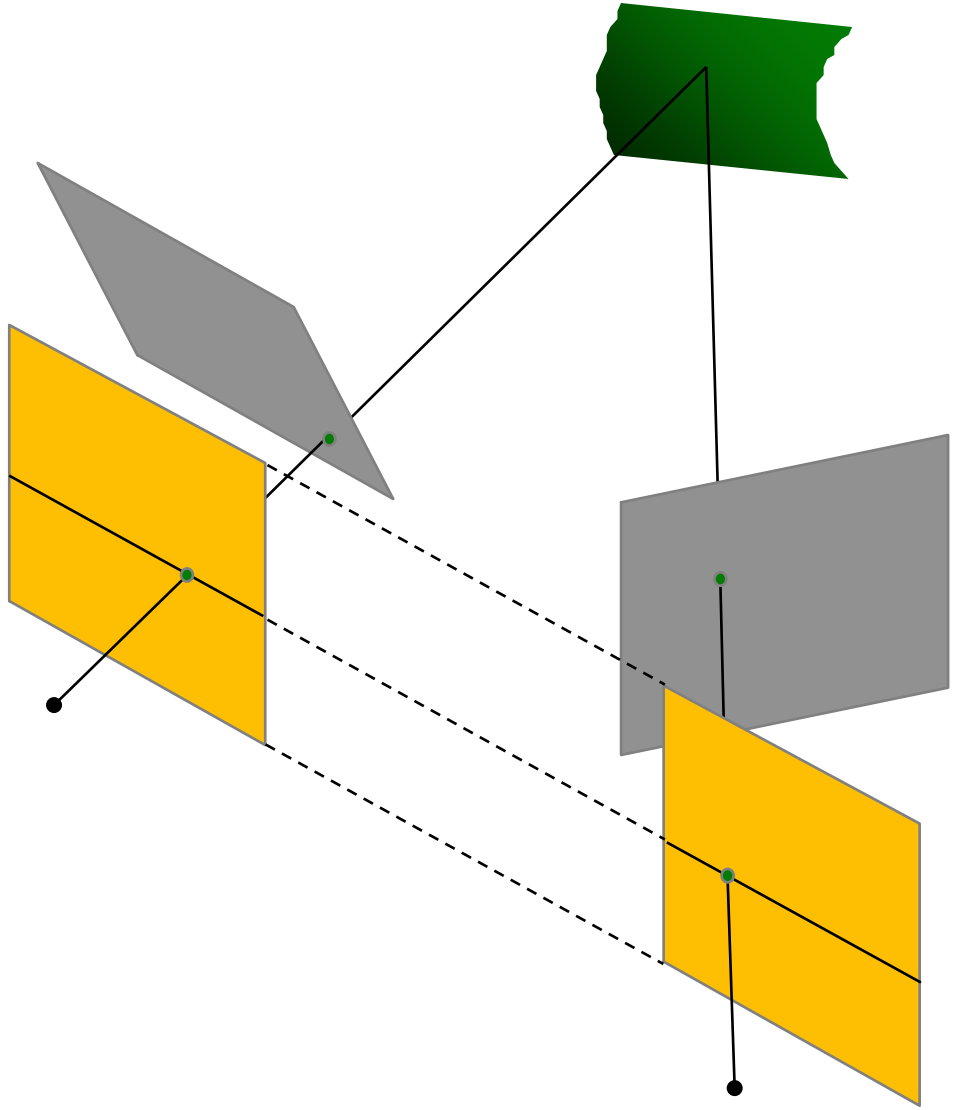


Use stereo rectification



What is stereo rectification?

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

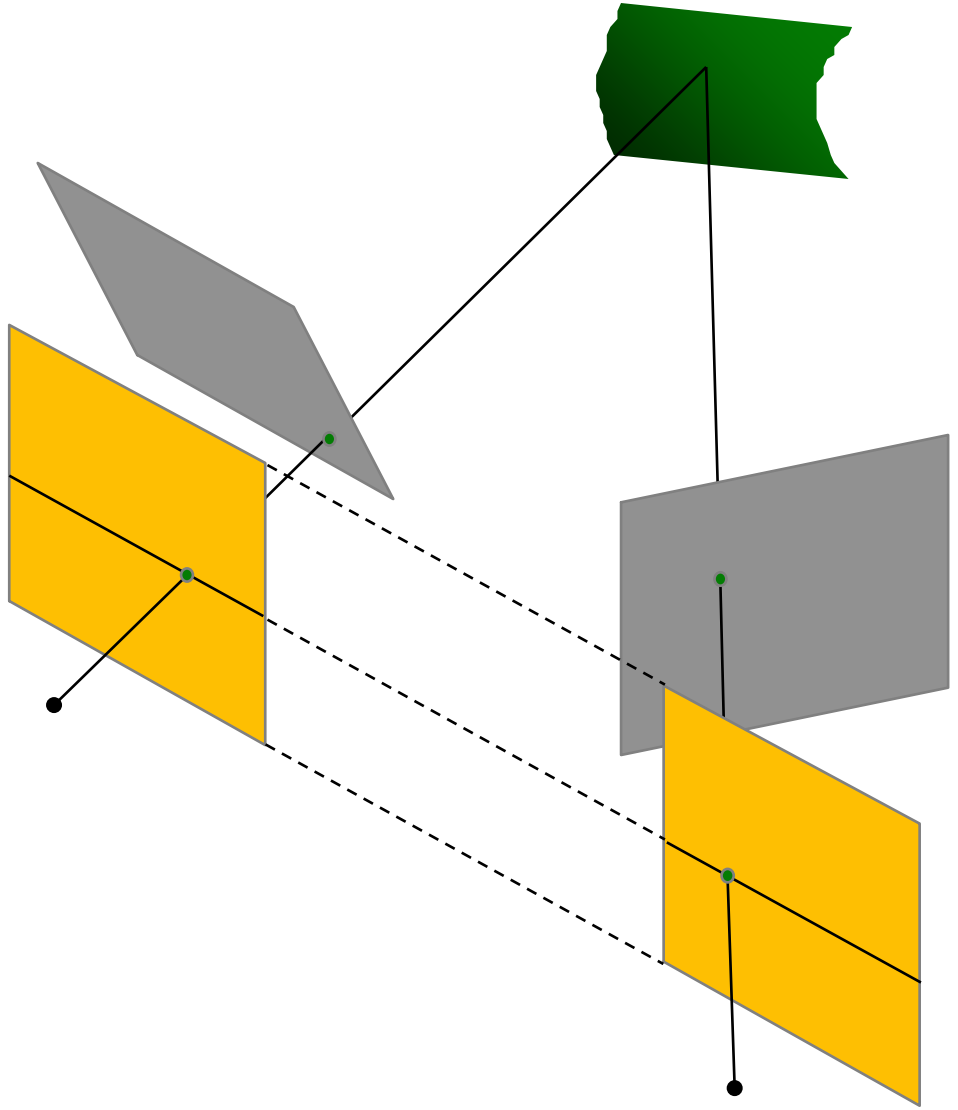


How can you do this?

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.

Stereo Rectification

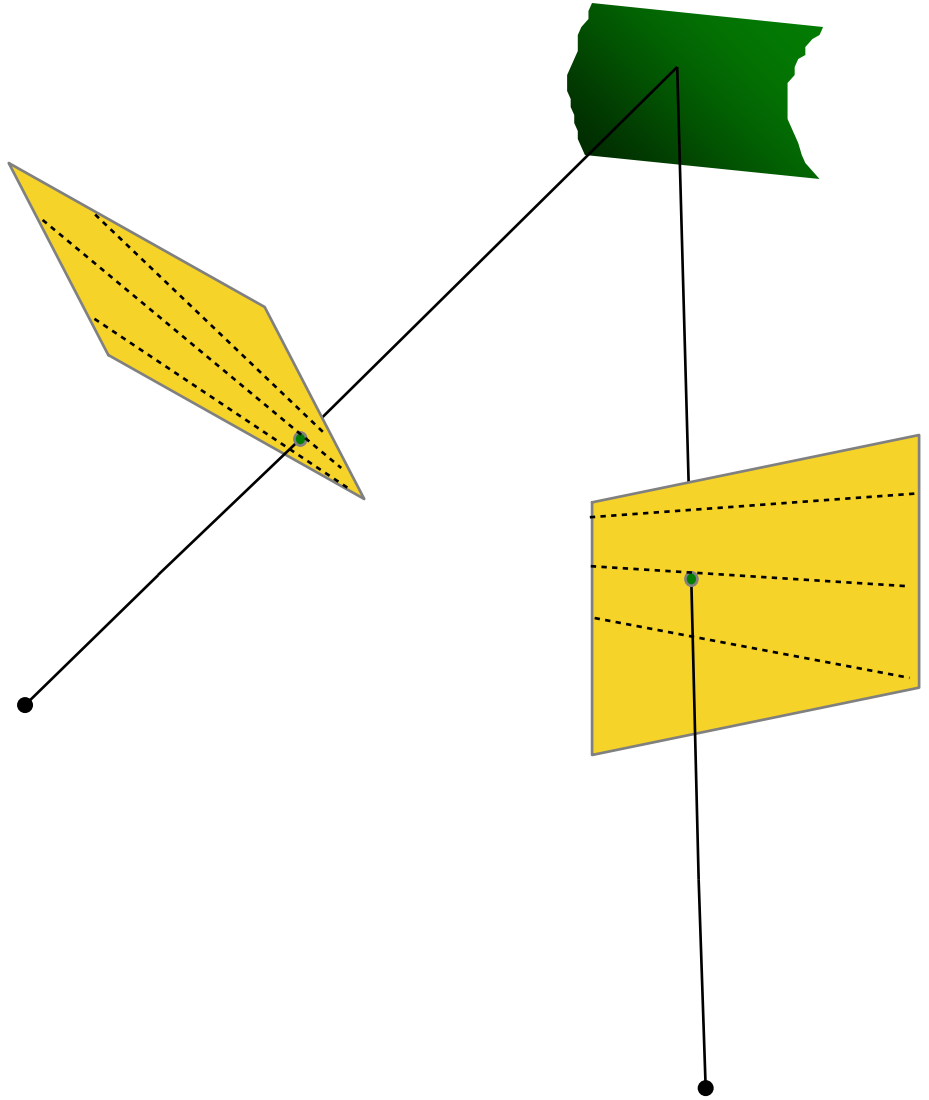
1. **Rotate** the right camera by **R**
   (aligns camera coordinate system orientation only)

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

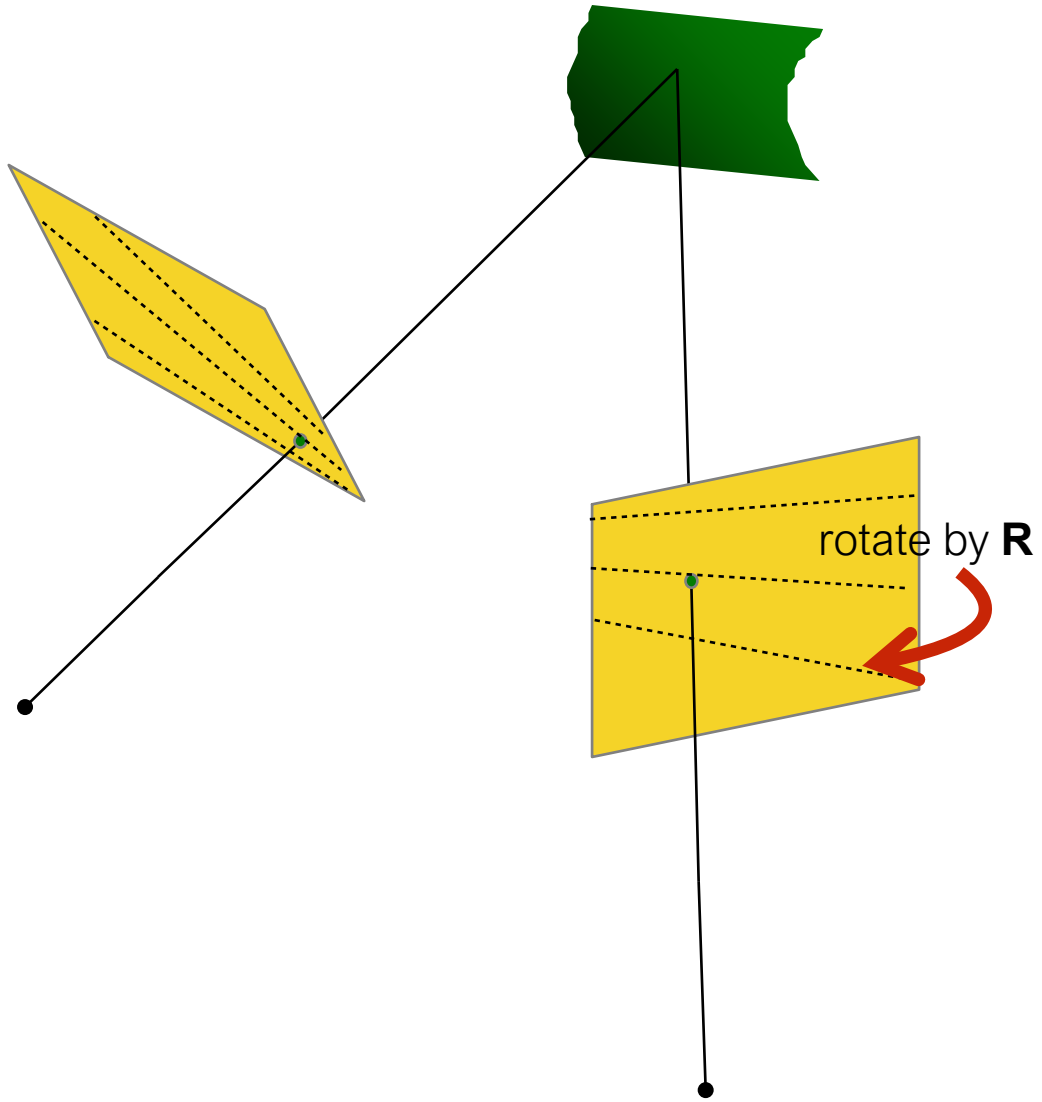
4. Adjust the **scale**

Stereo Rectification:



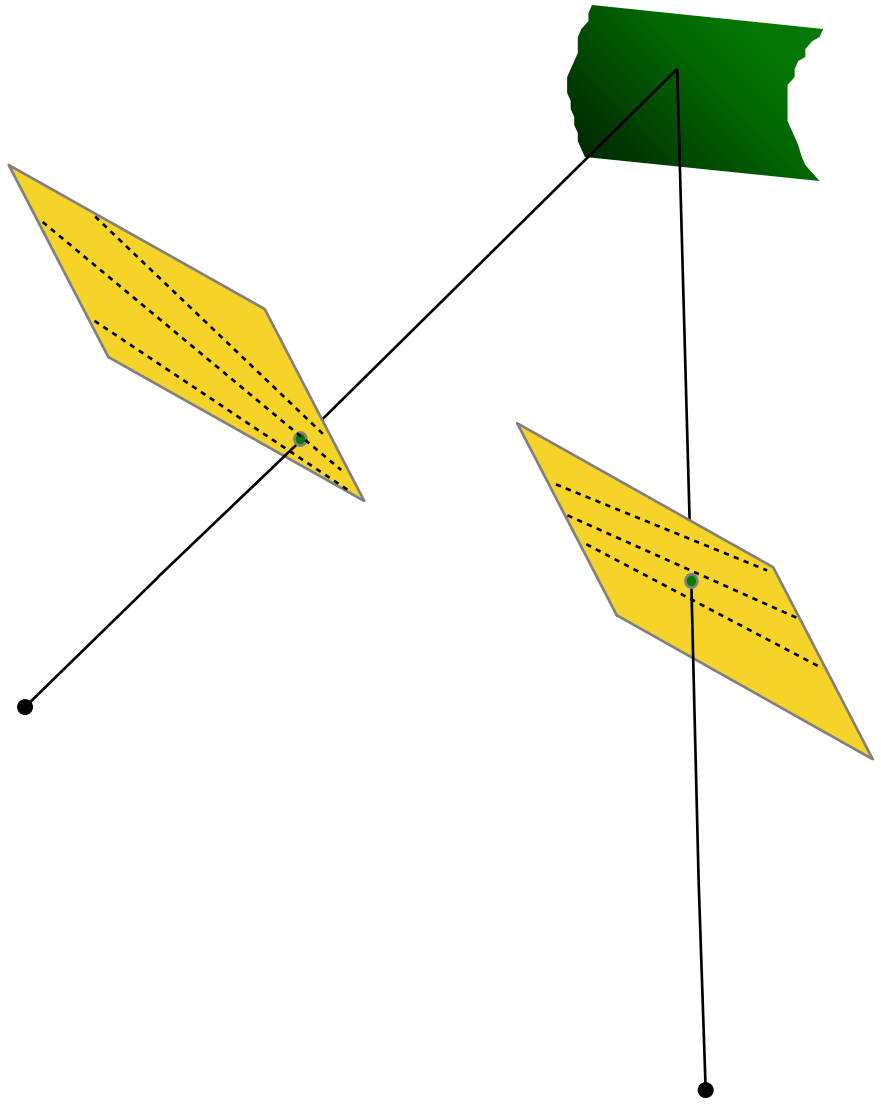
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R}_{\mathbf{rect}}$
4. Scale both images by **H**

Stereo Rectification:



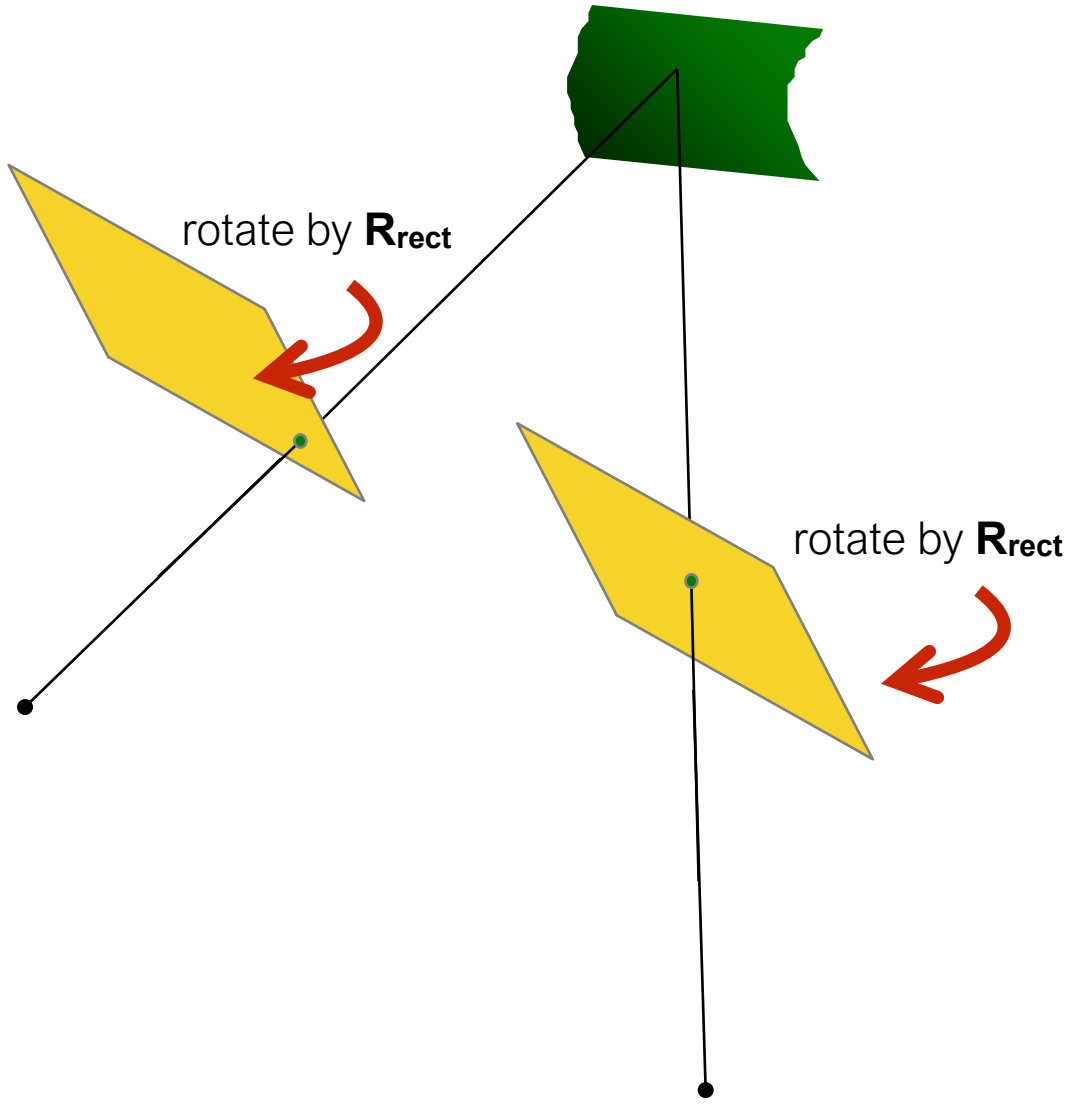
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R}_{\mathbf{rect}}$
4. Scale both images by **H**

Stereo Rectification:



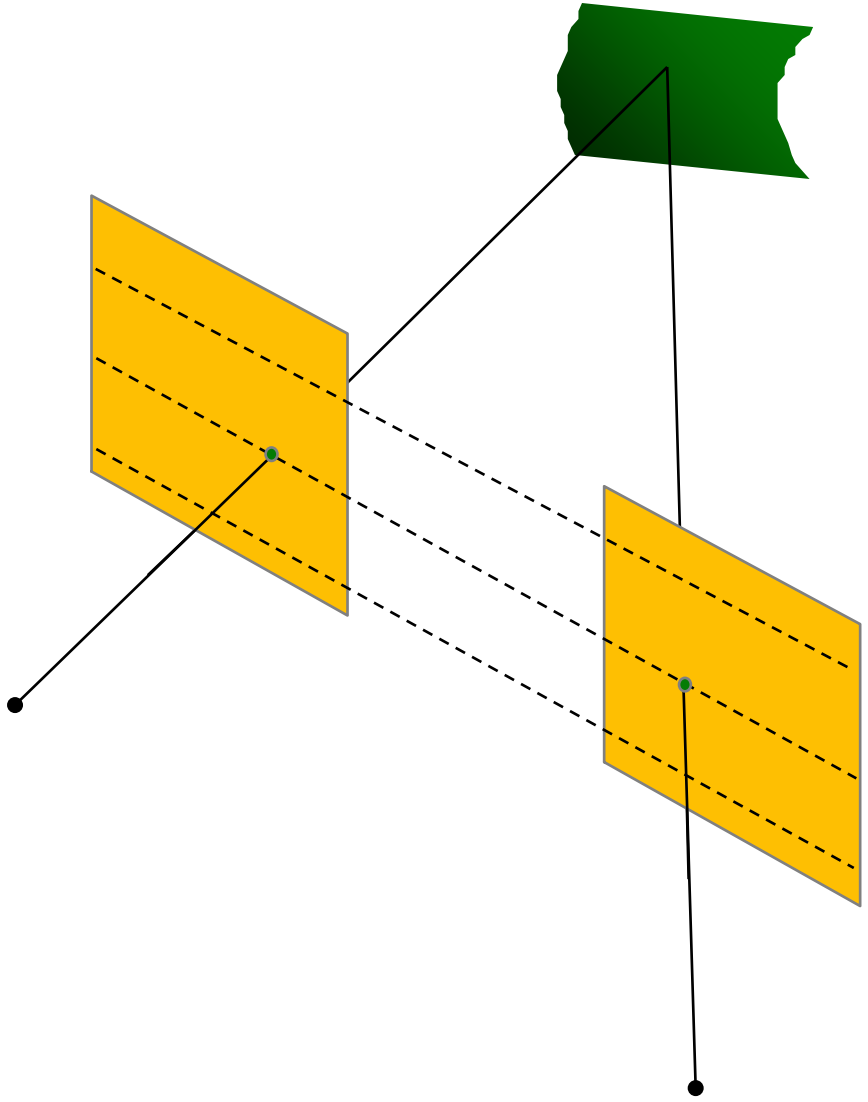
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R}_{\mathbf{rect}}$
4. Scale both images by **H**

Stereo Rectification:



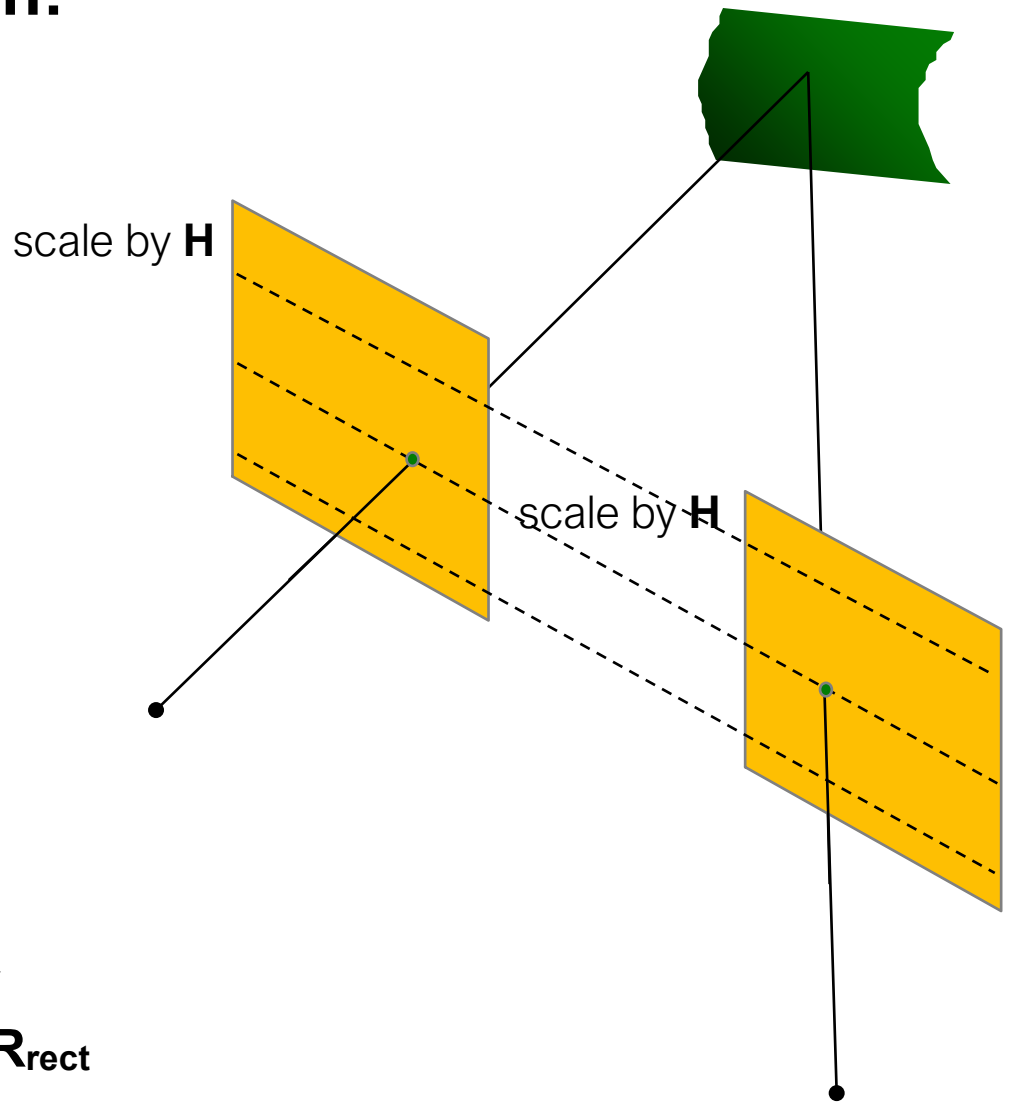
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R}_{\mathbf{rect}}$
4. Scale both images by **H**

Stereo Rectification:



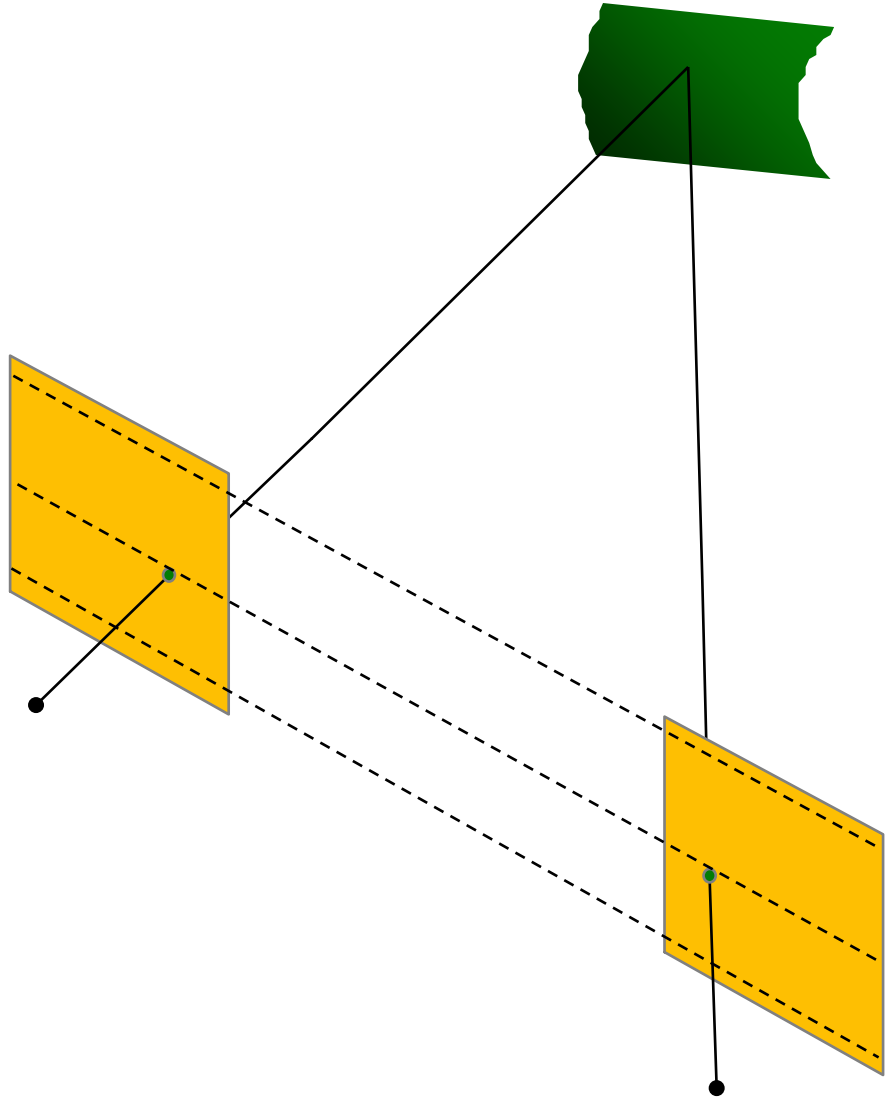
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R_{rect}}$
4. Scale both images by **H**

Stereo Rectification:



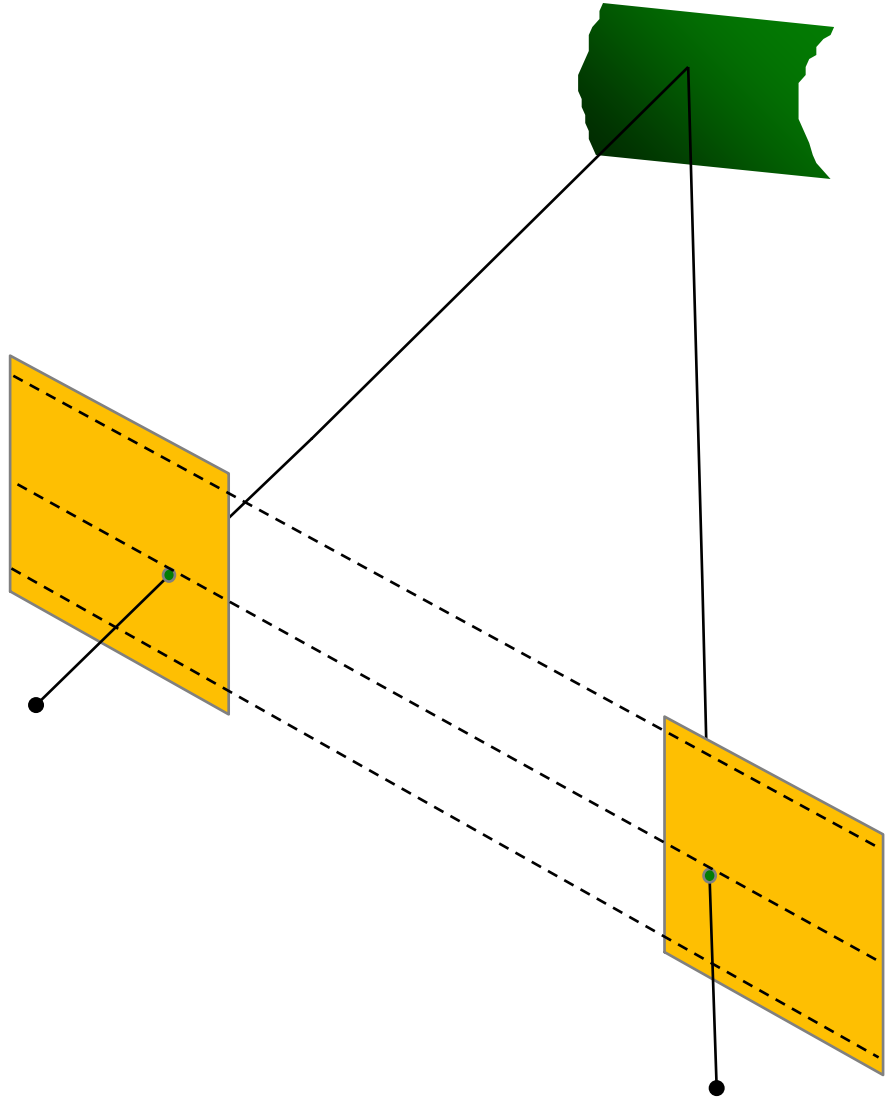
1. Compute **E** to get **R**
2. Rotate right image by **R**
3. Rotate both images by $\mathbf{R}_{\mathbf{rect}}$
4. Scale both images by **H**

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 $\mathbf{R}_{\mathbf{rect}}$
4. Scale both images by **H**

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

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

We get FOUR solutions:

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

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

two possible rotations

two possible translations

We get FOUR solutions:

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

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

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

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

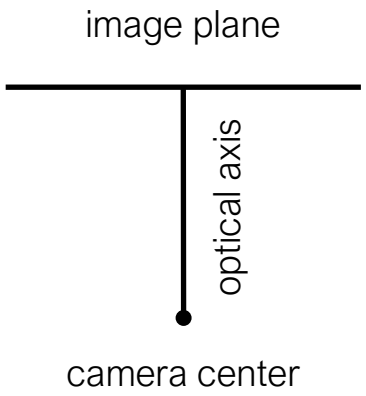
Which one do we choose?

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)

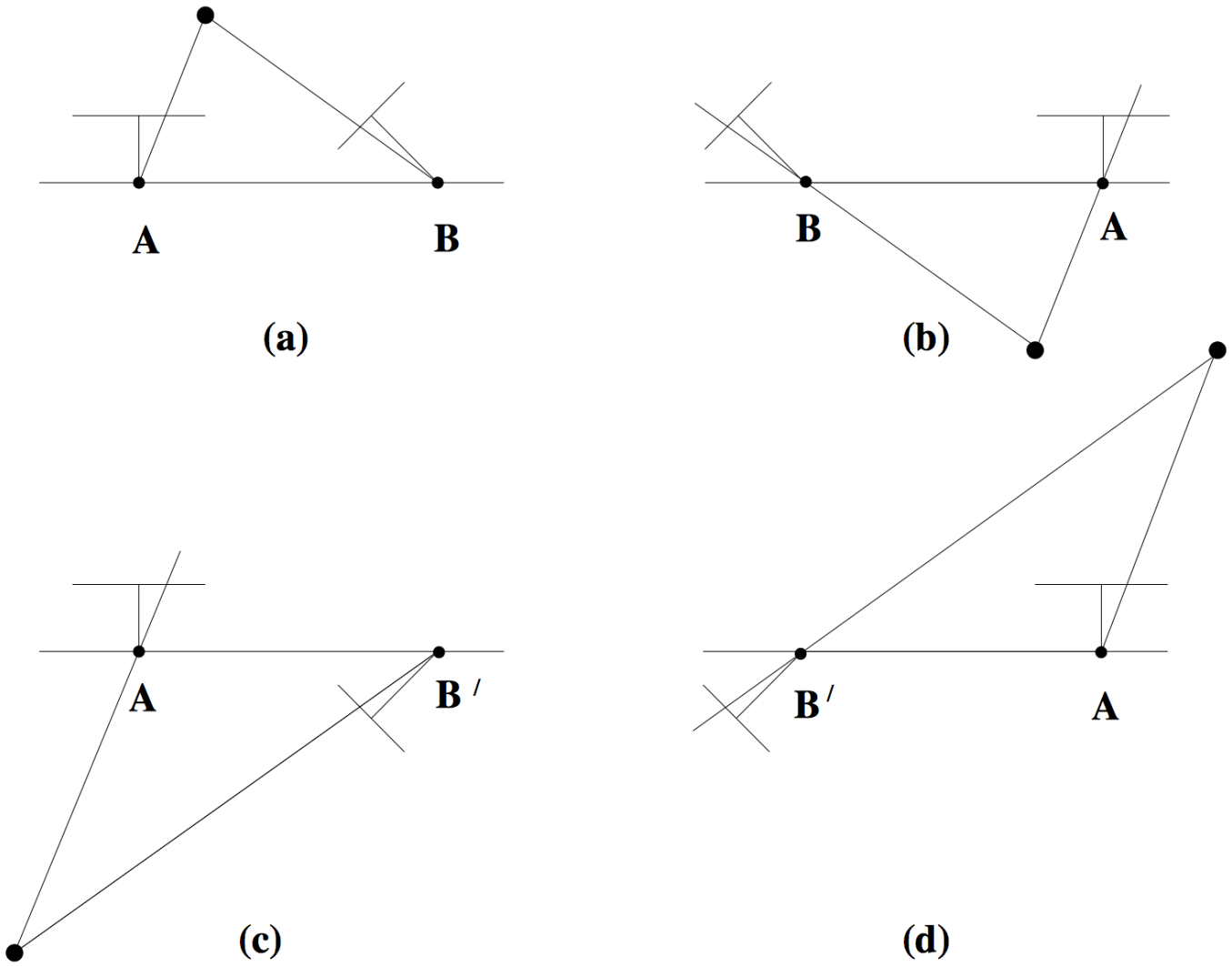
Let’s visualize the four configurations…

Camera Icon

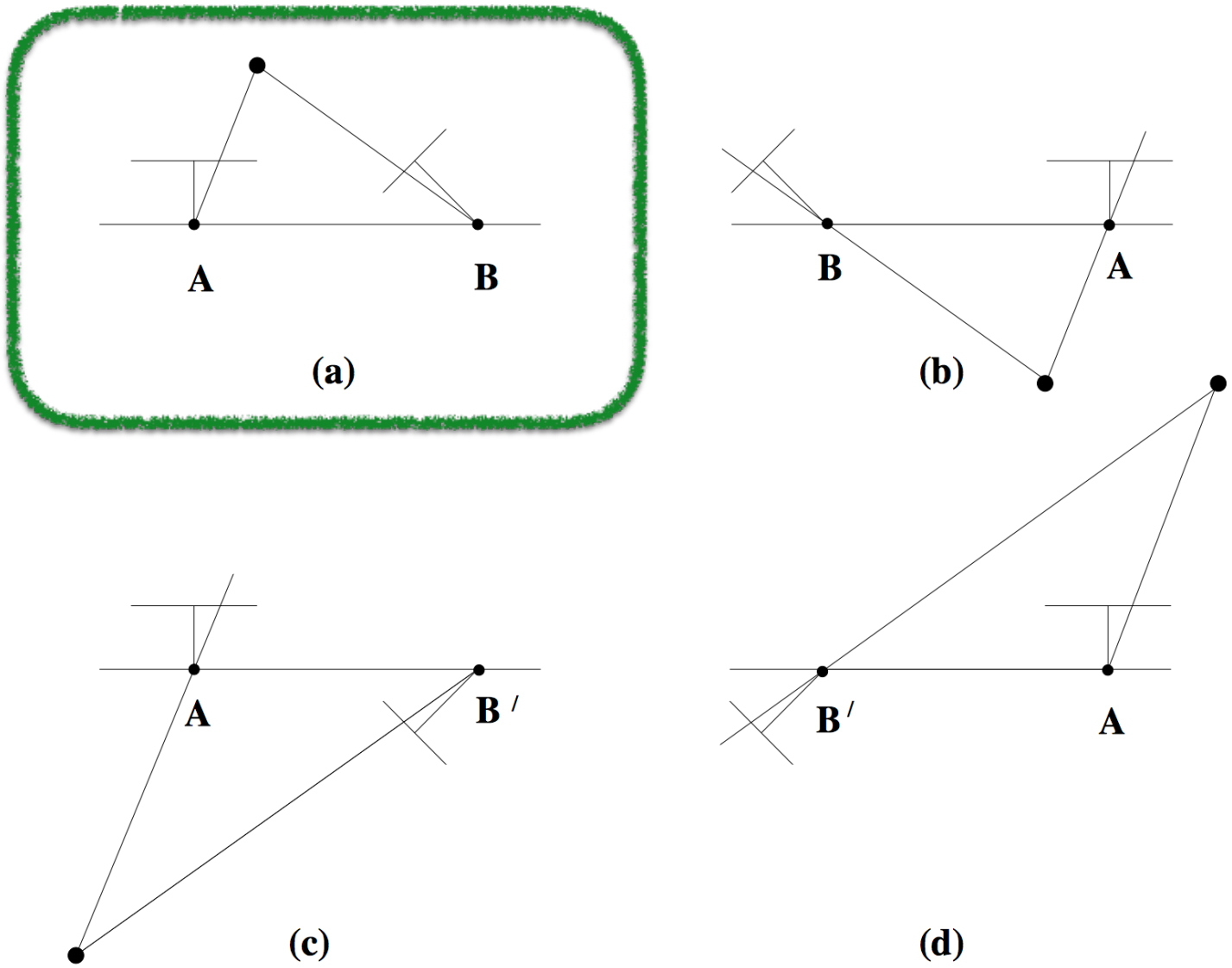


Find the configuration where the point is in front of both cameras

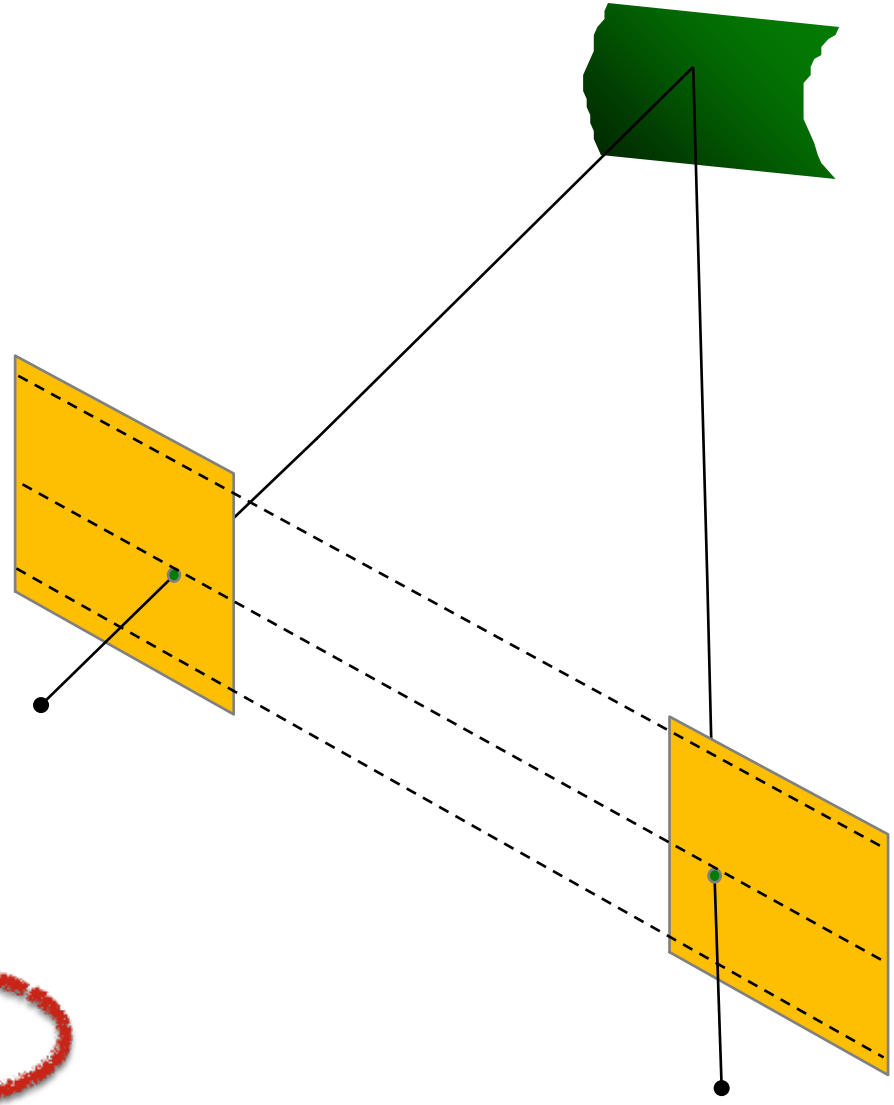
Find the configuration where the points is in front of both cameras



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**

Stereo Rectification

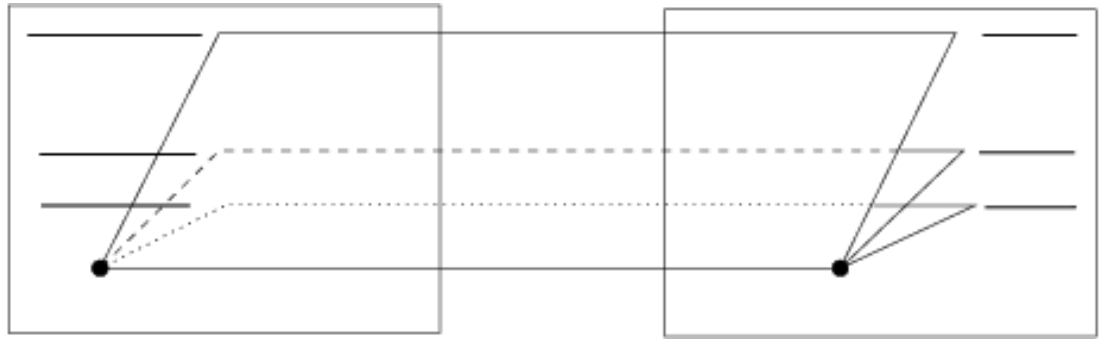
1. **Rotate** the right camera by **R**
   (aligns camera coordinate system orientation only)

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**

Parallel cameras

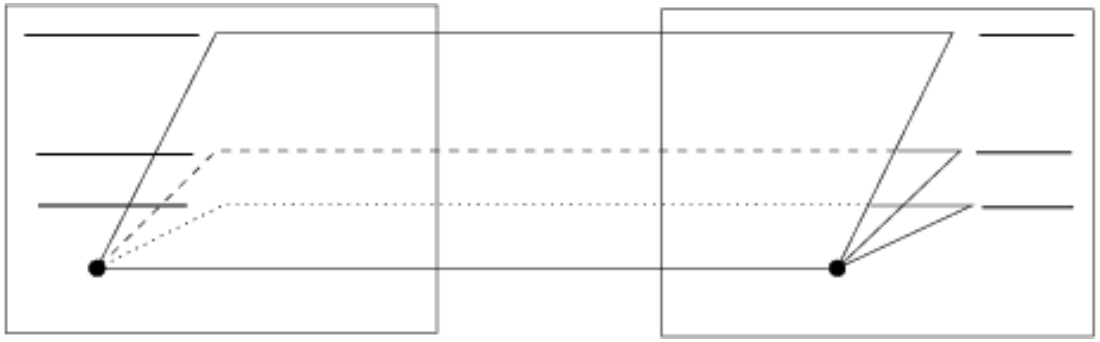


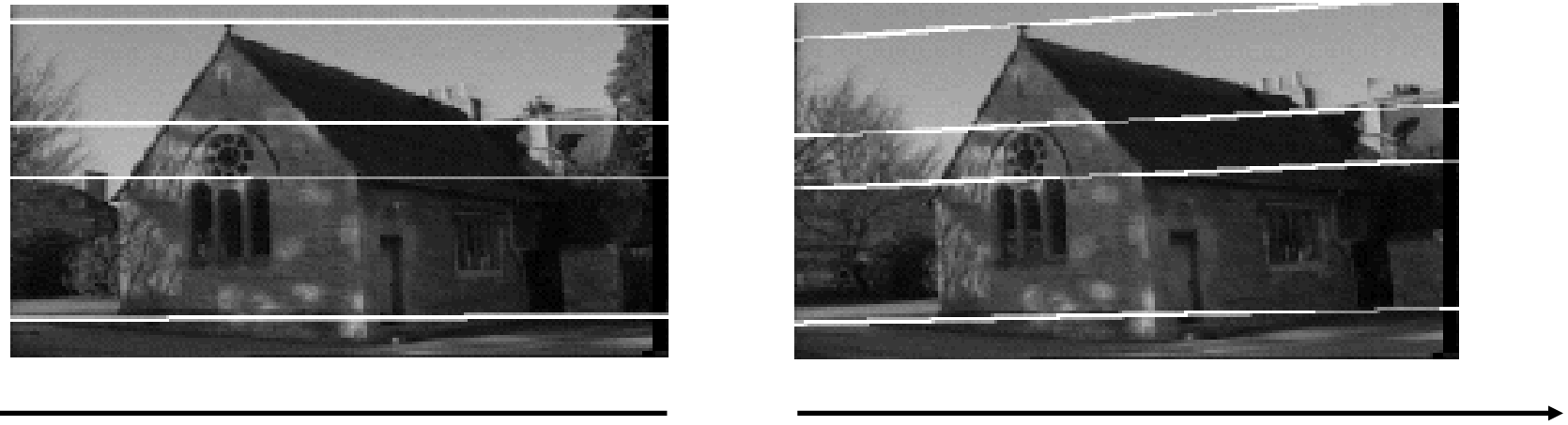




Where is the epipole?

Parallel cameras





epipole at infinity

Setting the epipole to infinity

(Building $\mathbf{R}_{\text{rect}}$ 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** (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 $\boldsymbol{r}_1 = \boldsymbol{e}_1 = \dfrac{T}{||T||}$ 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 $\boldsymbol{r}_1 = \boldsymbol{e}_1 = \dfrac{T}{||T||}$ 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?

If $\boldsymbol{r}_1 = \boldsymbol{e}_1 = \dfrac{T}{||T||}$ 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?

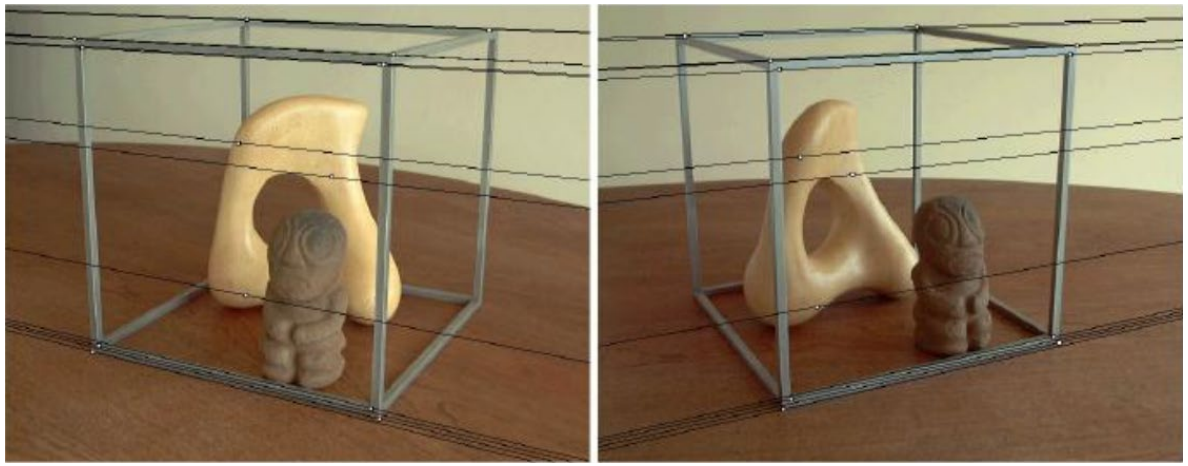
At x-infinity

Stereo Rectification Algorithm

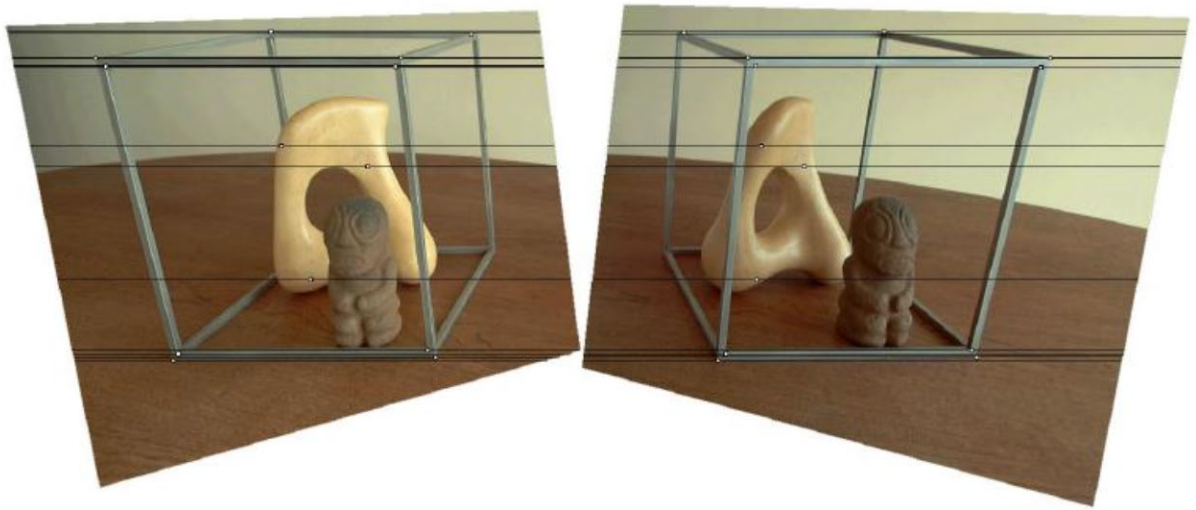
1. Estimate **E** using the 8 point algorithm (SVD)
2. Estimate the epipole **e** (SVD of **E**)
3. Build $\mathbf{R}_{rect}$ from **e**
4. Decompose **E** into **R** and **T**
5. Set $\mathbf{R}_1 = \mathbf{R}_{rect}$ and $\mathbf{R}_2 = \mathbf{R}\mathbf{R}_{rect}$
6. Rotate each left camera point (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 $\mathbf{R}_2$

Stereo Rectification Algorithm

1. Estimate **E** using the 8 point algorithm
2. Estimate the epipole **e** (solve **Ee**=0)
3. Build $\mathbf{R}_{rect}$ from **e**
4. Decompose **E** into **R** and **T**
5. Set $\mathbf{R}_1=\mathbf{R}_{rect}$ and $\mathbf{R}_2 = \mathbf{R}\mathbf{R}_{rect}$
6. Rotate each left camera point $\mathbf{x'} \sim \mathbf{Hx}$ where $\mathbf{H} = \mathbf{K}\mathbf{R}_1$
   *You may need to alter the focal length (inside **K**) to keep points within the original image size
7. Repeat 6 and 7 for right camera points using $\mathbf{R}_2$



What do we do after rectifying the two image planes?



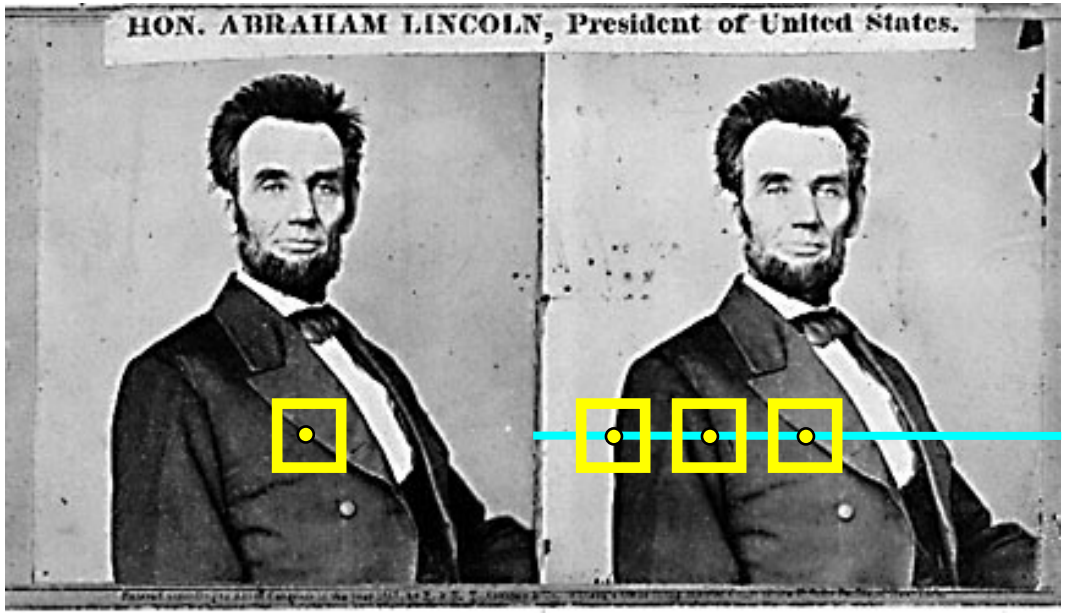
Stereo matching





Depth Estimation via Stereo Matching





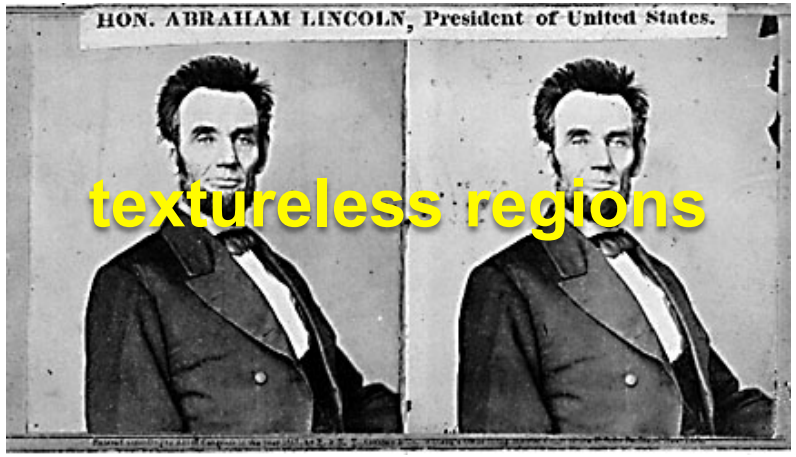
1. Rectify images
   (make epipolar lines horizontal)
2. For each pixel
   a. Find epipolar line
   b. Scan line for best match
   c. Compute depth from disparity

How would you do this?

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

When are correspondences difficult?

When are correspondences difficult?







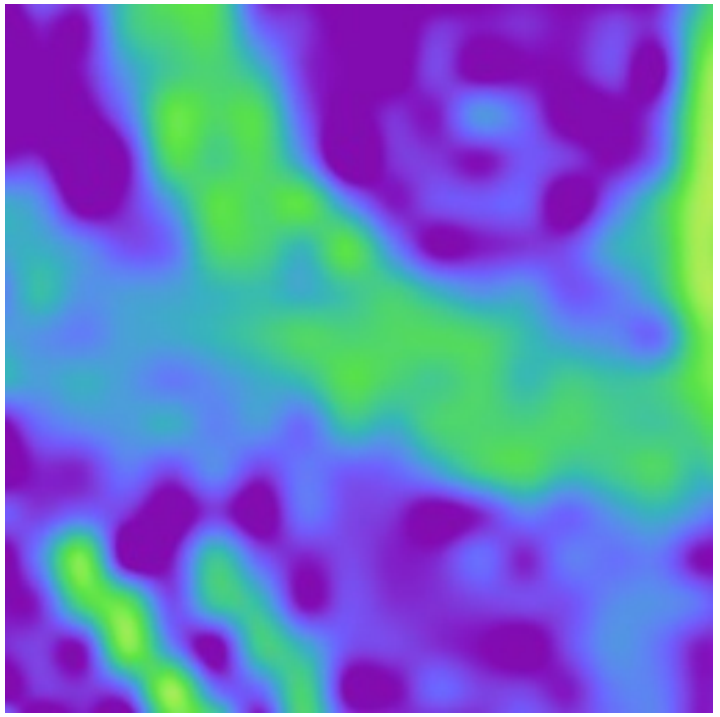


Depth discontinuities

What is the problem here?



One of two input images



Depth from disparity



Groundtruth depth

Depth discontinuities

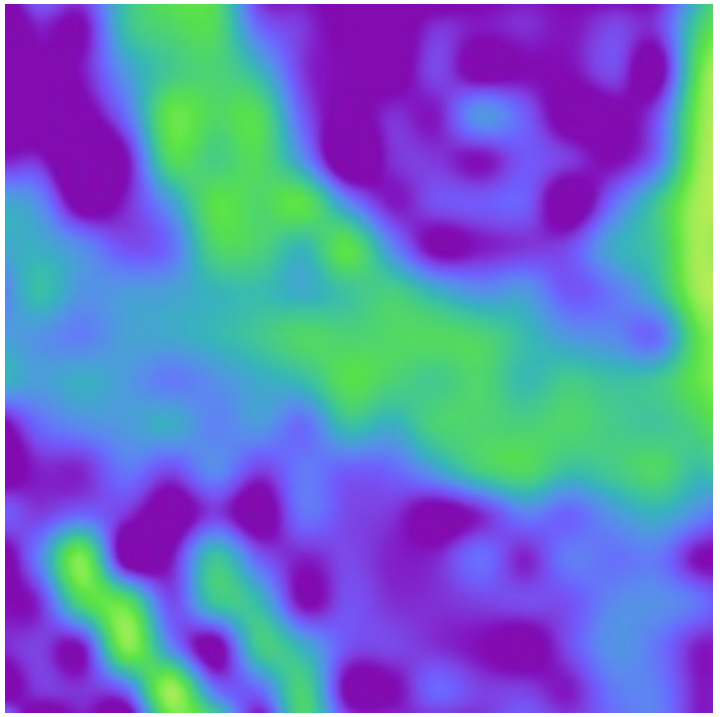
What is the problem here?

- (Patch-wise) stereo matching blurs along the edges.

How can we fix this?



One of two input images



Depth from disparity



Groundtruth depth

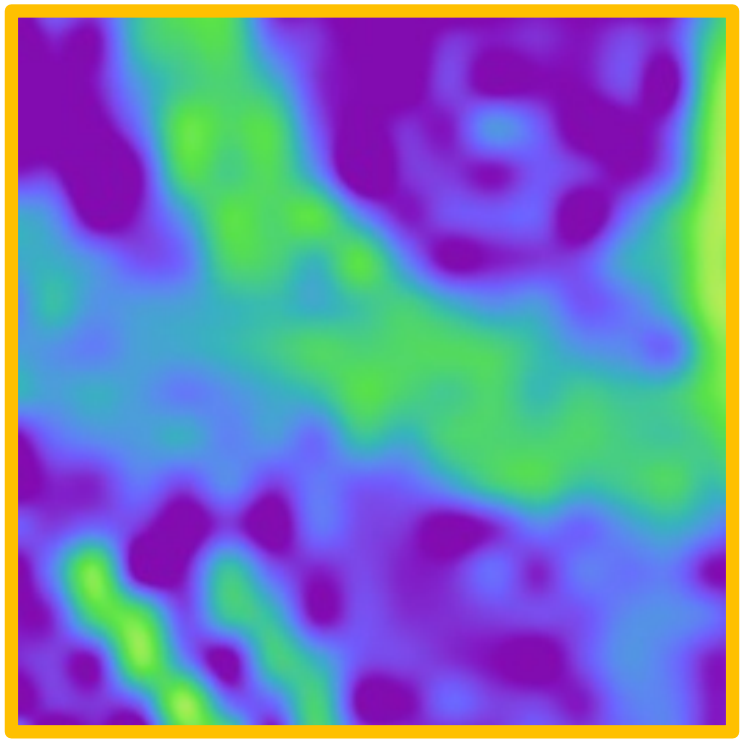
Edge-aware depth denoising

$$A_{p(col)} = \frac{1}{k(p(col))} \sum_{p' \in \Omega} g_d(|p - p'|)\, g_r(F_{p(col)} - F_{p'(col)}) A_{p'(col)}$$

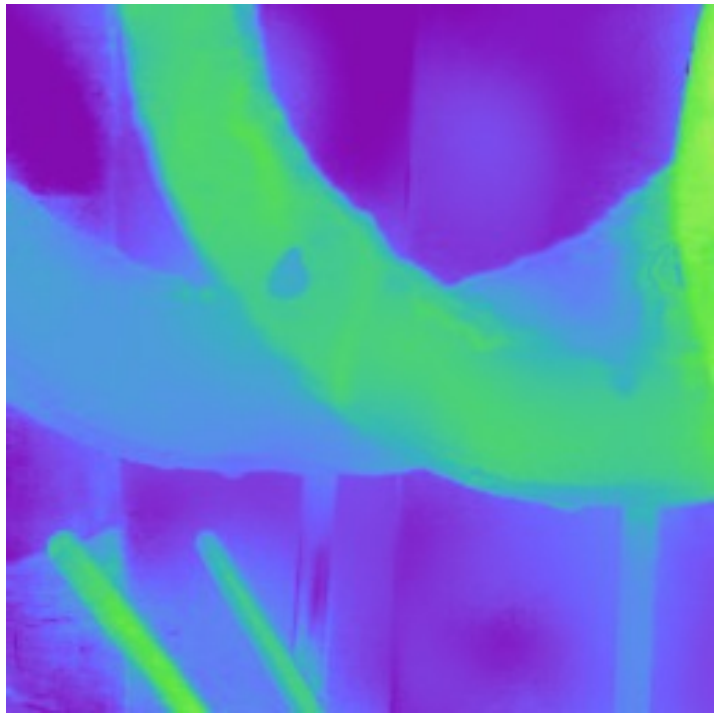
Use joint bilateral filtering, with the input image as guide.



One of two input images



Depth from disparity



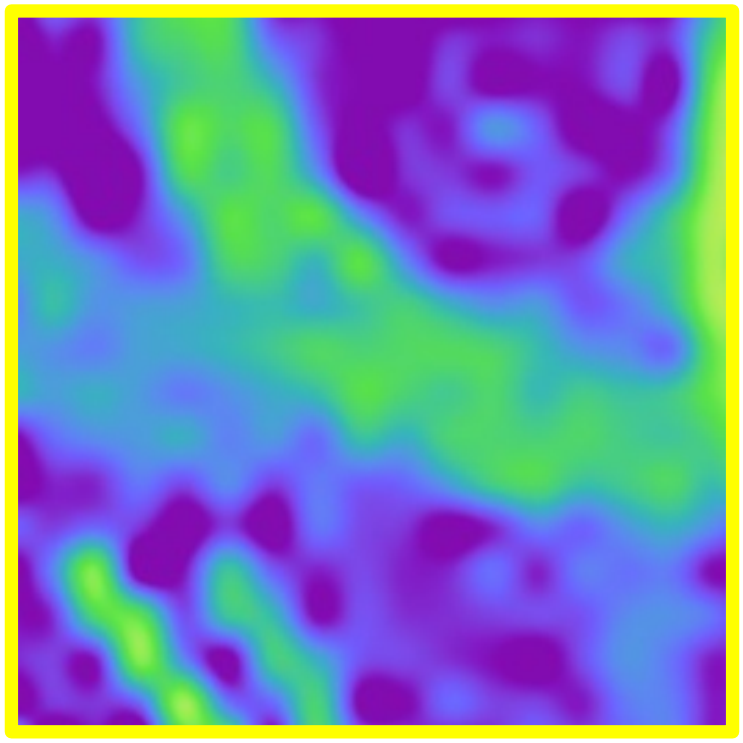
Guided filtering

Fast bilateral solver

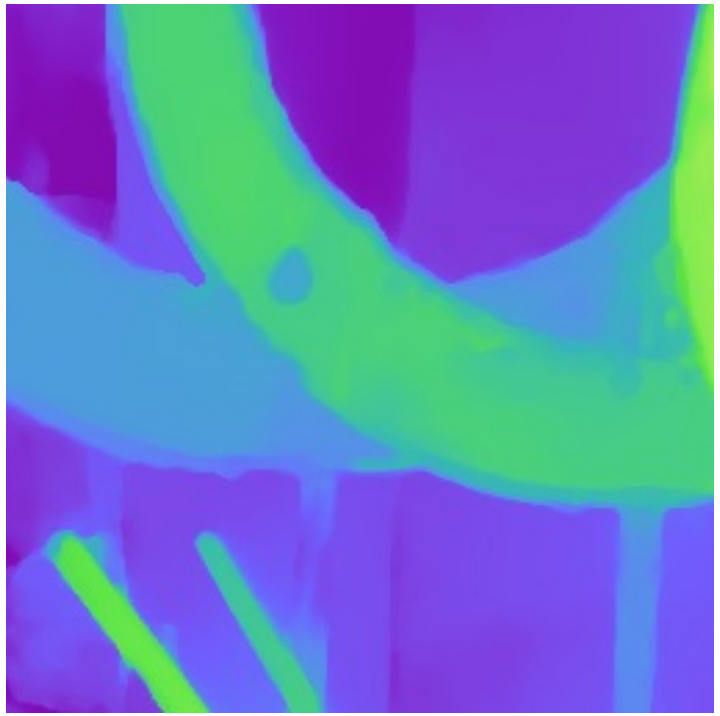
Possible to *combine* edge-enforcement and matching in a single optimization problem, instead of just filtering in post-processing.



One of two input images



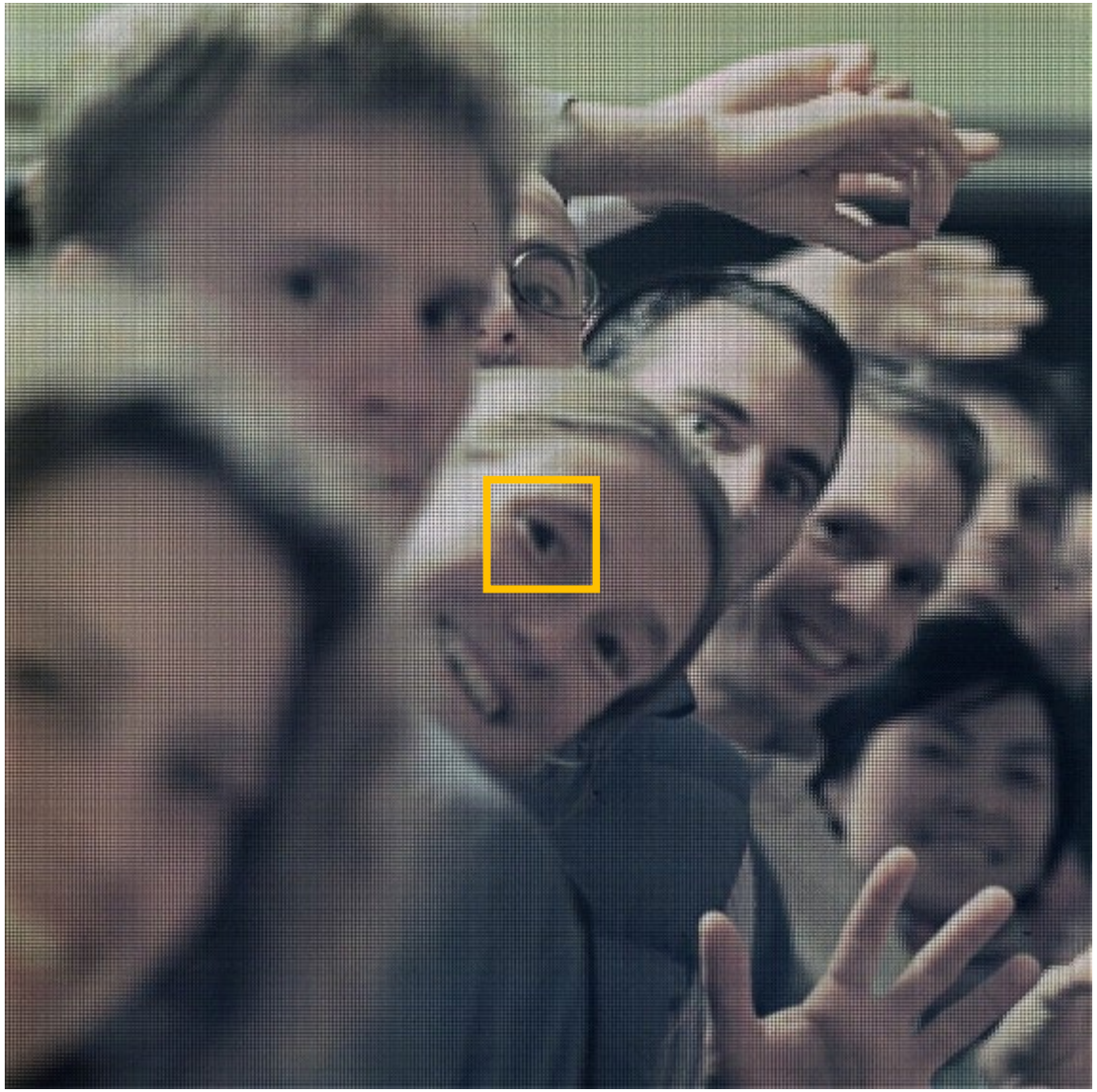
Depth from disparity

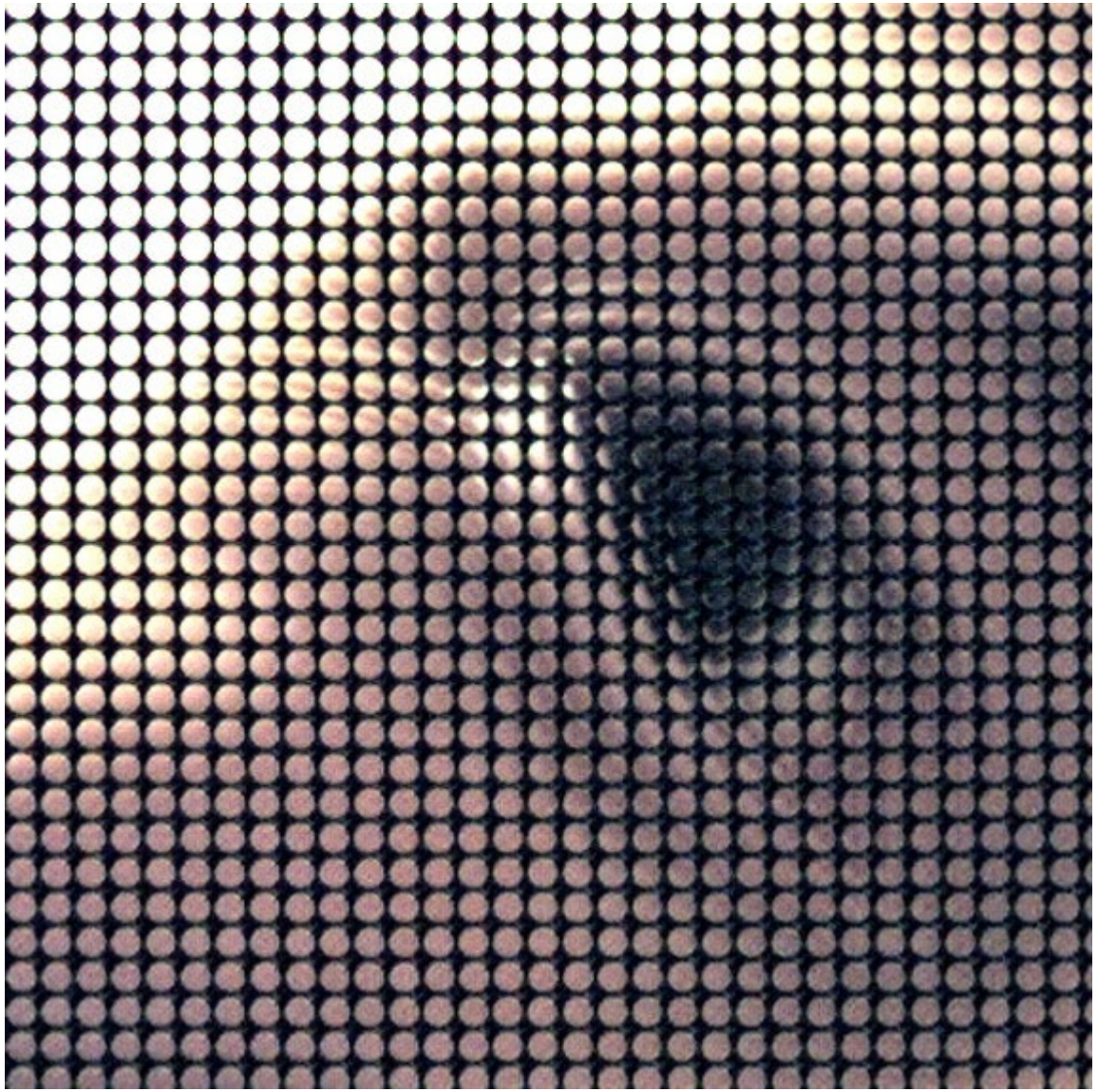


Bilateral stereo matching

Disparity and lightfields

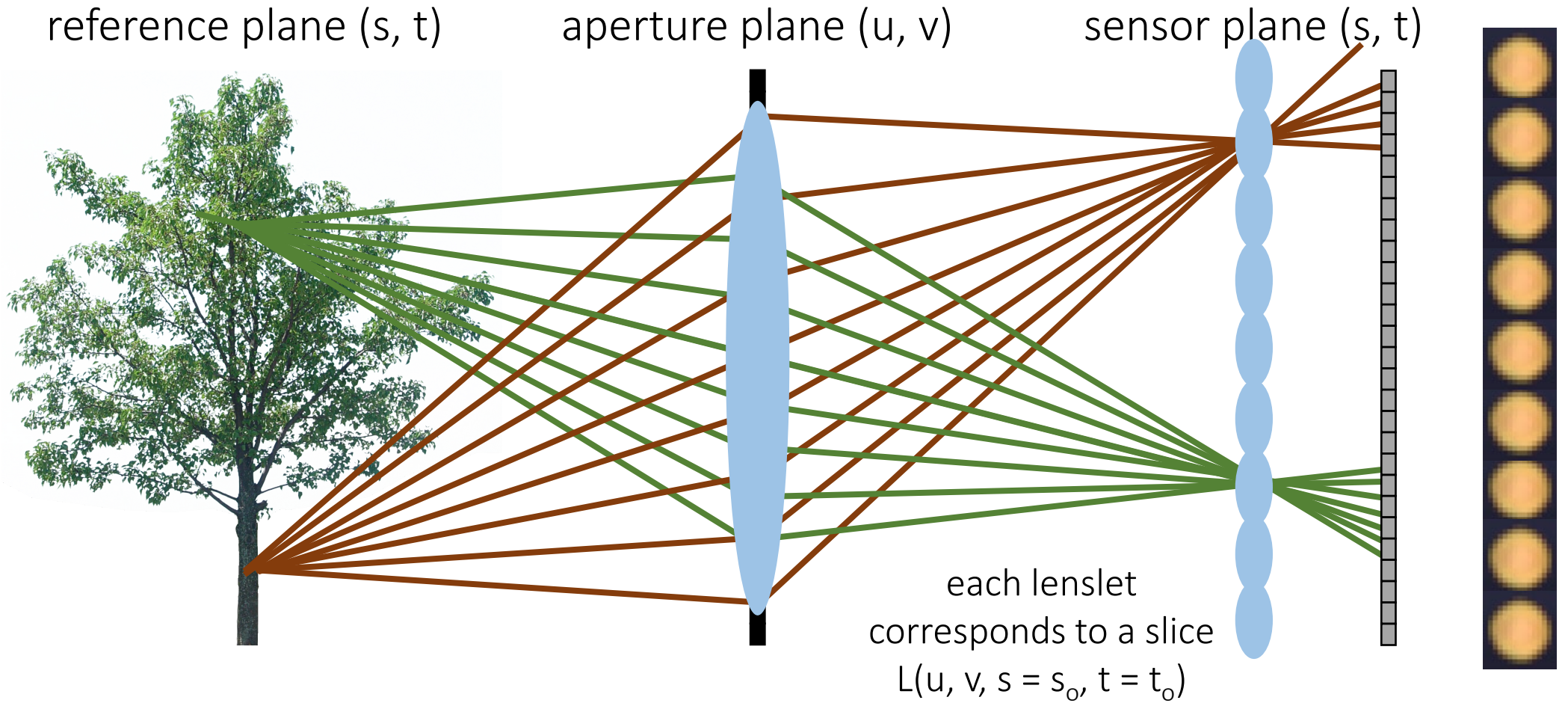
Reminder: a plenoptic “image”





What are these circles?

Reminder: a plenoptic camera

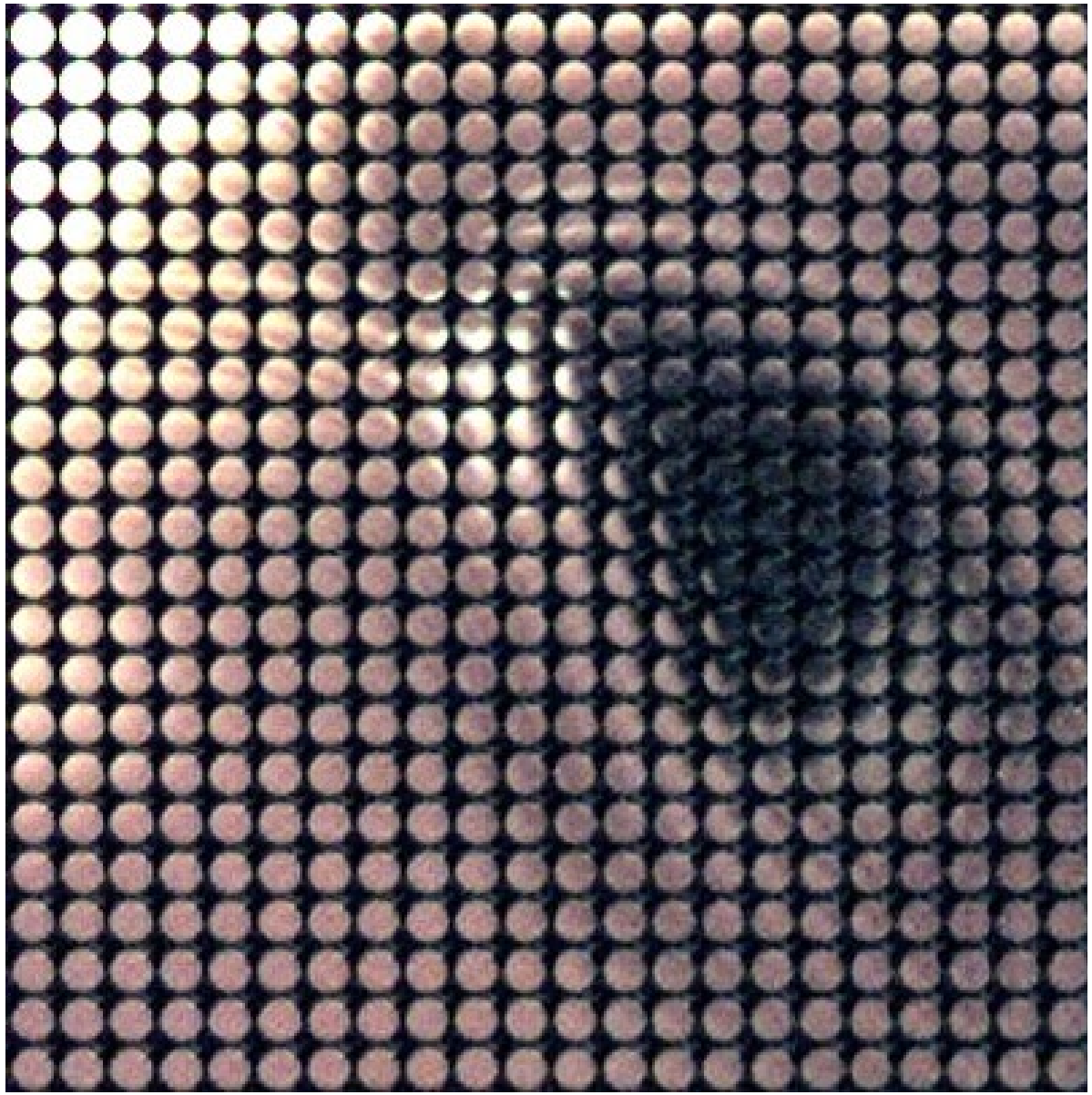


Lightfield L(u, v, s, t)

Reminder: the plenoptic image

Which coordinates do I change when I move from one circle to another?

Which coordinates do I change when I move within each circle?



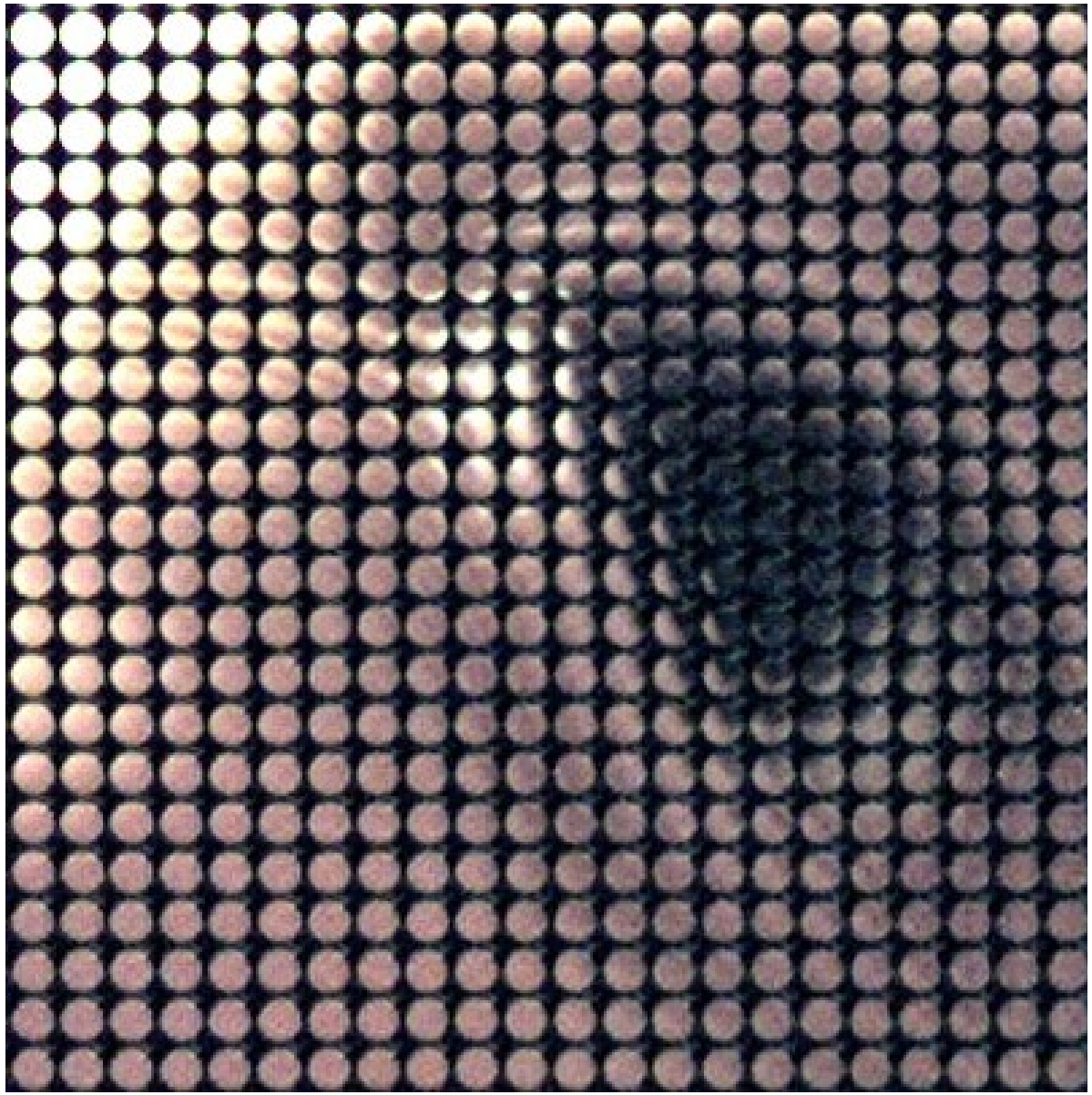
Reminder: the plenoptic image

Which coordinates do I change when I move from one circle to another?

- I change s, t (sensor plane) coordinates.

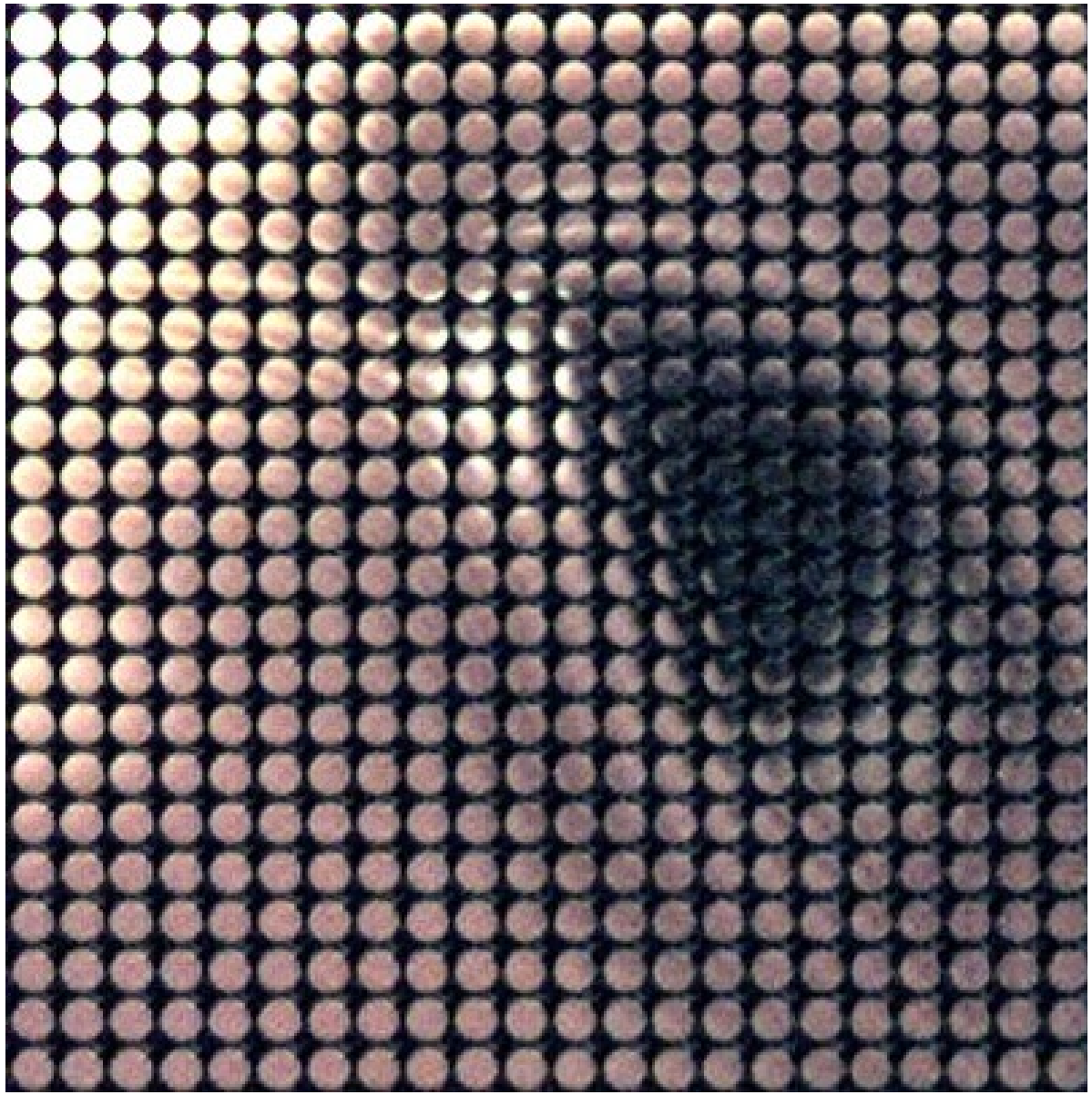
Which coordinates do I change when I move within each circle?

- I change u, v (aperture plane) coordinates.



Reminder: how do I…

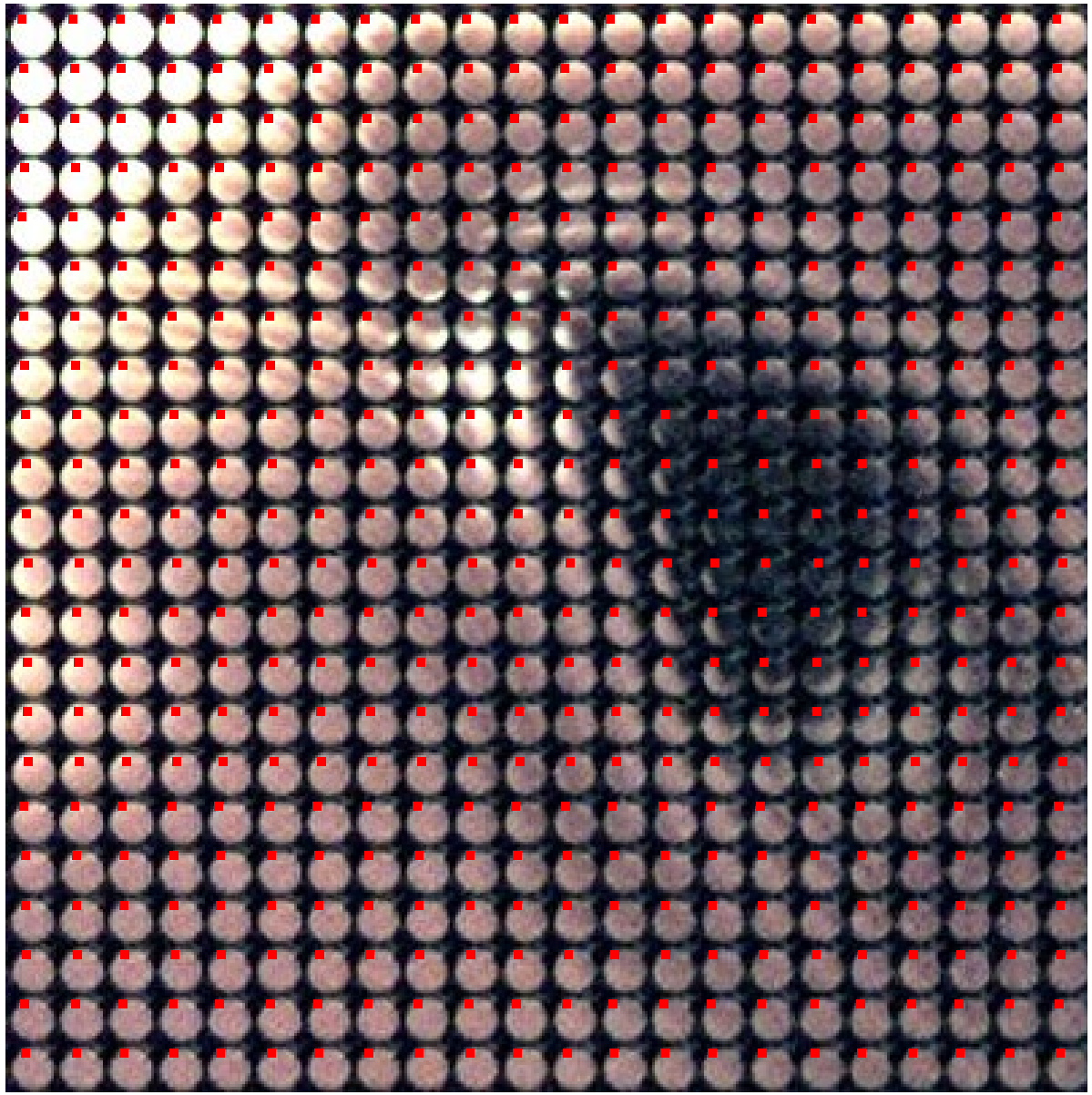
Simulate different viewpoints?



Reminder: how do I…

Simulate different viewpoints?

- Pick same pixel within each lenslet view

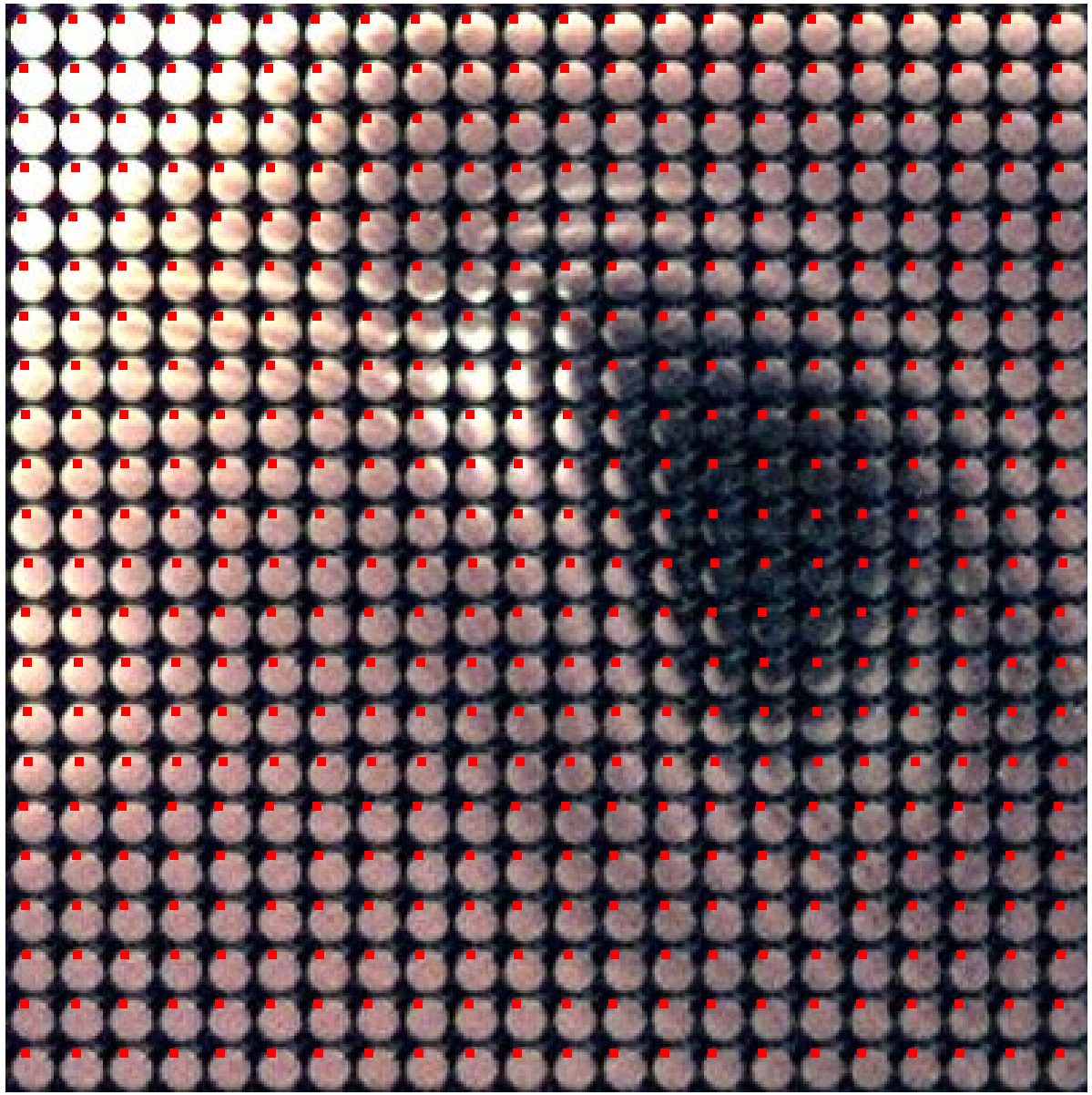


Reminder: how do I…

Simulate different viewpoints?

- Pick same pixel within each lenslet view

Simulate different aperture sizes?



Reminder: how do I…

Simulate different viewpoints?

- Pick same pixel within each lenslet view

Simulate different aperture sizes?

- Sum more than one pixels within each lenslet view



Reminder: how do I…

Simulate different viewpoints?

- Pick same pixel within each lenslet view

Simulate different aperture sizes?

- Sum more than one pixels within each lenslet view

Simulate lens at current focus setting?



Reminder: how do I…

Simulate different viewpoints?

- Pick same pixel within each lenslet view

Simulate different aperture sizes?

- Sum more than one pixels within each lenslet view

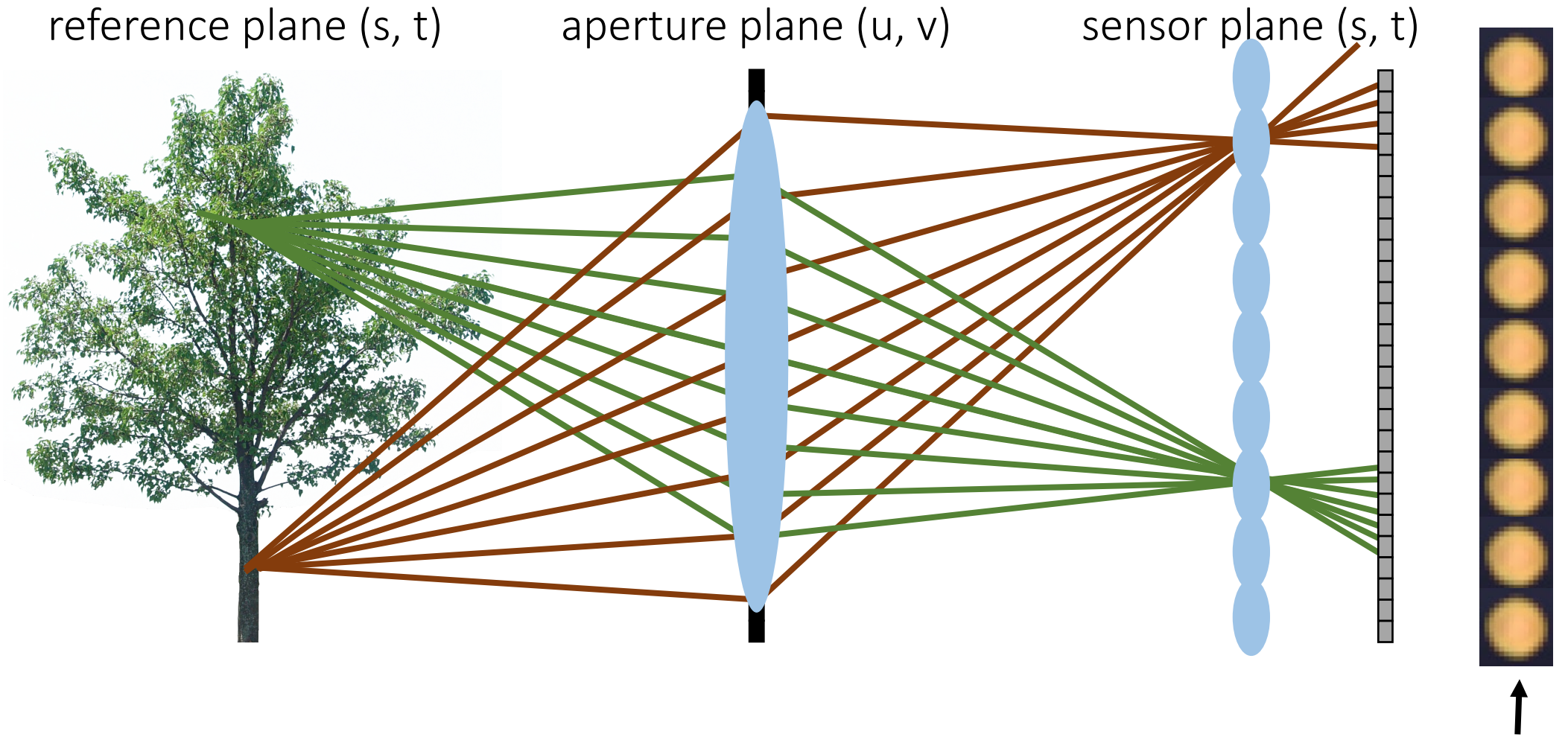
Simulate lens at current focus setting?

- Same as above. Sum all pixels for max aperture setting.

How do I change focus setting?



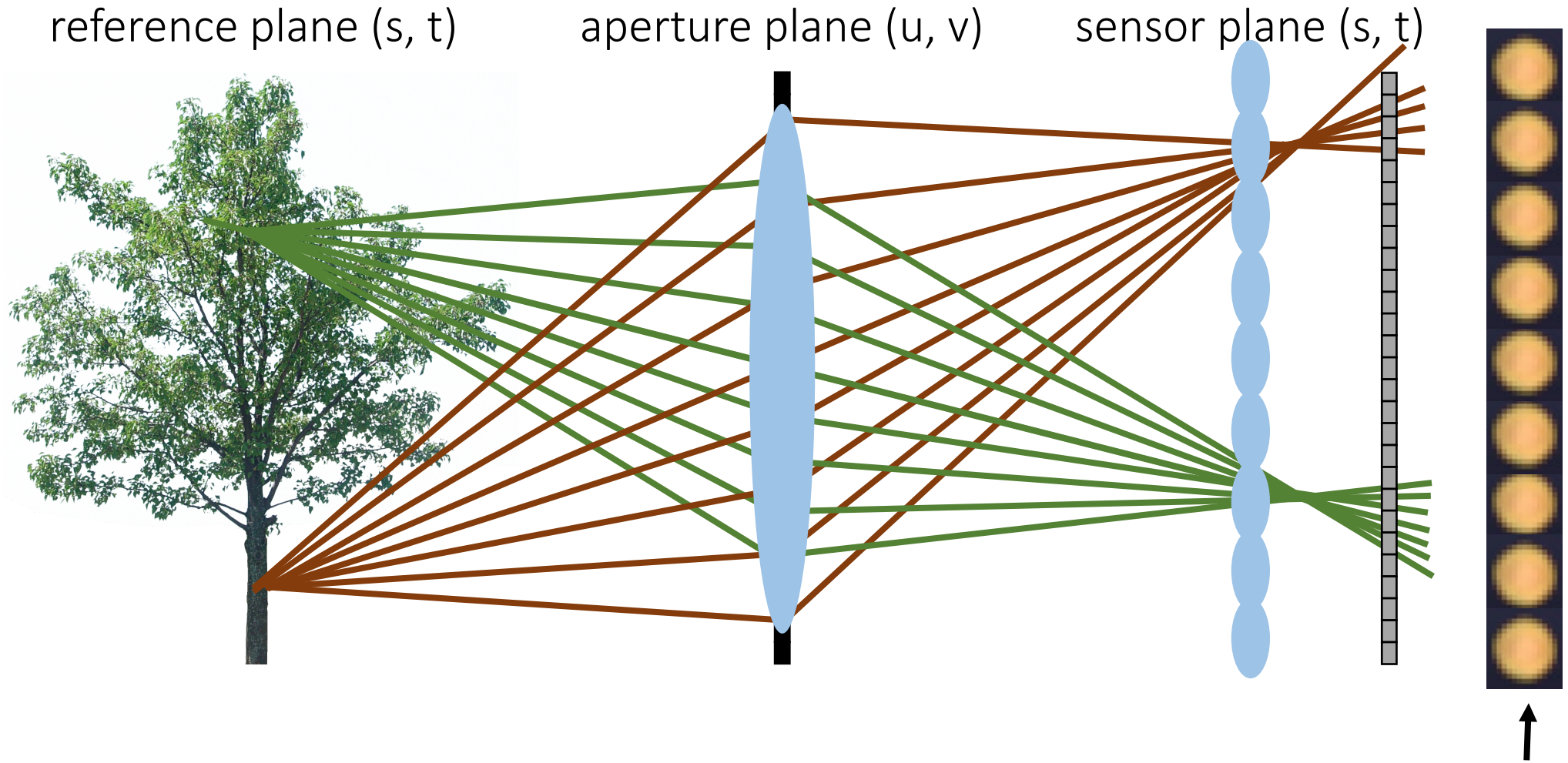
Reminder: form lens image



Sum all pixels in each lenslet view.

How do I refocus?

Reminder: form lens image



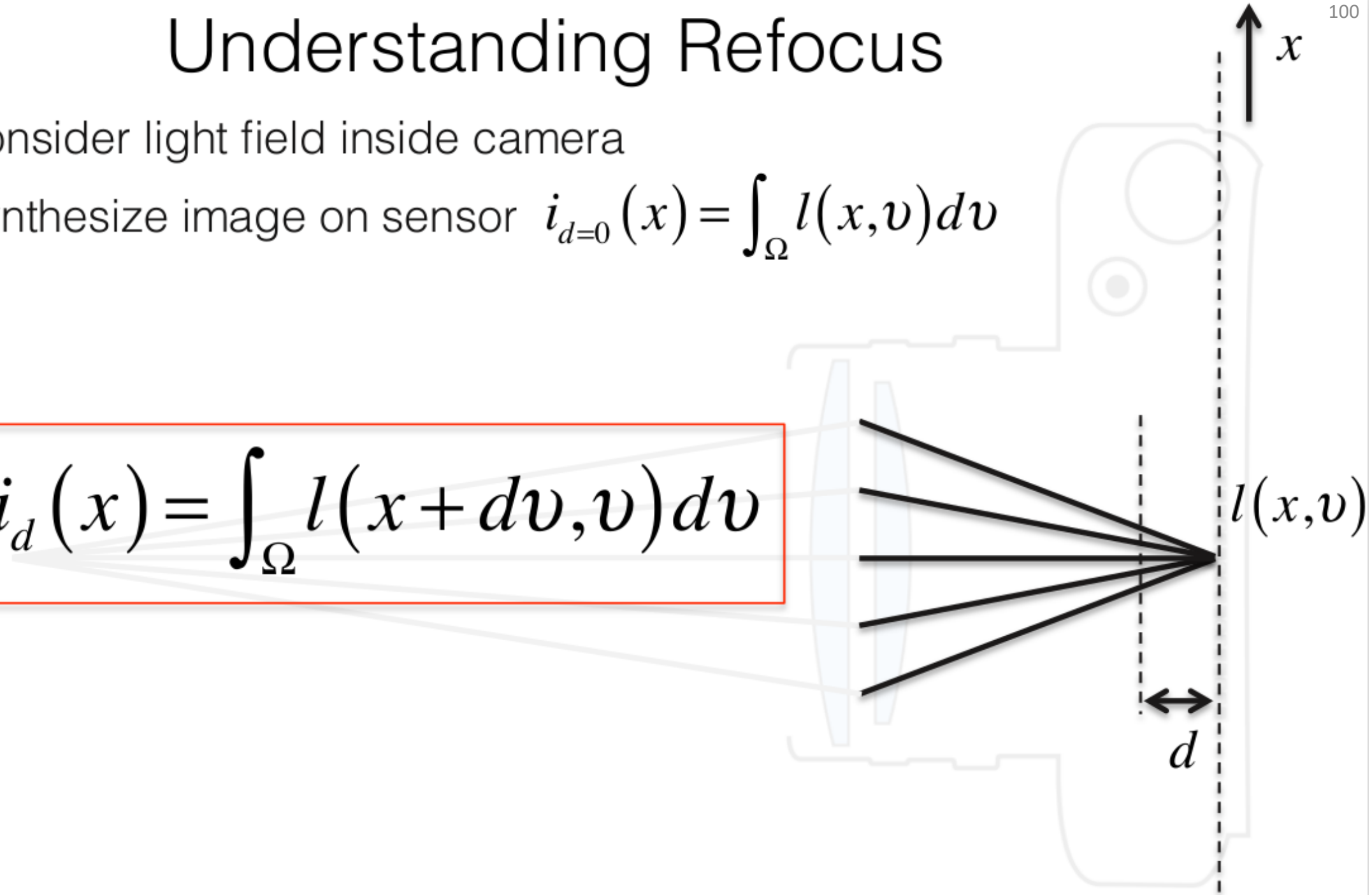
How do I refocus?

- Need to move sensor plane to a different location.

Understanding Refocus

- consider light field inside camera
- synthesize image on sensor $i_{d=0}(x) = \int_{\Omega} l(x,\upsilon) d\upsilon$

$$i_d(x) = \int_{\Omega} l(x + d\upsilon, \upsilon) d\upsilon$$

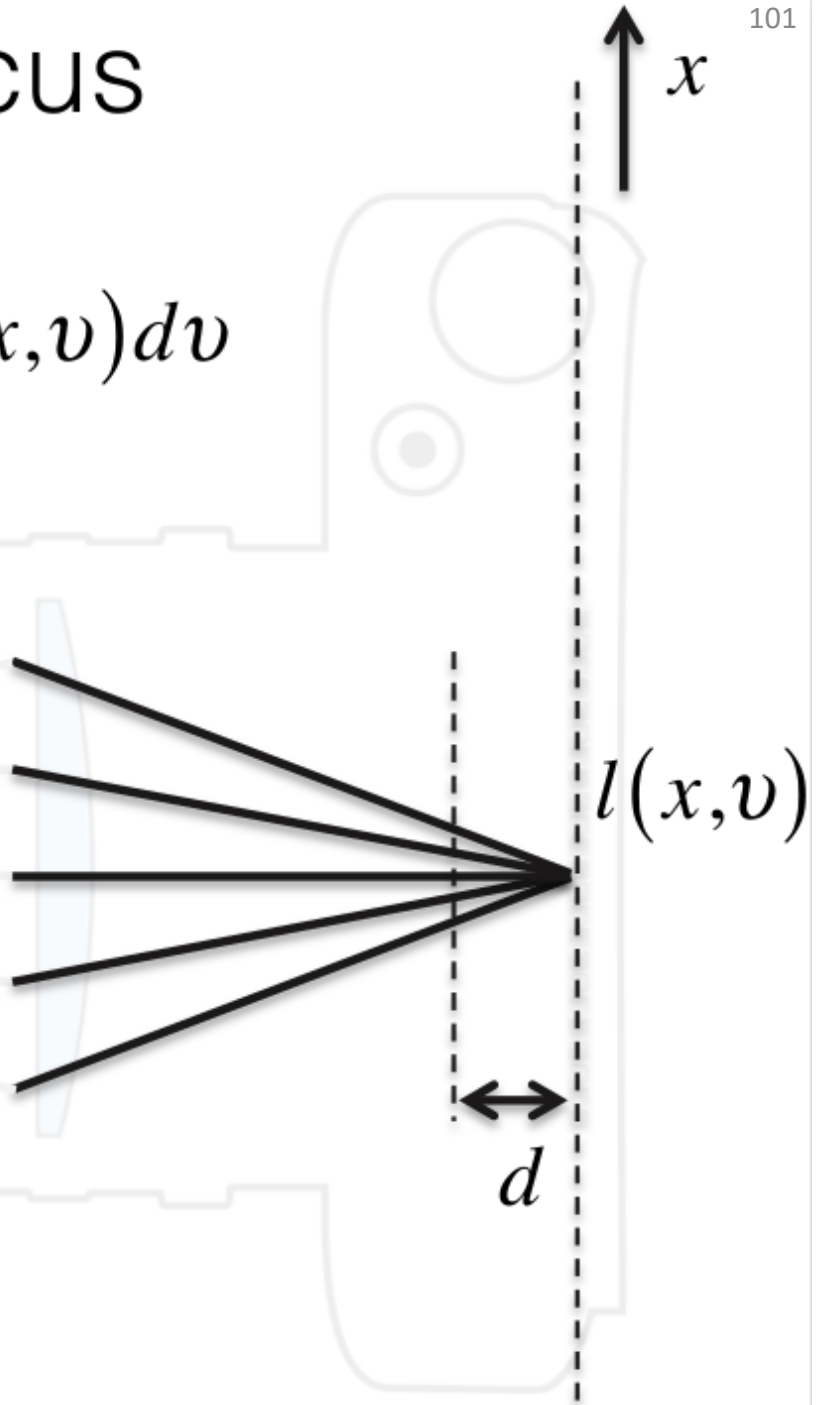


Understanding Refocus

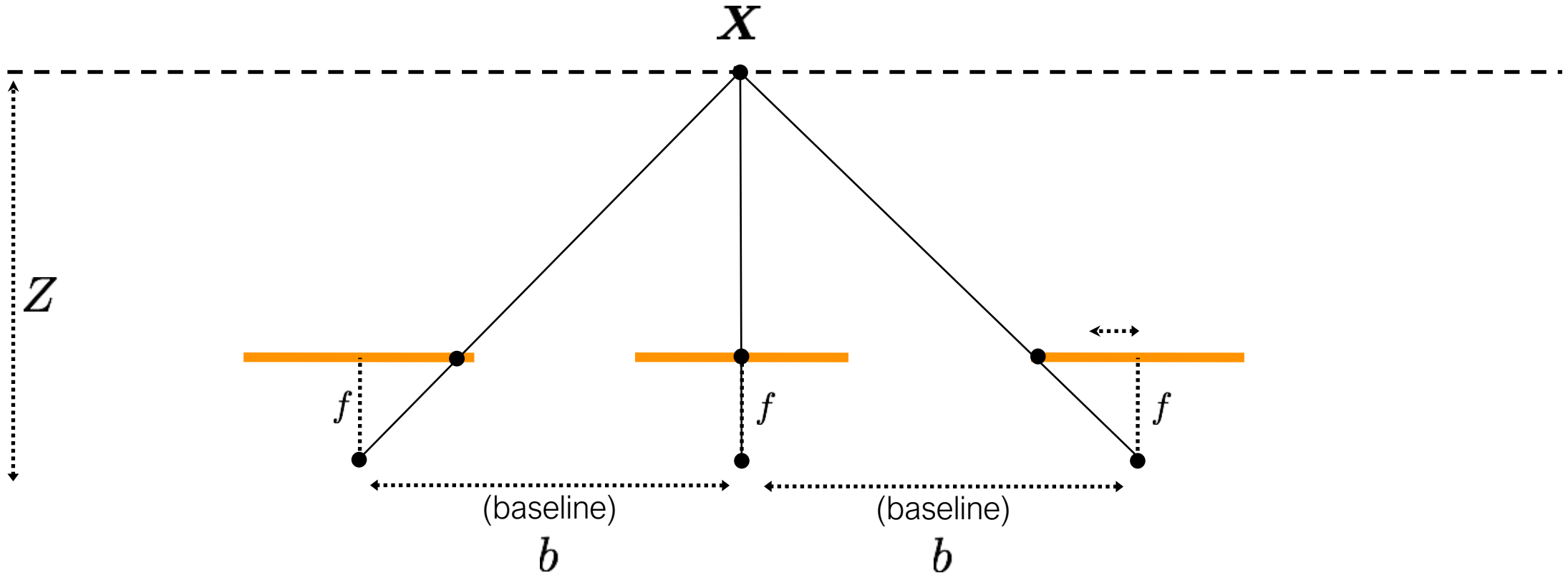
- consider light field inside camera
- synthesize image on sensor $i_{d=0}(x) = \int_{\Omega} l(x,\upsilon) d\upsilon$

$$i_d(x) = \int_{\Omega} l(x + d\upsilon, \upsilon) d\upsilon$$

Where did this equation come from?

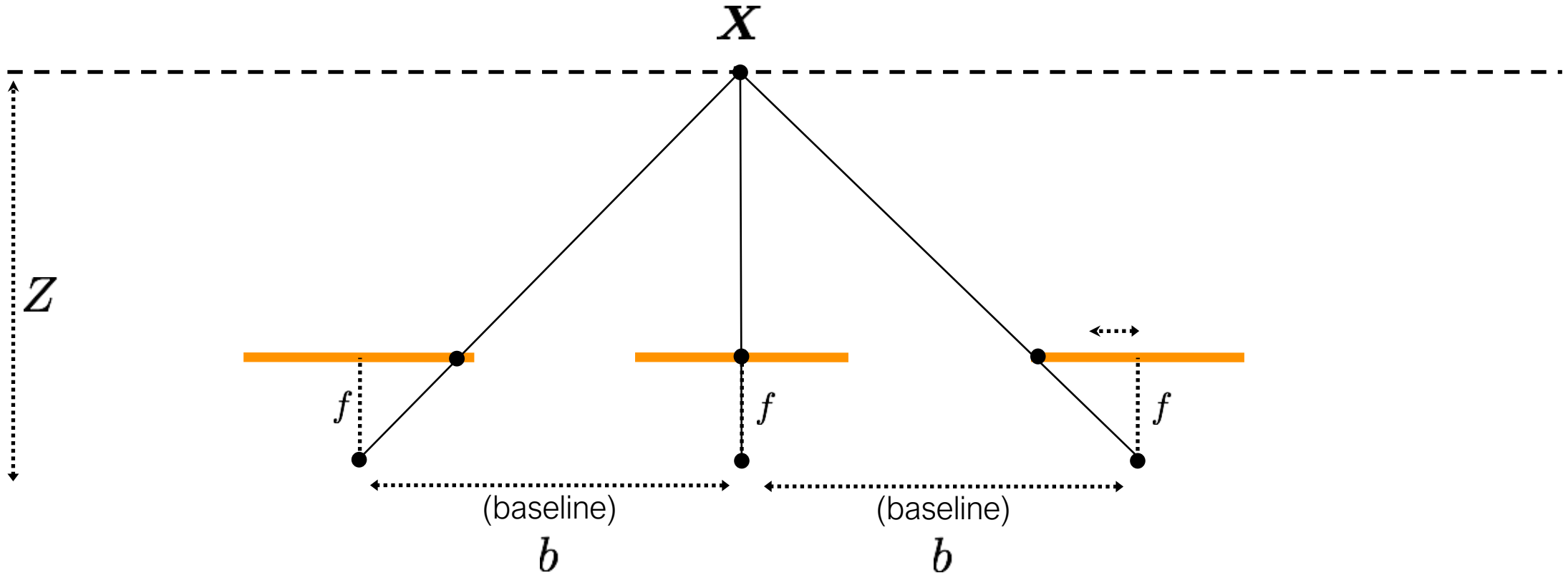


Stereo view of a lightfield camera



What are the different “cameras” in the lightfield case?

Stereo view of a lightfield camera



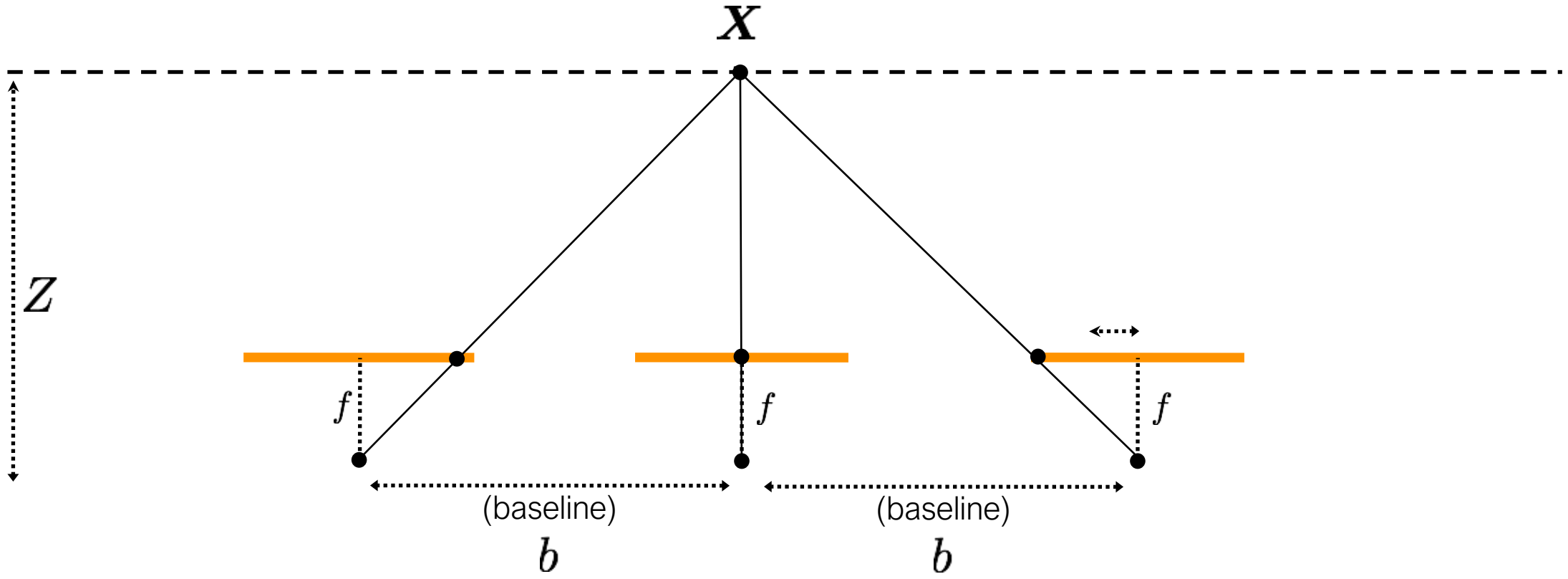
What are the different “cameras” in the lightfield case?

- Different aperture views L(u = u_o, v = v_o, s, t).

By how much do I need to shift each aperture to focus (i.e., *align*) at depth Z?

.

Stereo view of a lightfield camera



What are the different “cameras” in the lightfield case?

- Different aperture views L(u = u_o, v = v_o, s, t).

By how much do I need to shift each aperture to focus (i.e., *align*) at depth Z?

- By an amount equal to the disparity relative to the center view for depth Z.

Refocusing example



Refocusing example



Refocusing example

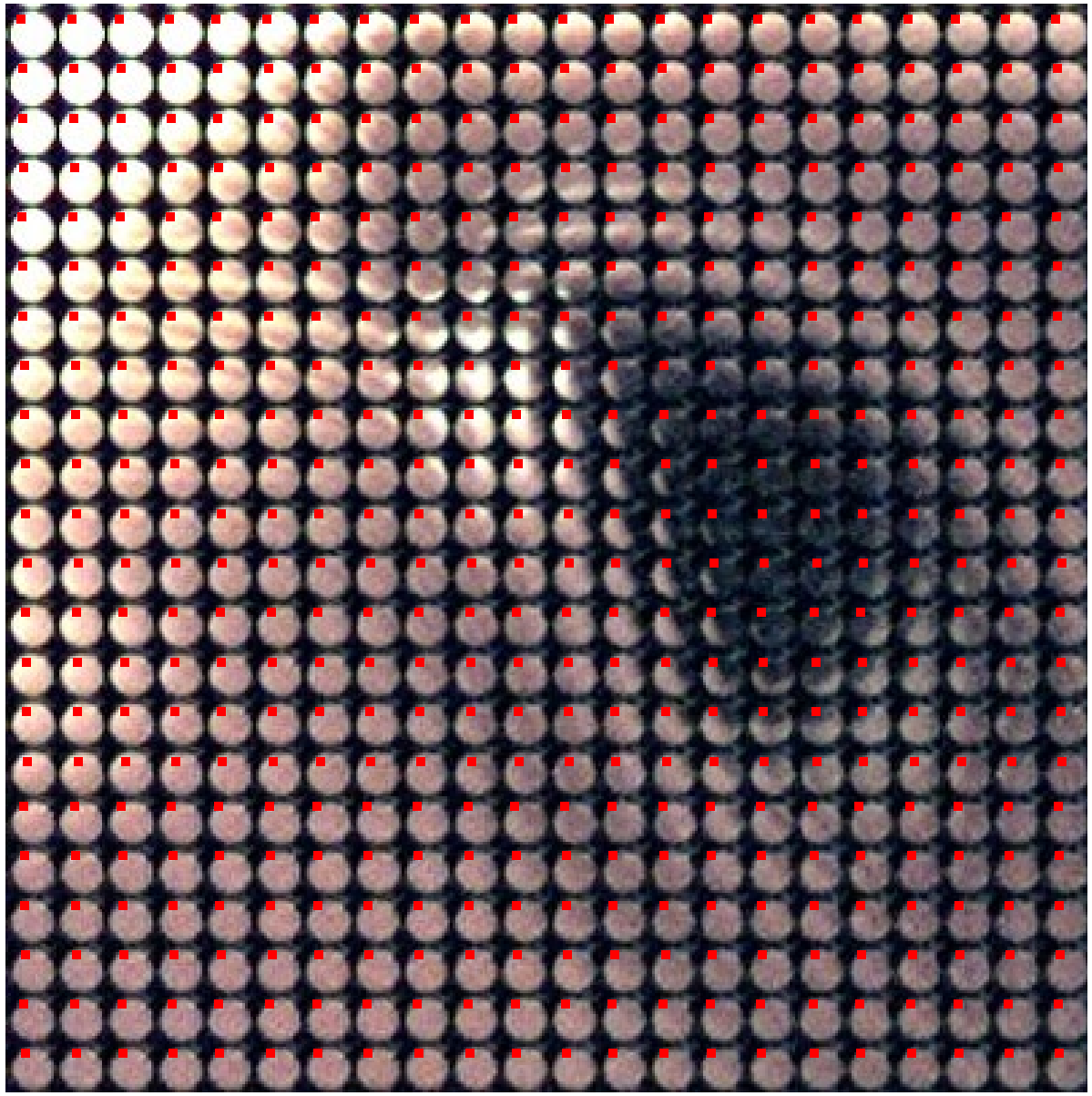


3D from lightfield

Simulate different viewpoints?

- Pick same pixel within each aperture view

Can we use different viewpoints for stereo?



3D from lightfield

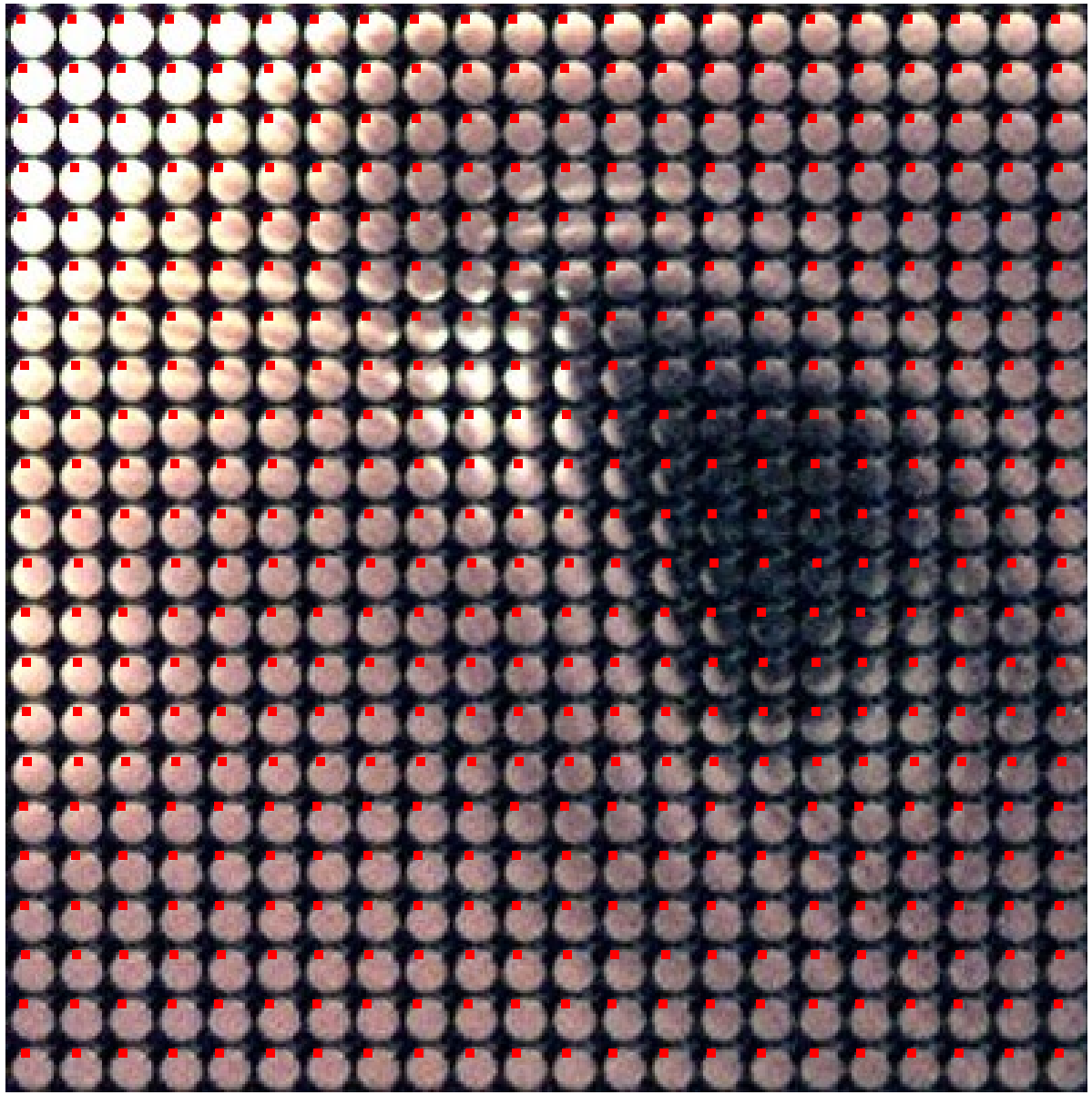
Simulate different viewpoints?

- Pick same pixel within each aperture view

Can we use different viewpoints for stereo?

- Very small baseline to use disparity algorithm.
- Standard algorithm only works with two views.

Can we do something better?



3D from lightfield

Simulate different viewpoints?

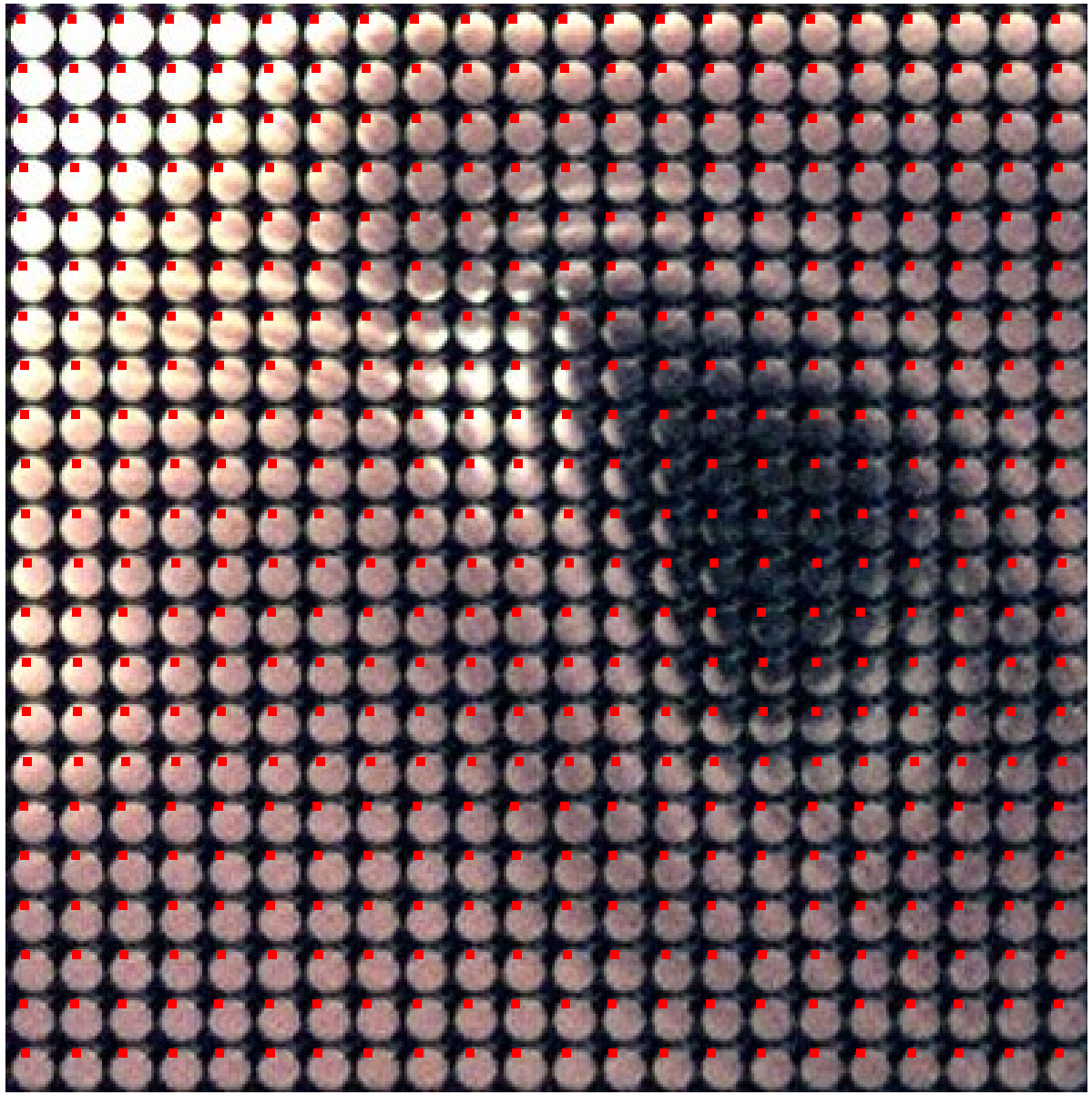
- Pick same pixel within each aperture view

Can we use different viewpoints for stereo?

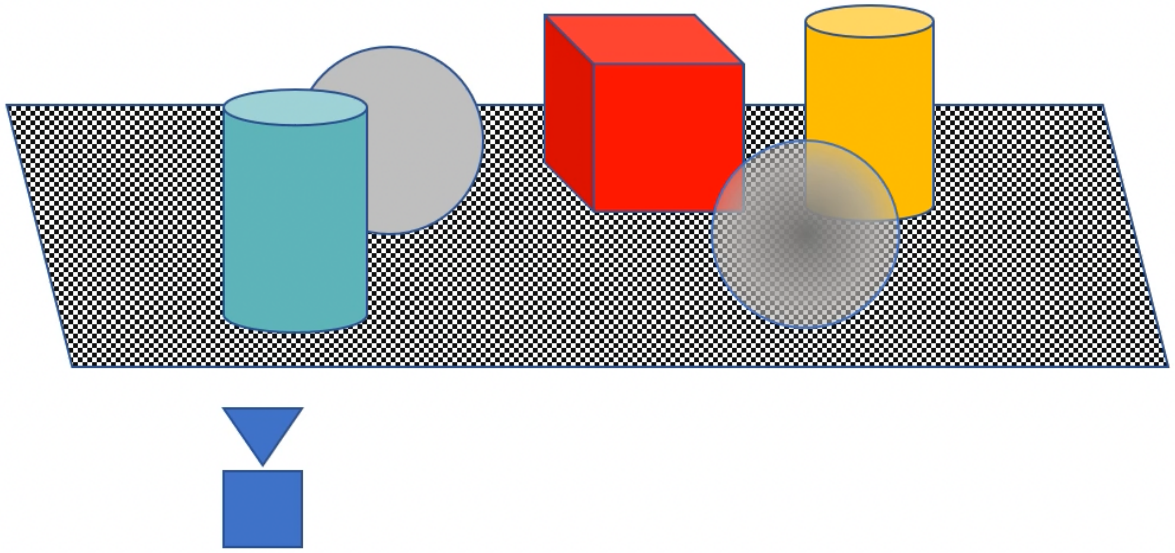
- Very small baseline to use disparity algorithm.
- Standard algorithm only works with two views.

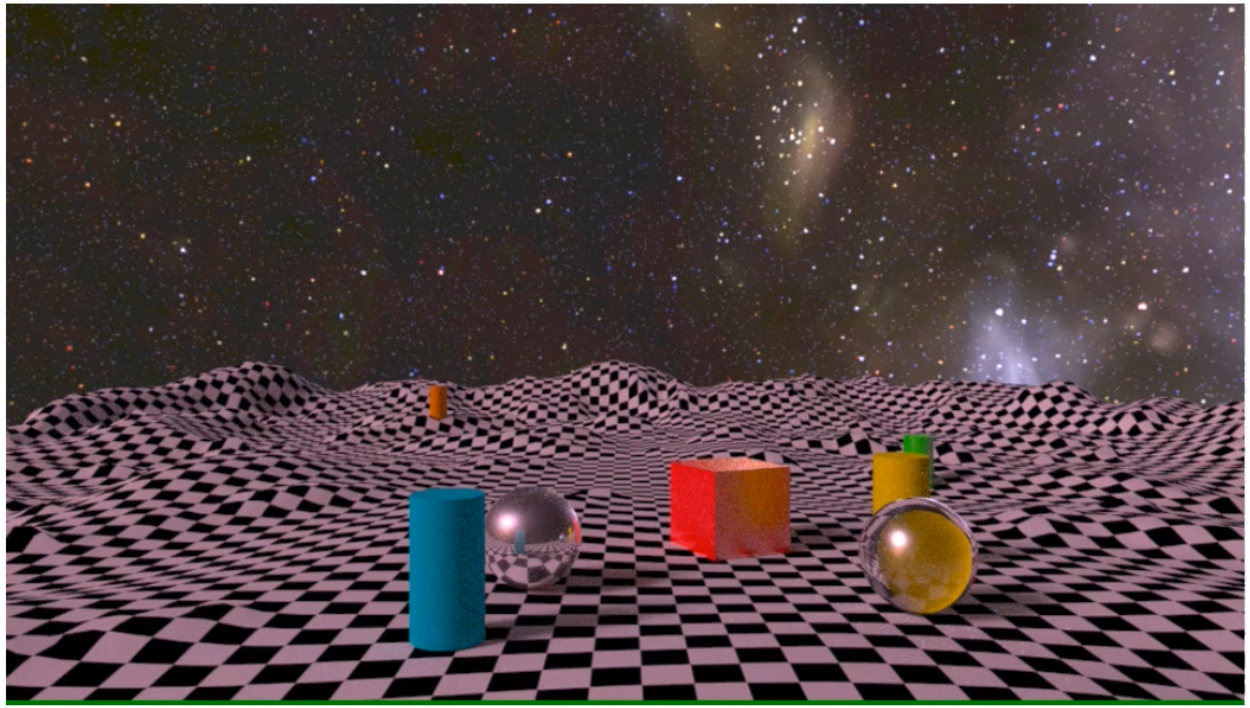
Can we do something better?

- Take advantage of *dense* set of views.
- Use disparity to explain changes in views.



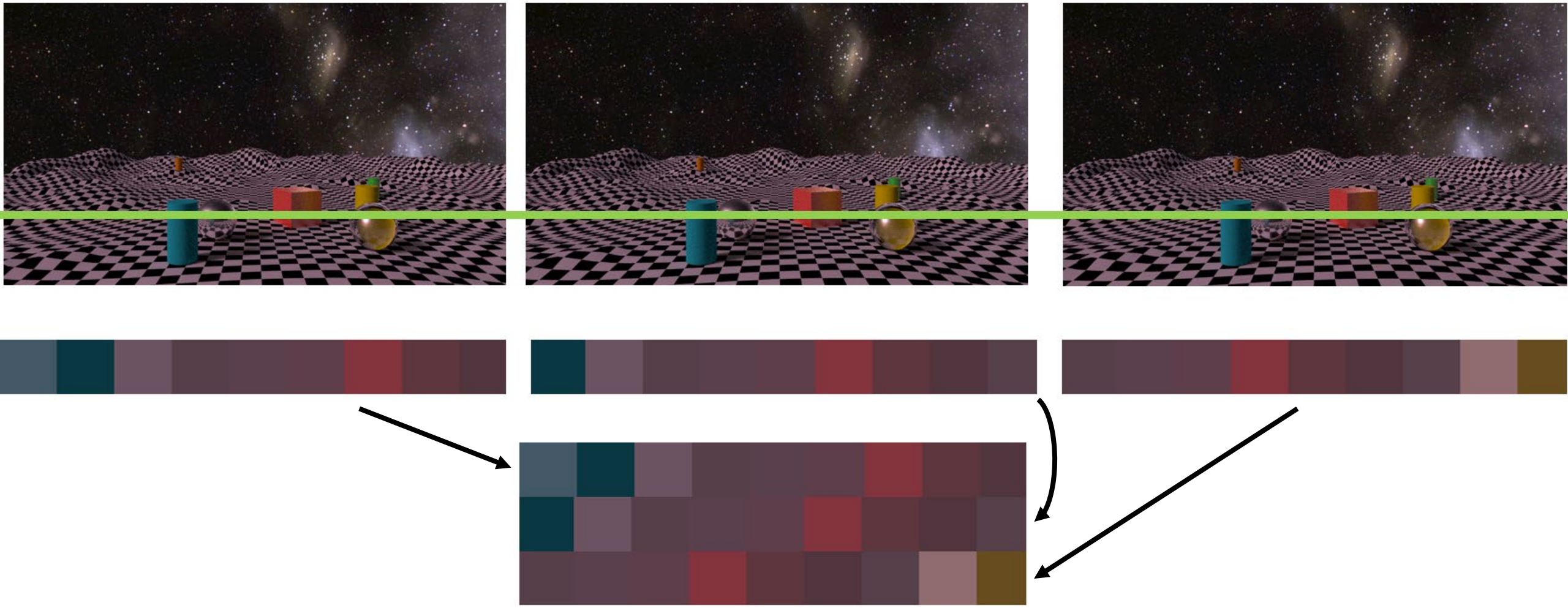
Epipolar plane images (EPIs)





Use lightfield to synthesize images for all aperture views on a horizontal line (*scanline*).

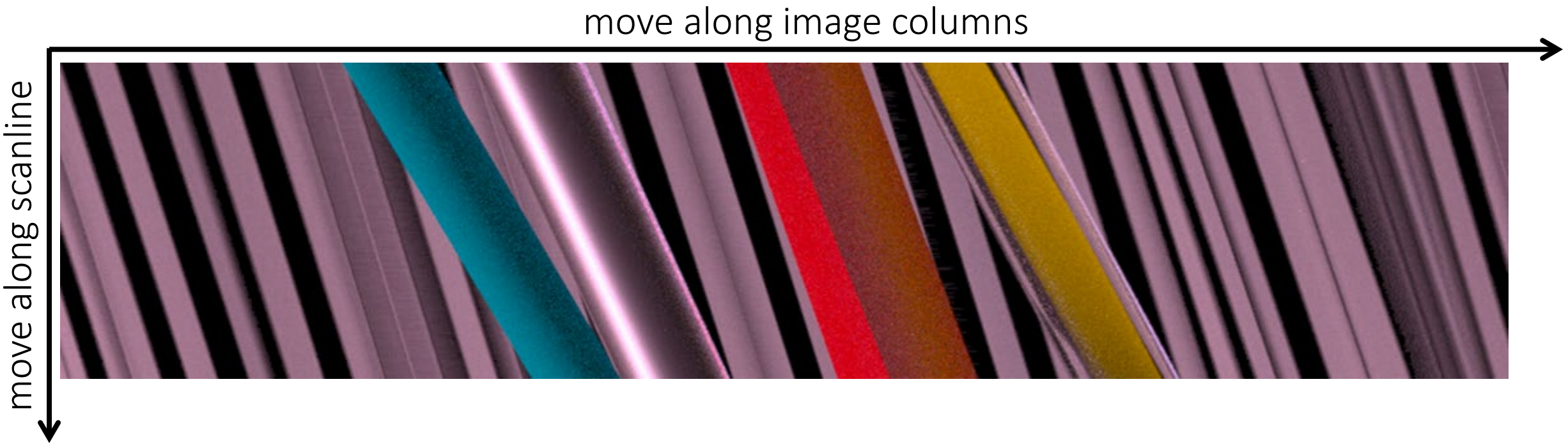
Epipolar plane images (EPIs)



Take the same row out of all images in a scanline, and stack these rows in a new 2D image.

Epipolar plane images (EPIs)

Why do we see straight lines?



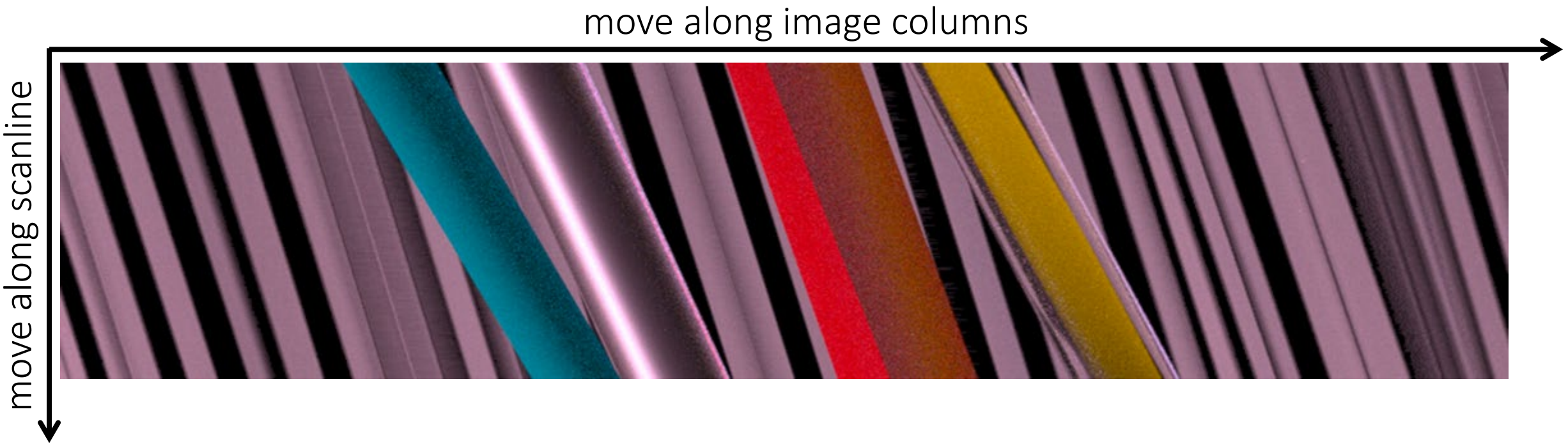
Take the same row out of all images in a scanline, and stack these rows in a new 2D image.

Epipolar plane images (EPIs)

Why do we see straight lines?

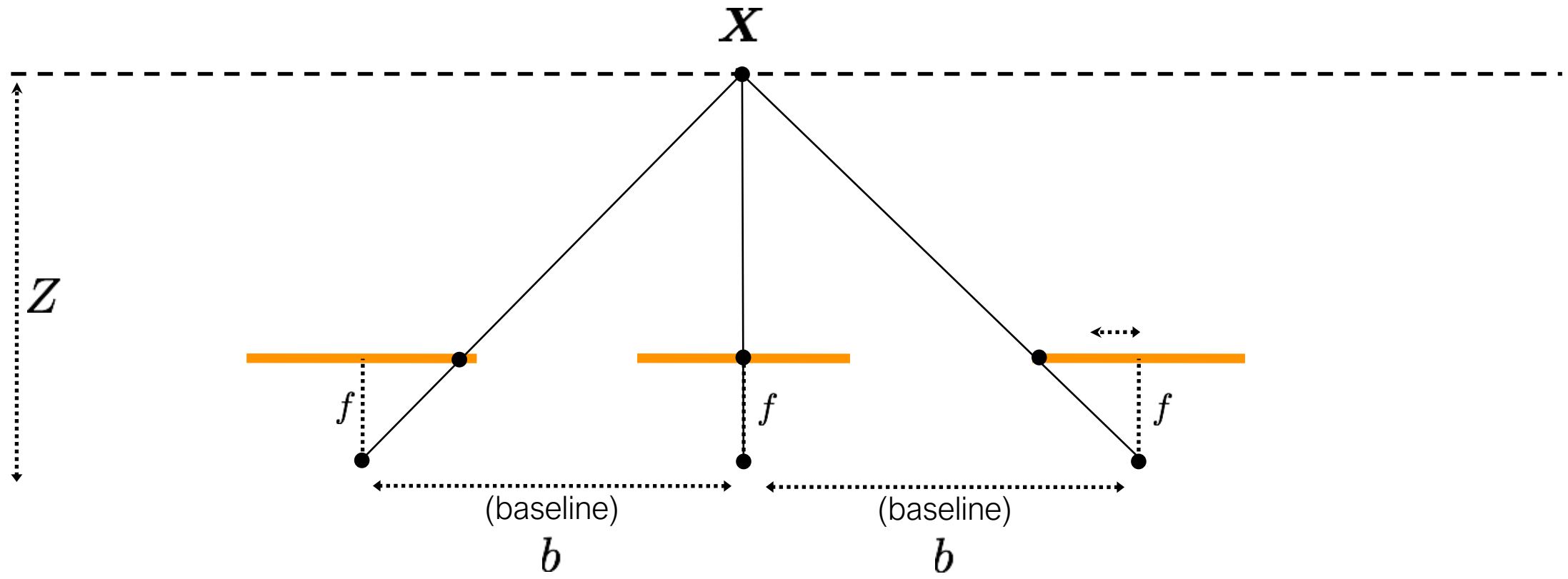
- Same 3D point changes location as viewpoint changes (i.e., *disparity*).

What does the slope of each line correspond to?



Take the same row out of all images in a scanline, and stack these rows in a new 2D image.

Stereo view of a lightfield camera



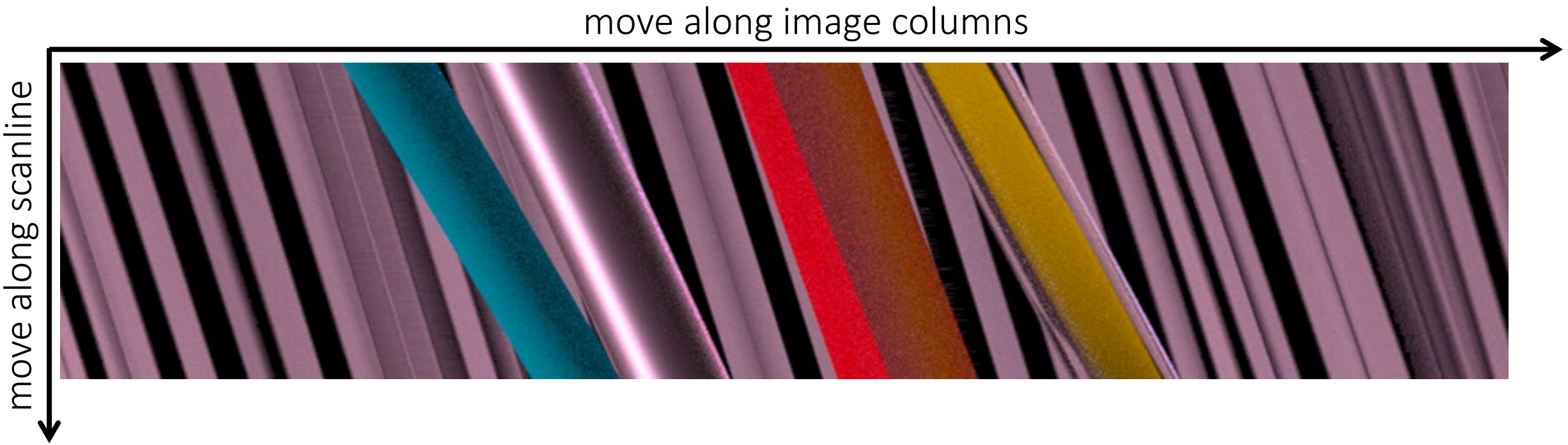
Disparity relationship:

$$d = x - x' = \frac{bf}{Z}$$

- Changing baseline b corresponds to moving along the scanline.
- Projections x of **X** are on a line of slope inversely proportional to depth.

Epipolar plane images (EPIs)

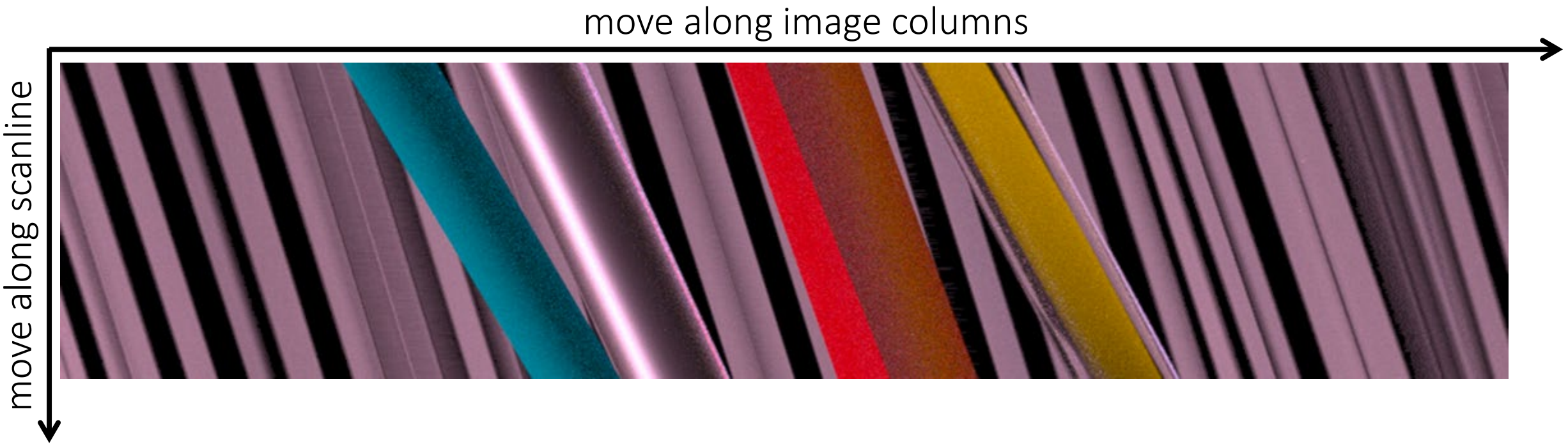
Per-pixel depth detection through line fitting and slope estimation.



Take the same row out of all images in a scanline, and stack these rows in a new 2D image.

Epipolar plane images (EPIs)

Per-pixel depth detection through line fitting and slope estimation.



Take the same row out of all images in a scanline, and stack these rows in a new 2D image.

Epipolar plane images (EPIs)

Scene Reconstruction from High Spatio-Angular Resolution Light Fields

Changil Kim[1,2] Henning Zimmer[1,2] Yael Pritch[1] Alexander Sorkine-Hornung[1] Markus Gross[1,2]

[1]Disney Research Zurich [2]ETH Zurich

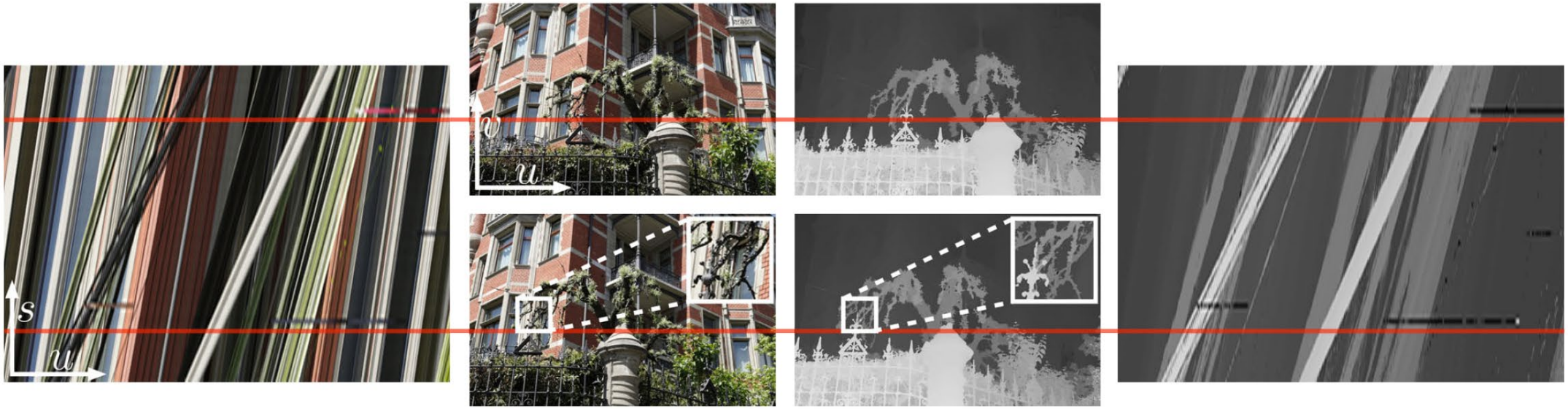
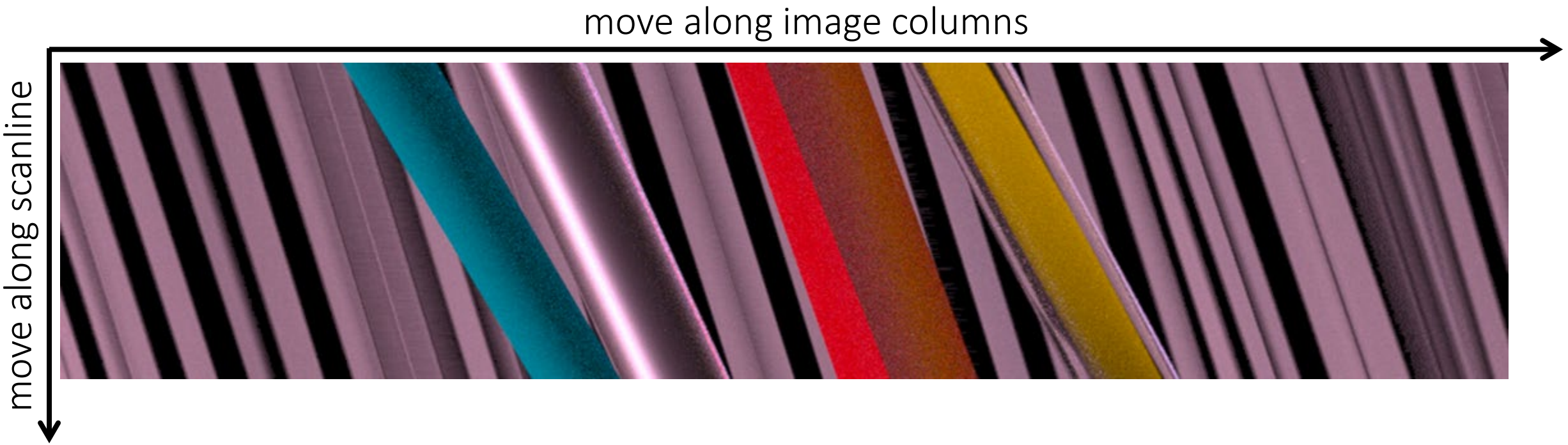


Figure 1: *Our method reconstructs accurate depth from light fields of complex scenes. The images on the left show a 2D slice of a 3D input light field, a so called epipolar-plane image (EPI), and two out of one hundred 21 megapixel images that were used to construct the light field. Our method computes 3D depth information for all visible scene points, illustrated by the depth EPI on the right. From this representation, individual depth maps or segmentation masks for any of the input views can be extracted as well as other representations like 3D point clouds. The horizontal red lines connect corresponding scanlines in the images with their respective position in the EPI.*

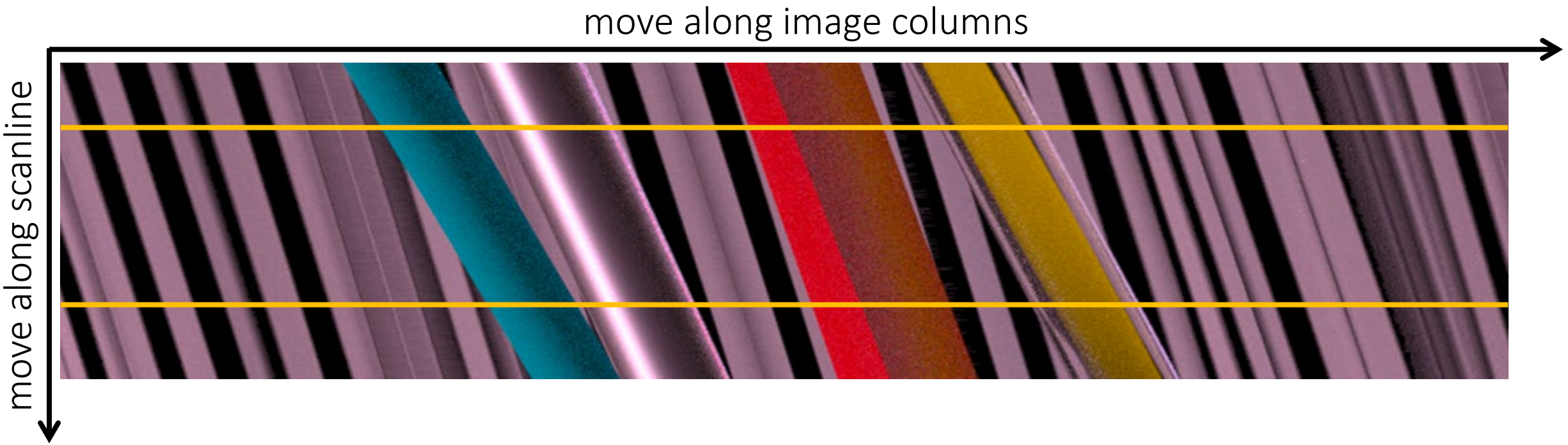
Aside: different types of cameras

What part of the EPI is captured when we use a stereo pair of cameras?



Aside: different types of cameras

What part of the EPI is captured when we use a stereo pair of cameras?

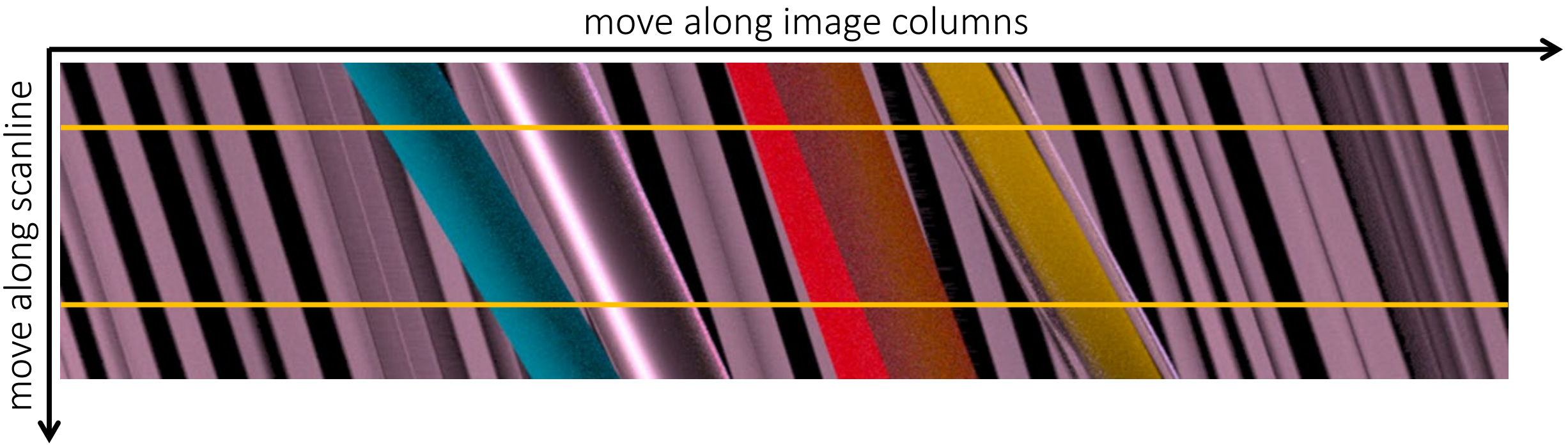


Aside: different types of cameras

What part of the EPI is captured when we use a stereo pair of cameras?

- Two horizontal lines.

When are these two views sufficient to infer depth?



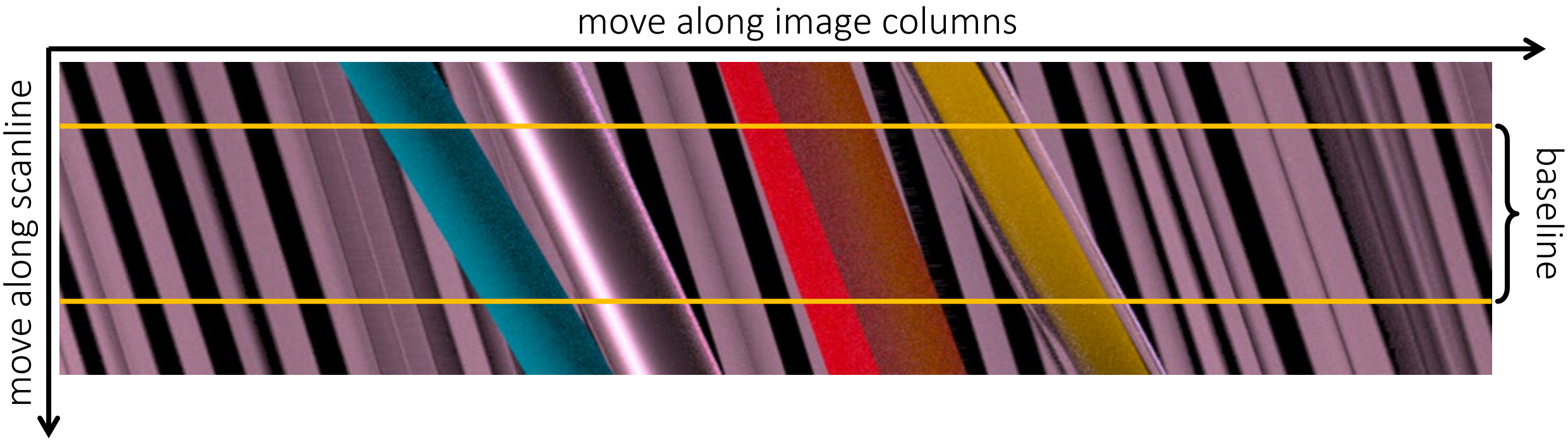
Aside: different types of cameras

What part of the EPI is captured when we use a stereo pair of cameras?

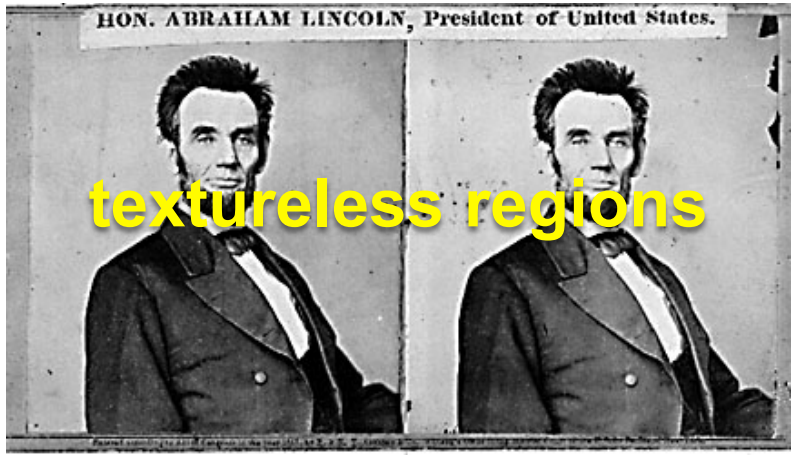
- Two horizontal lines.

When are these two views sufficient to infer depth?

- When their *baseline* is large enough to infer the slope of the lines in EPIs.



When are correspondences difficult?









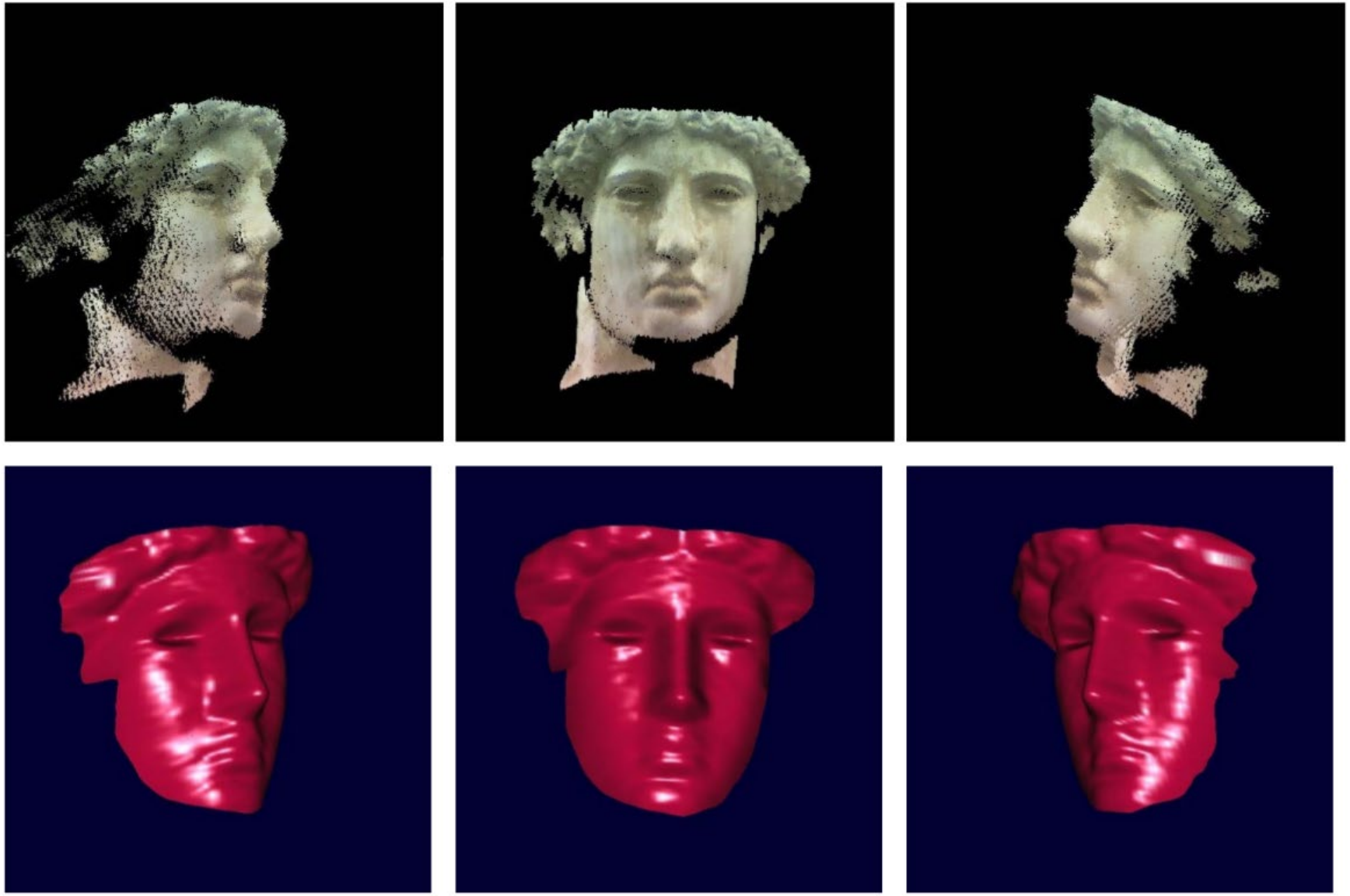
Structured light

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

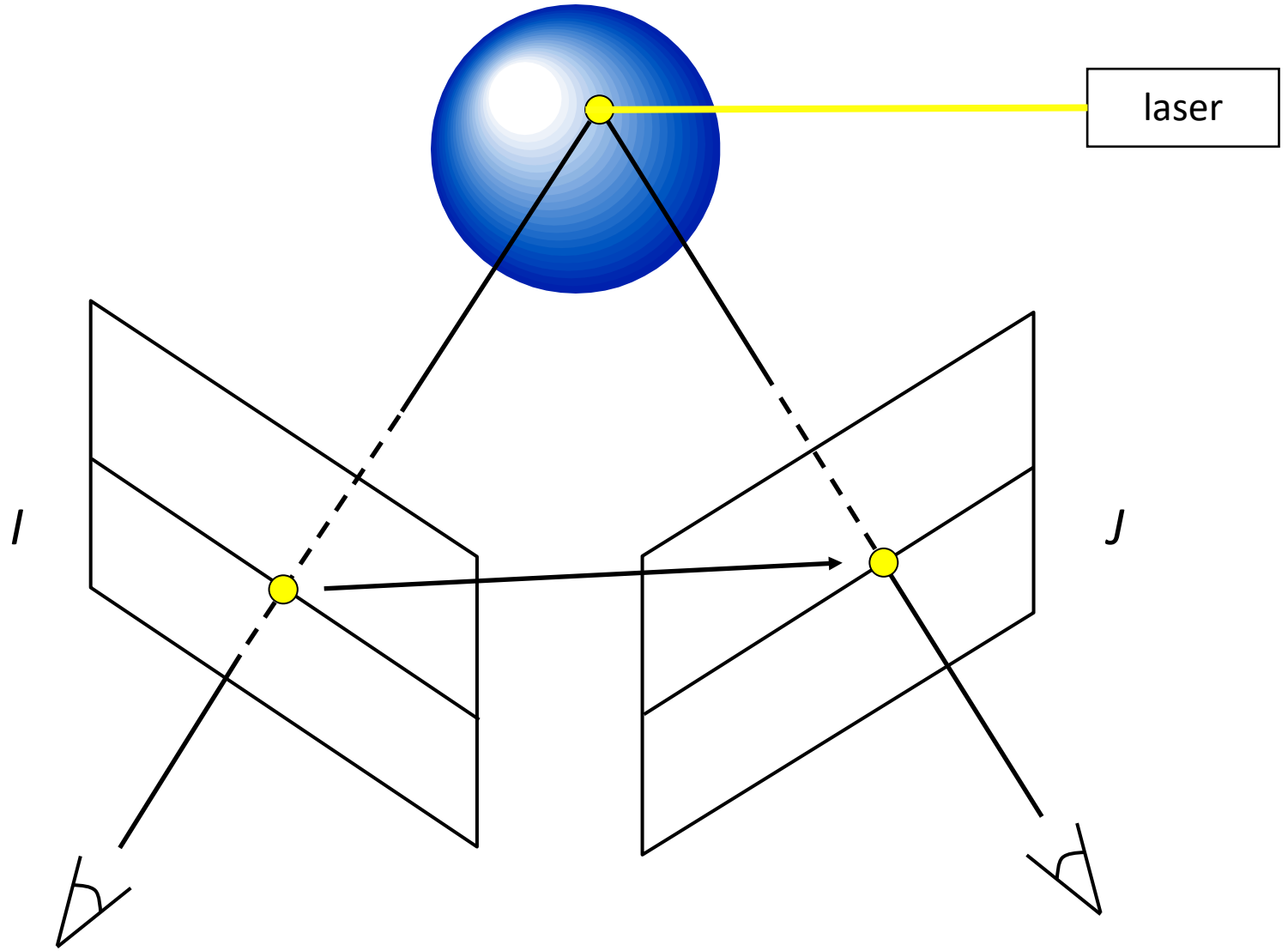
Disparity between laser points on the same scanline in the images determines the 3-D coordinates of the laser point on object



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

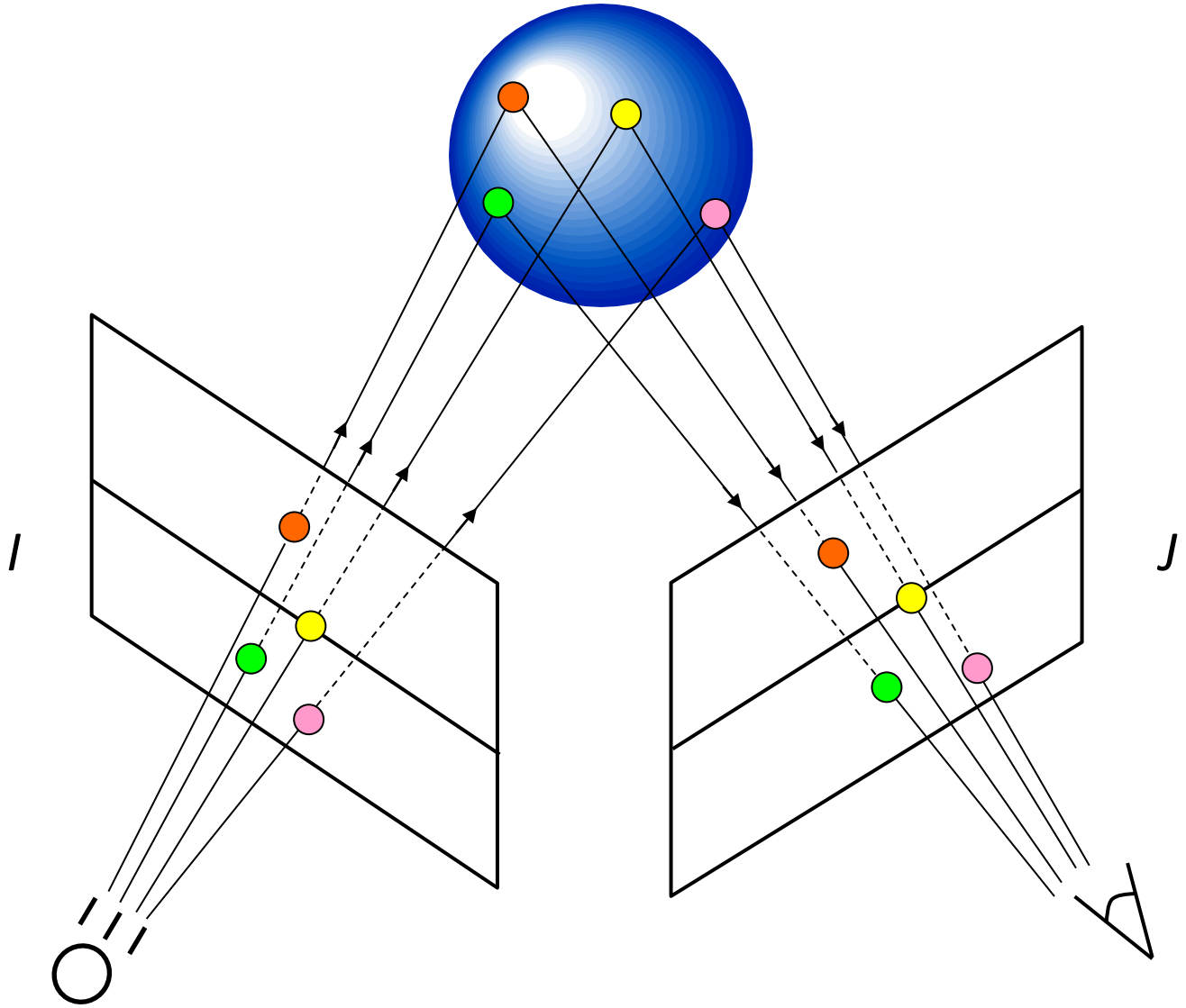


Structured light and two cameras



Structured light and one camera

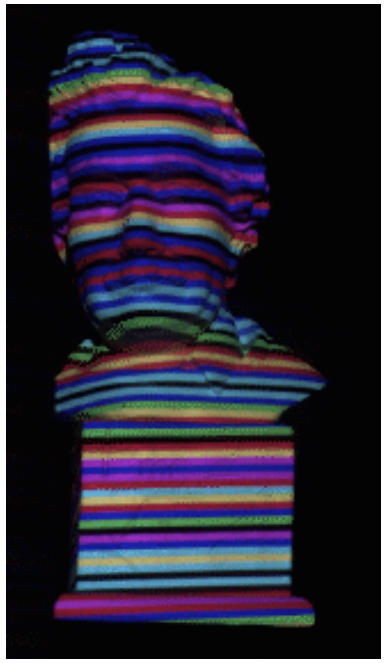
Projector acts like
“reverse” camera



Structured Light





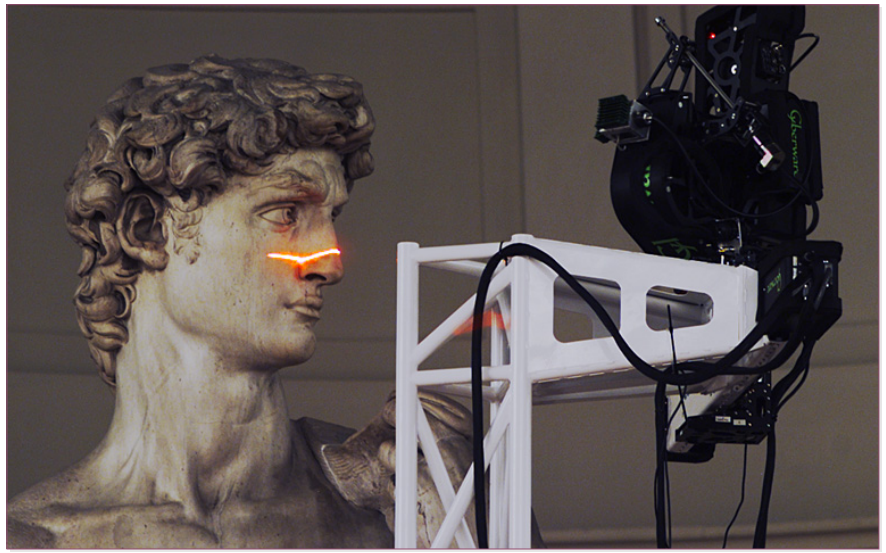


- Any spatio-temporal pattern of light projected on a surface (or volume).
- Cleverly illuminate the scene to extract scene properties (eg., 3D).
- Avoids problems of 3D estimation in scenes with complex texture/BRDFs.
- Very popular in vision and successful in industrial applications (parts assembly, inspection, etc).

3D Scanning using structured light



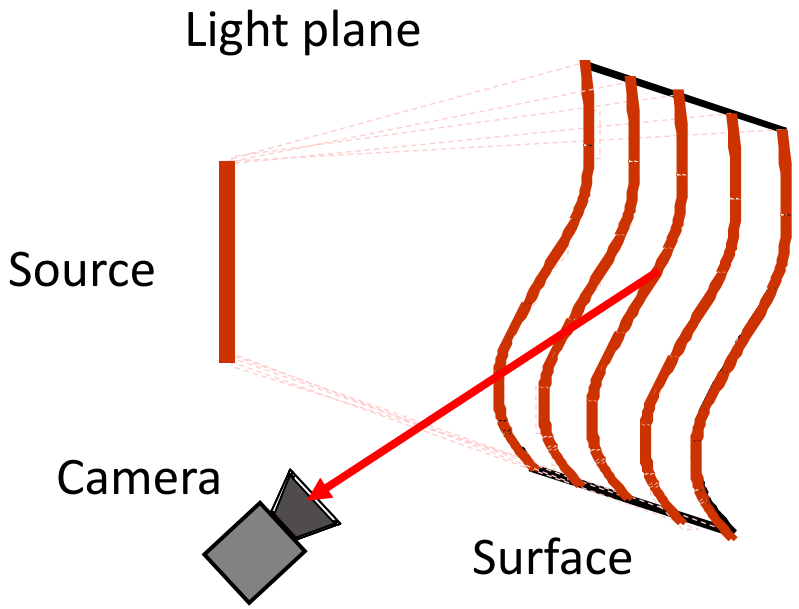


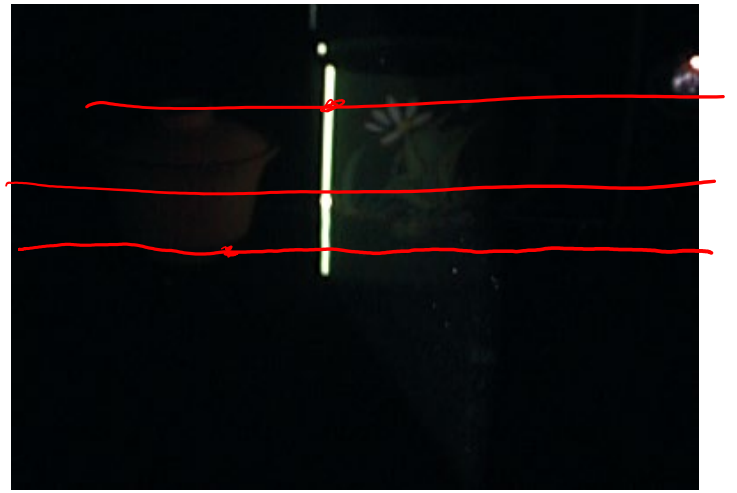




Do we need to illuminate the scene point by point?

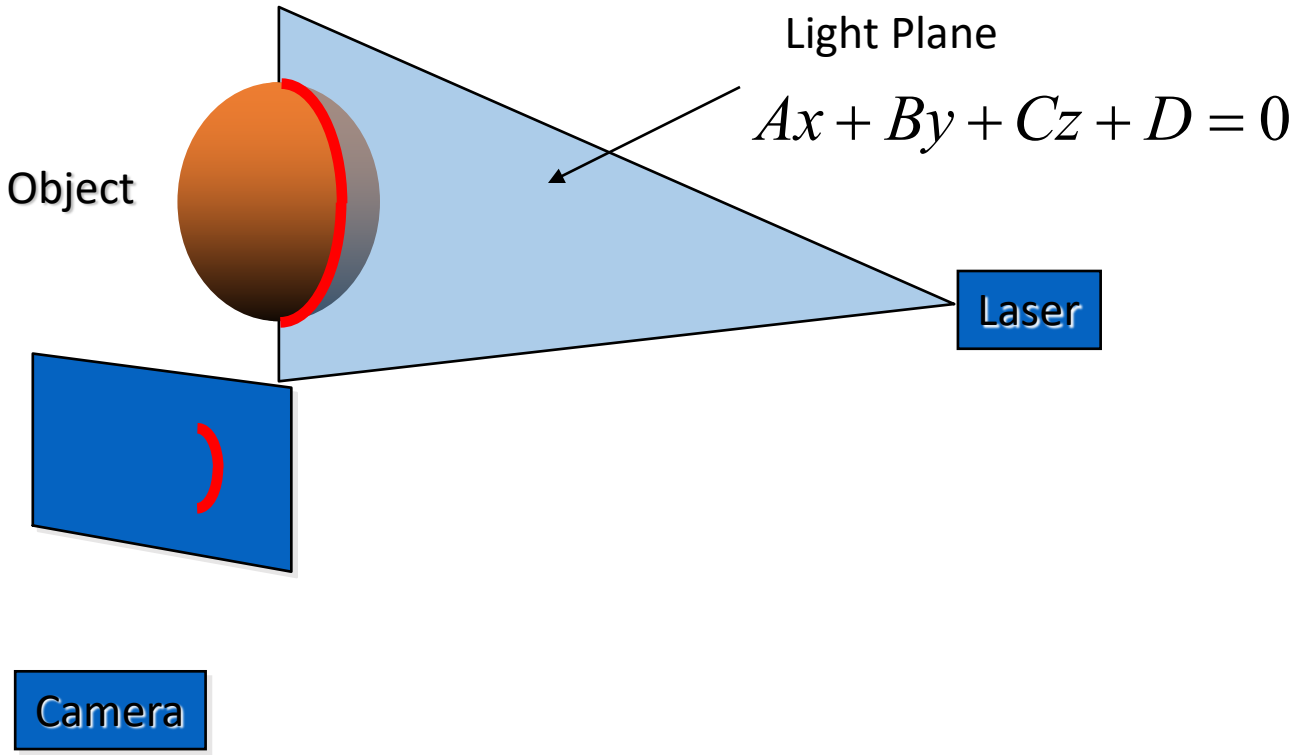
Light Stripe Scanning – Single Stripe





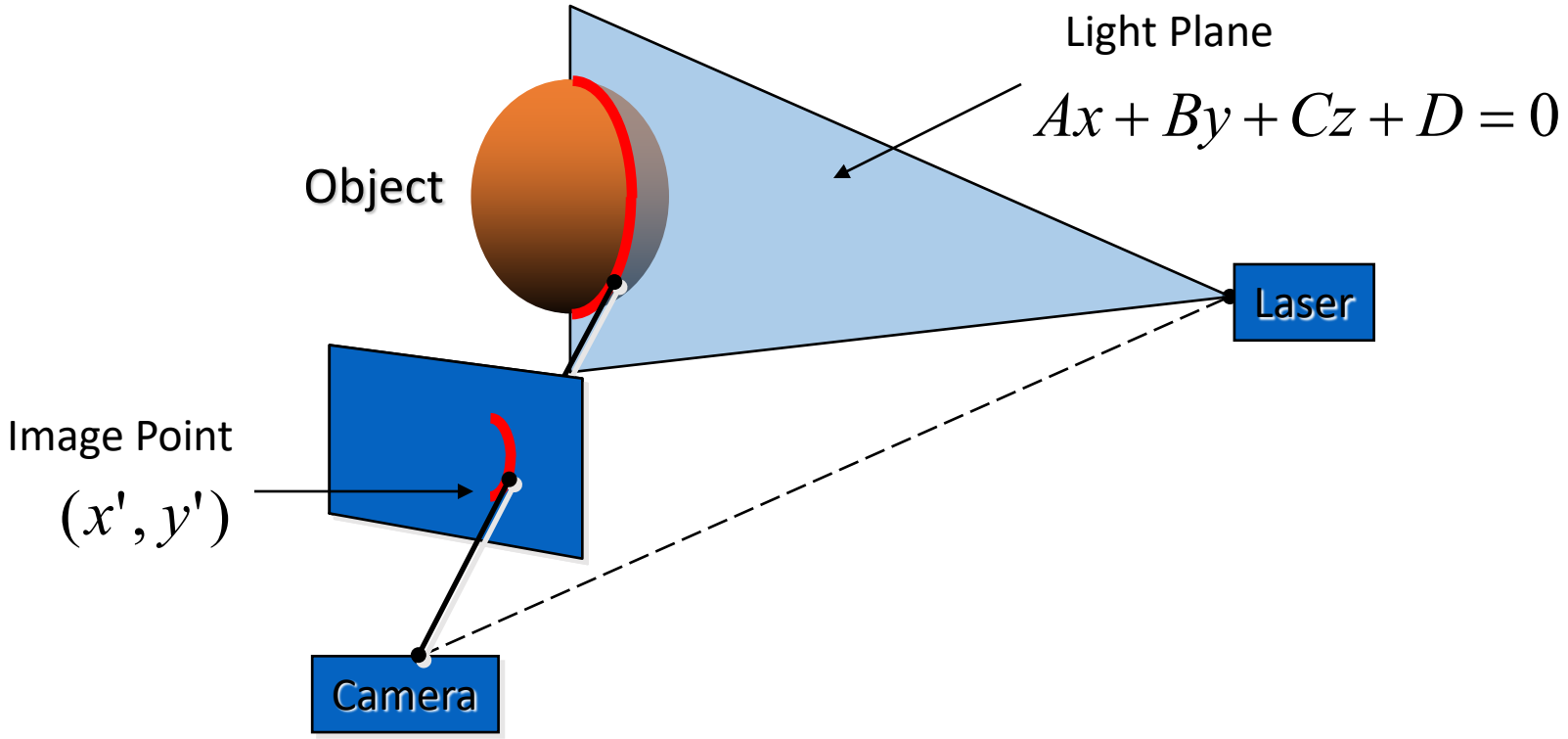
- Faster optical triangulation:
  - Project a single stripe of laser light
  - Scan it across the surface of the object
  - This is a very precise version of structured light scanning
  - Good for high resolution 3D, but still needs many images and takes time

Triangulation



- Project laser stripe onto object

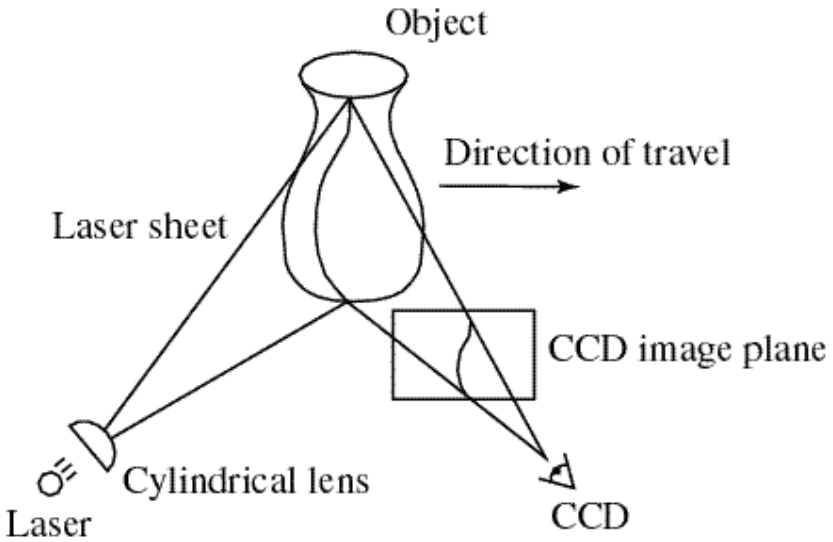
Triangulation

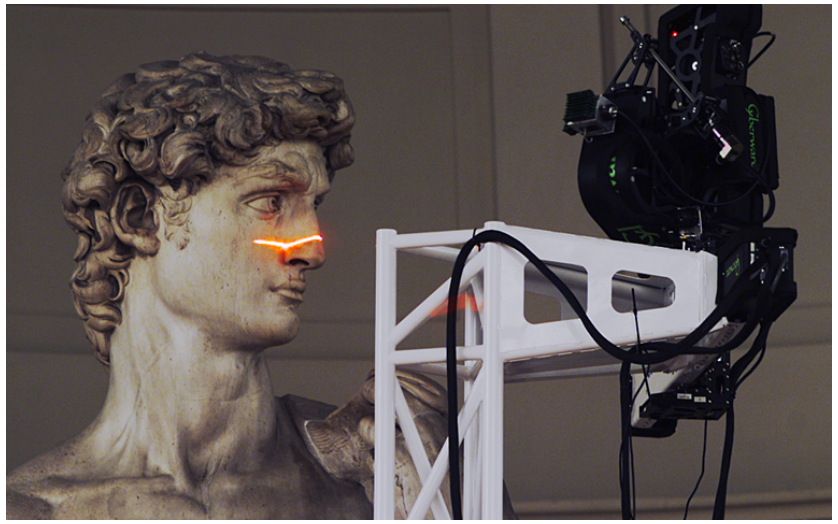


- Depth from ray-plane triangulation:
  - Intersect camera ray with light plane

$$x = x'z / f \qquad y = y'z / f \qquad z = \frac{-Df}{Ax'+By'+Cf}$$

Example: Laser scanner





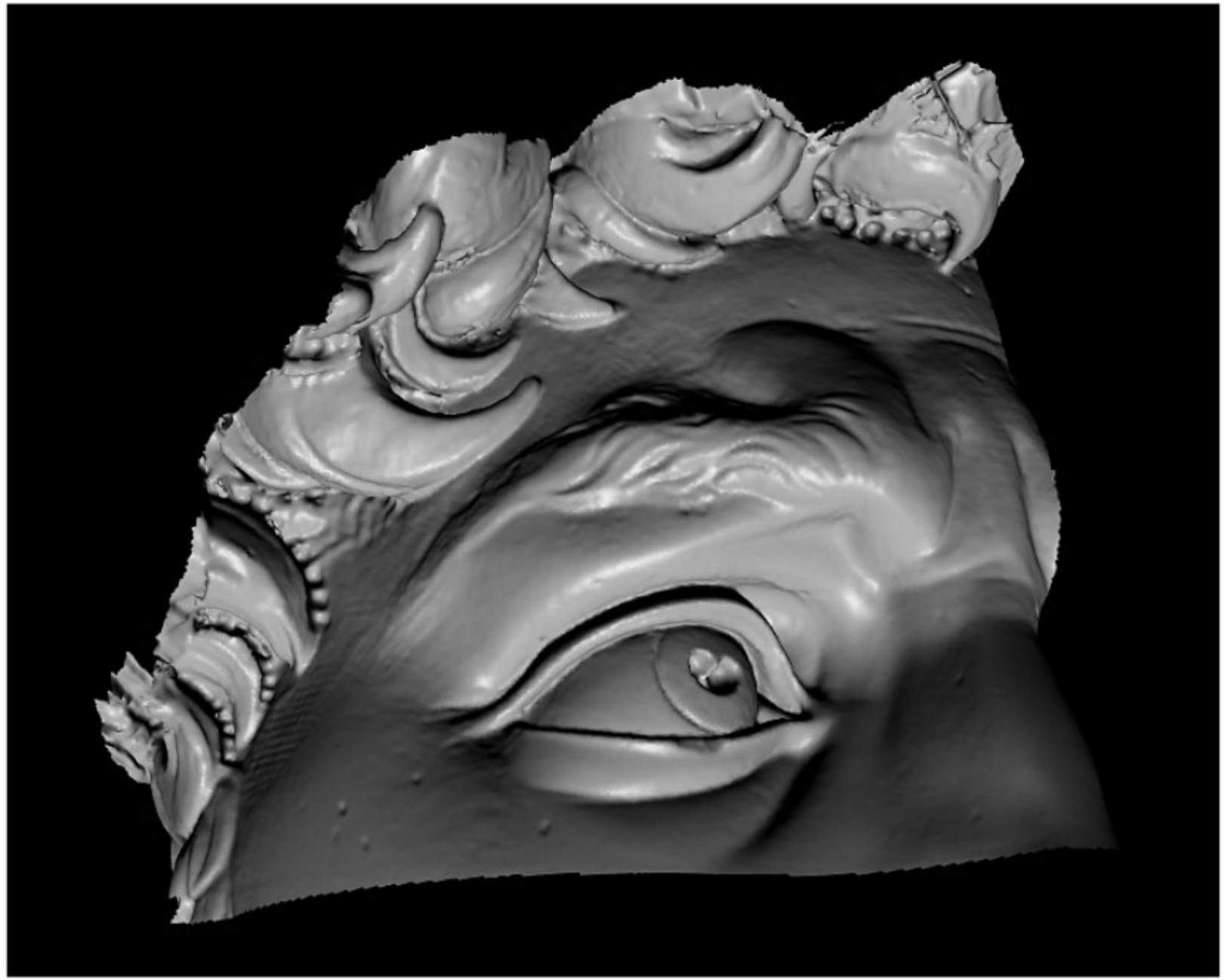
Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/



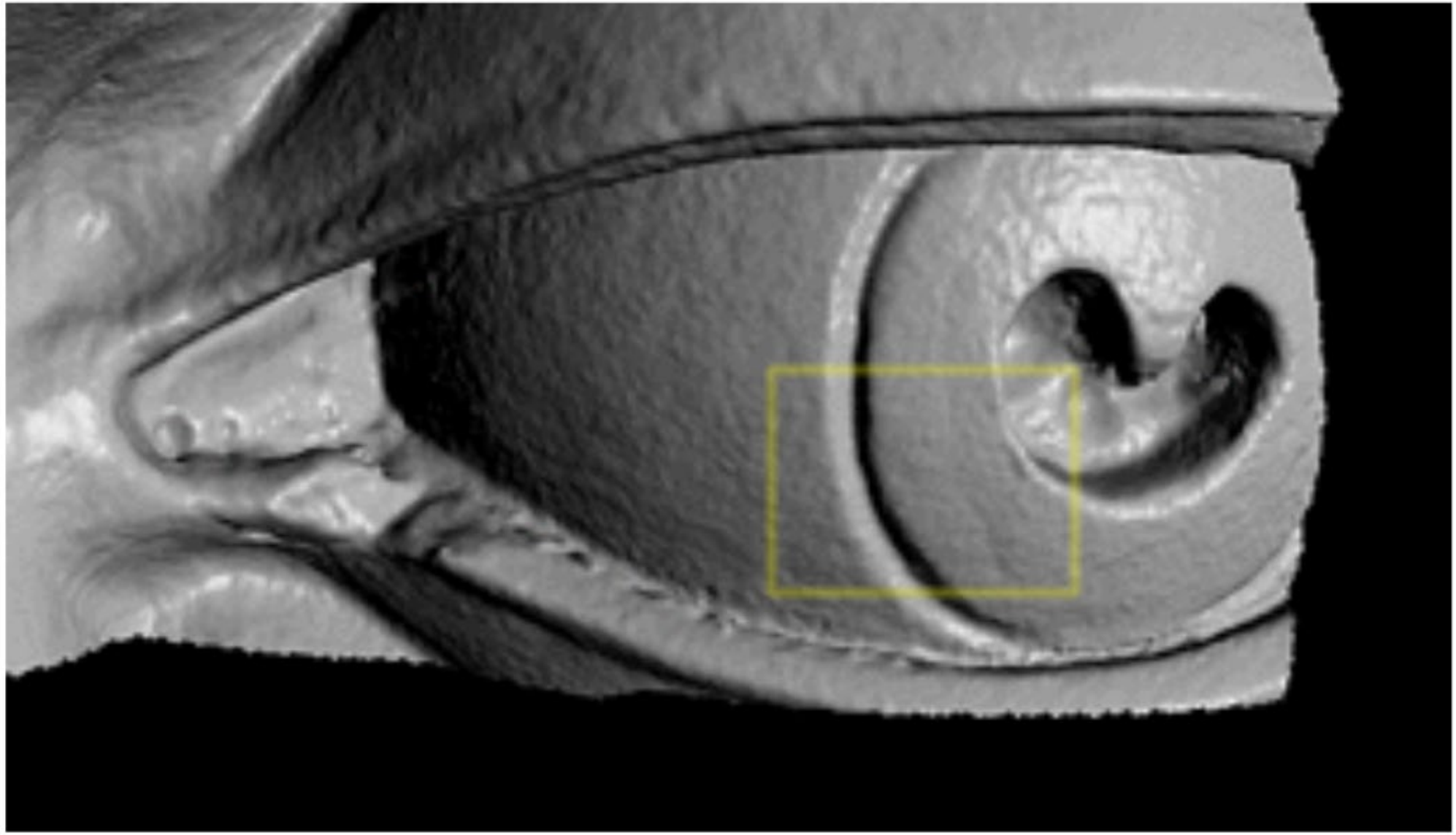
The Digital Michelangelo Project, Levoy et al.



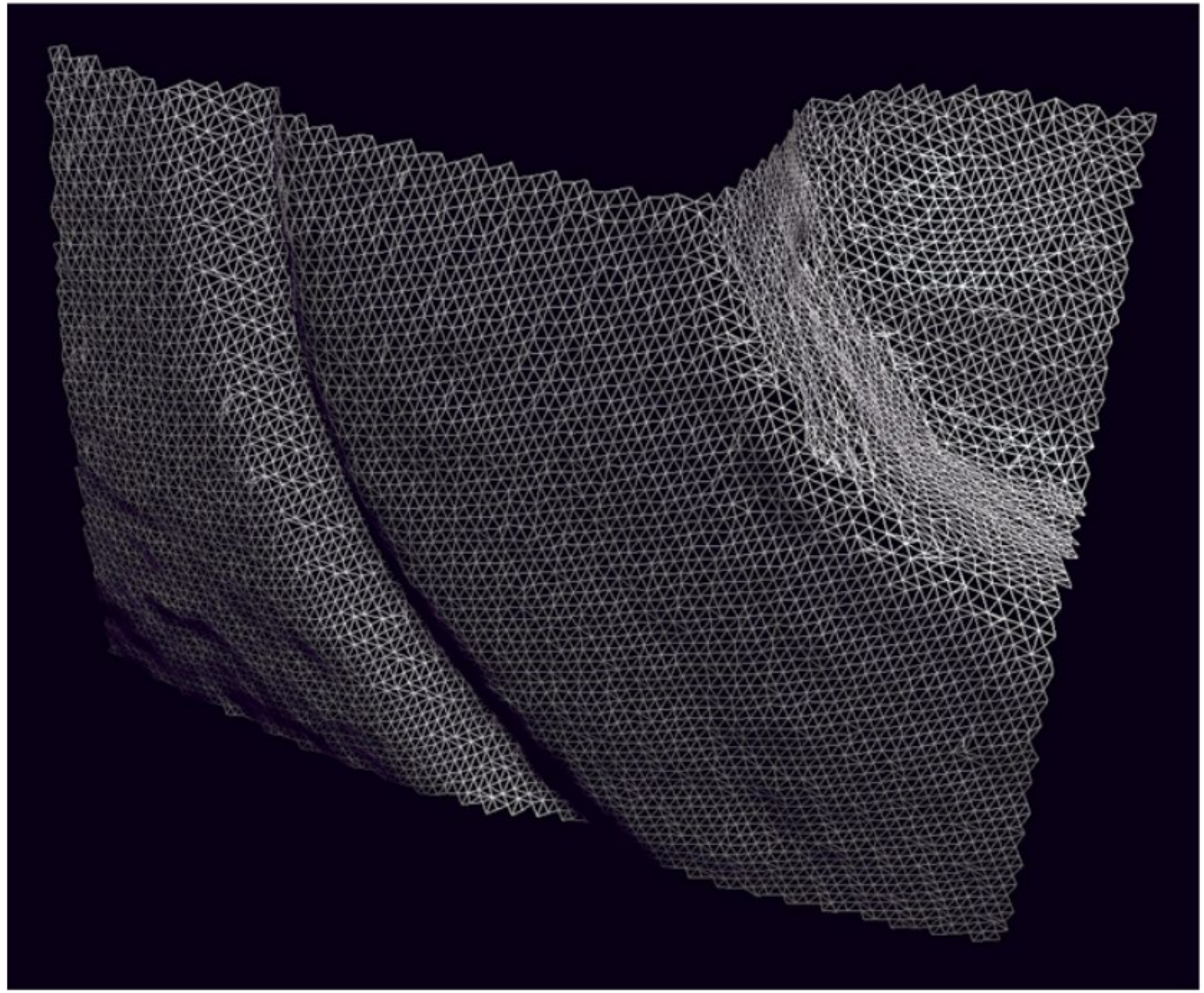
The Digital Michelangelo Project, Levoy et al.



The Digital Michelangelo Project, Levoy et al.



The Digital Michelangelo Project, Levoy et al.



The Digital Michelangelo Project, Levoy et al.

Portable 3D laser scanner (this one by Minolta)







Faster Acquisition?

Binary coding

Faster Acquisition?

- Project multiple stripes simultaneously
- What is the problem with this?

Faster Acquisition?

- Project multiple stripes simultaneously
- Correspondence problem: which stripe is which?

- Common types of patterns:

    - Binary coded light striping
    - Gray/color coded light striping

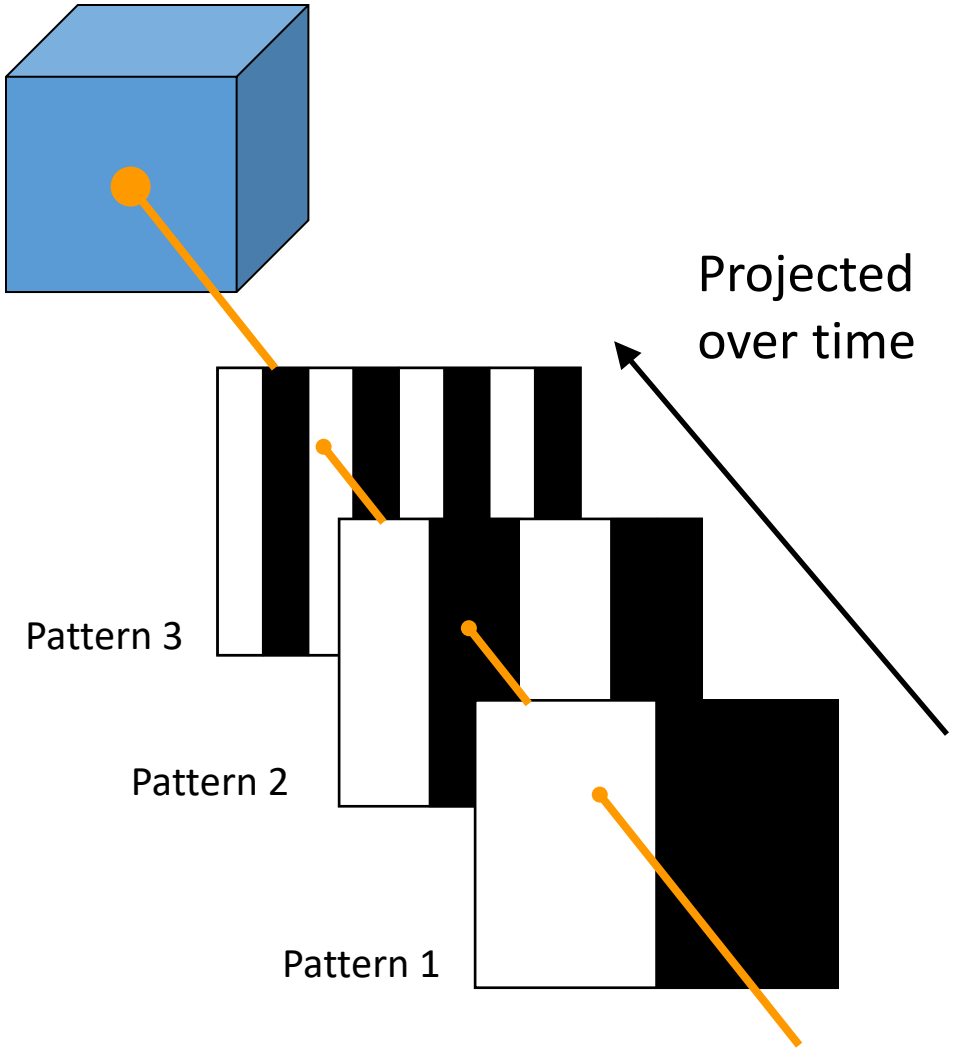
Binary Coding

Faster:

$2^n - 1$ stripes in n images.

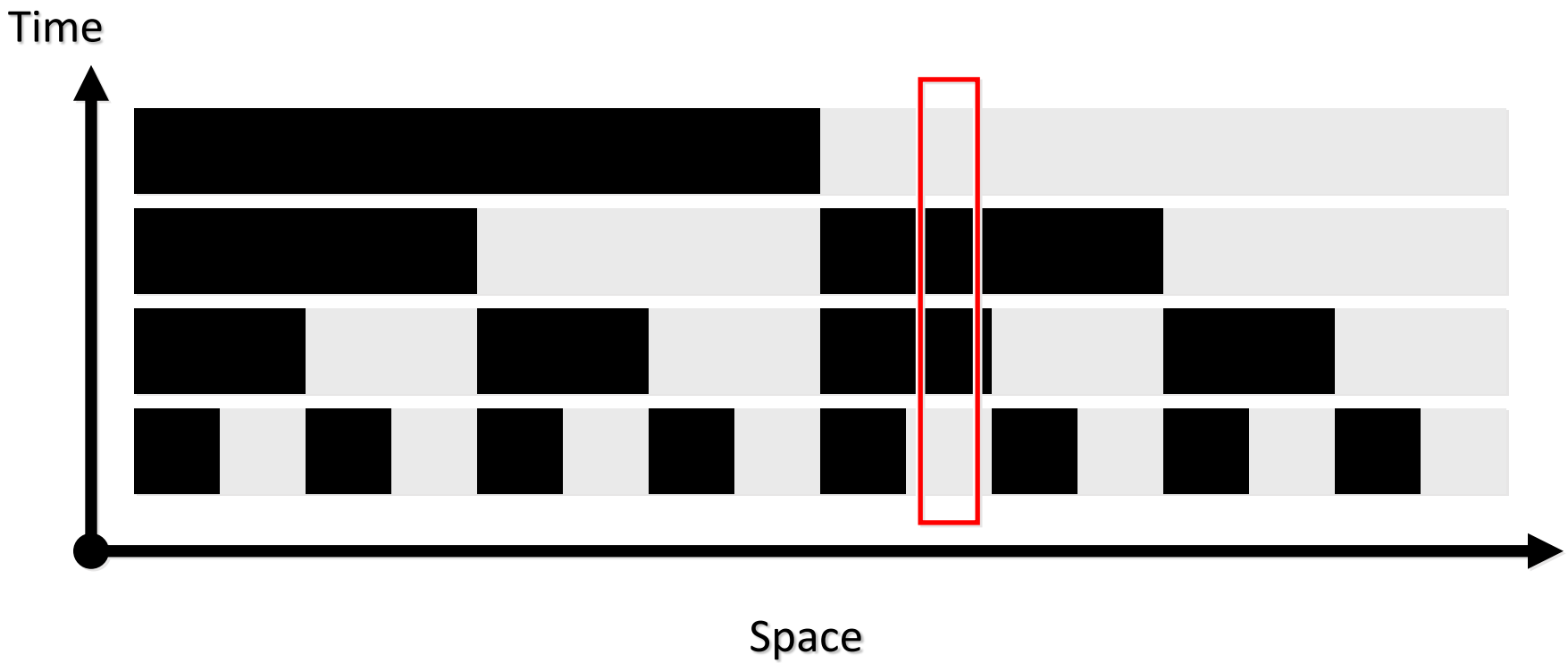
Example:

3 binary-encoded patterns which allows the measuring surface to be divided in 8 sub-regions

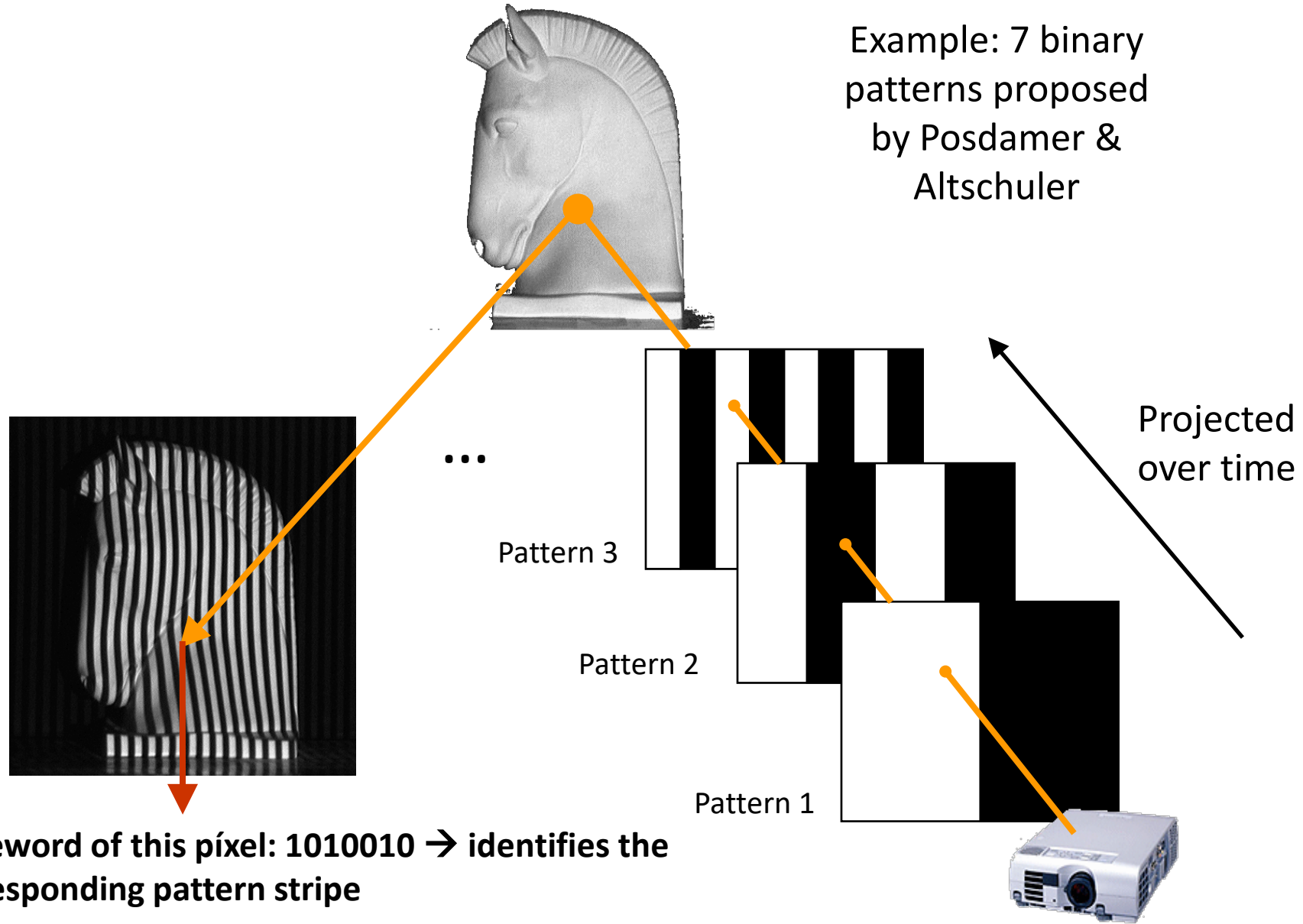


Binary Coding

- Assign each stripe a unique illumination code over time [Posdamer 82]

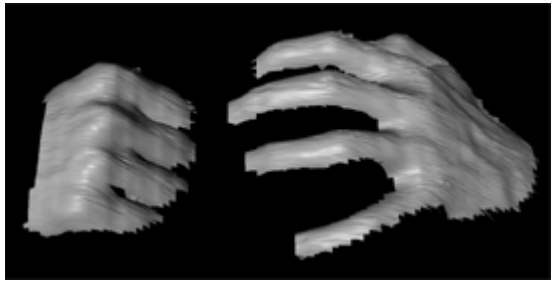


Binary Coding

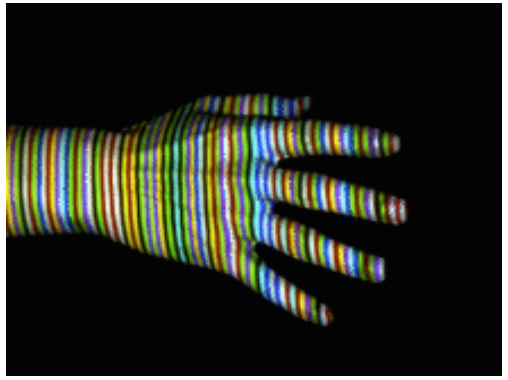
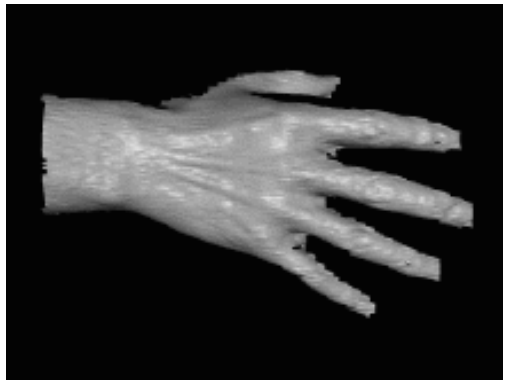


More complex patterns





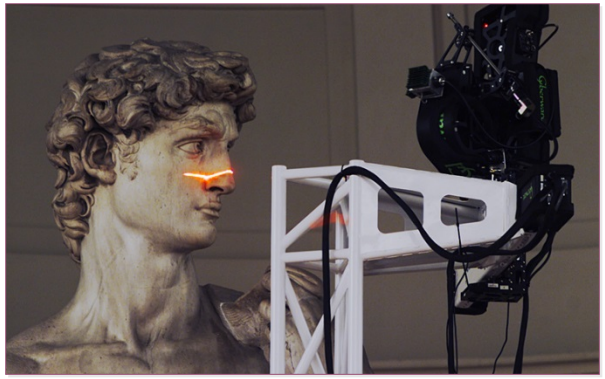
Works despite complex appearances

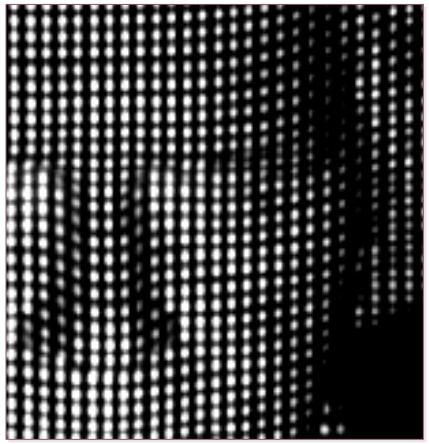



Works in real-time and on dynamic scenes

- Need very few images (one or two).
- But needs a more complex correspondence algorithm

Continuum of Triangulation Methods





Multi-stripe
Multi-frame

Single-stripe

Single-frame

$L(s,p)$

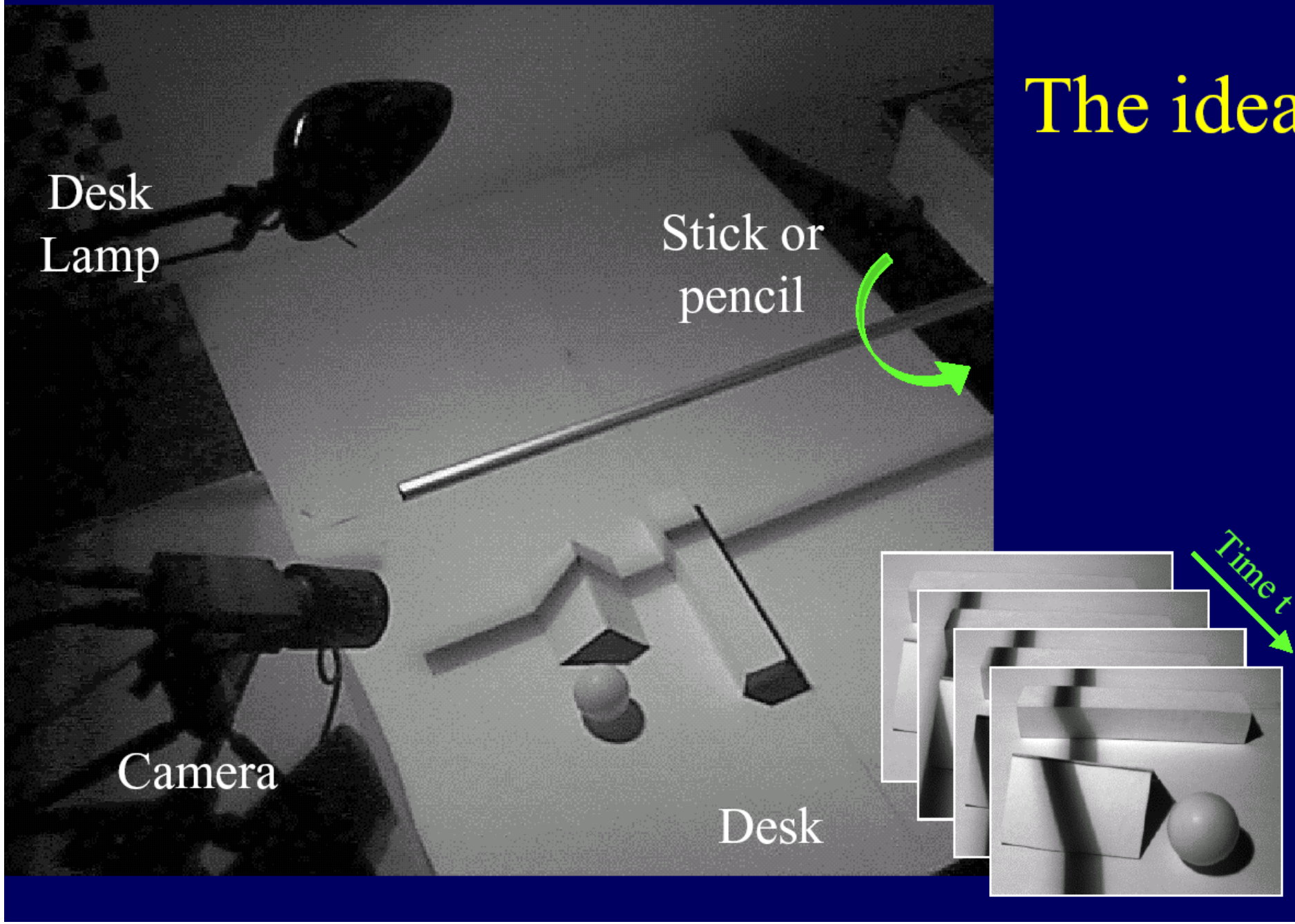
$\frac{L(s,p+\varepsilon) - L(s,p)}{\varepsilon}$

Slow, robust

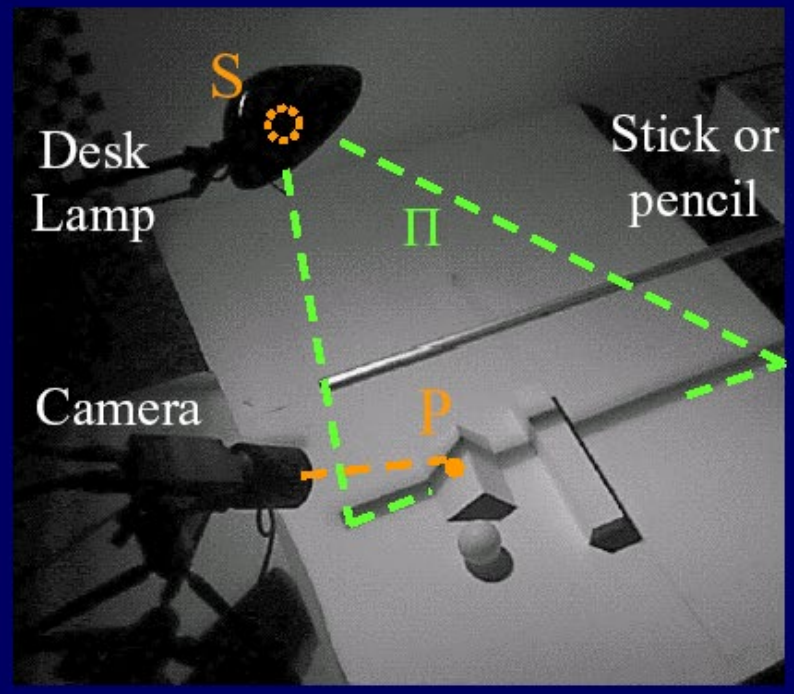
Fast, fragile

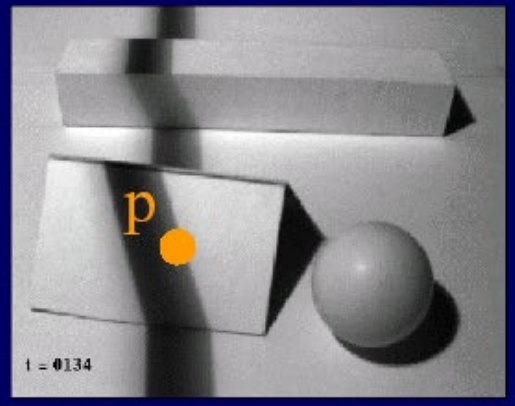
Using shadows

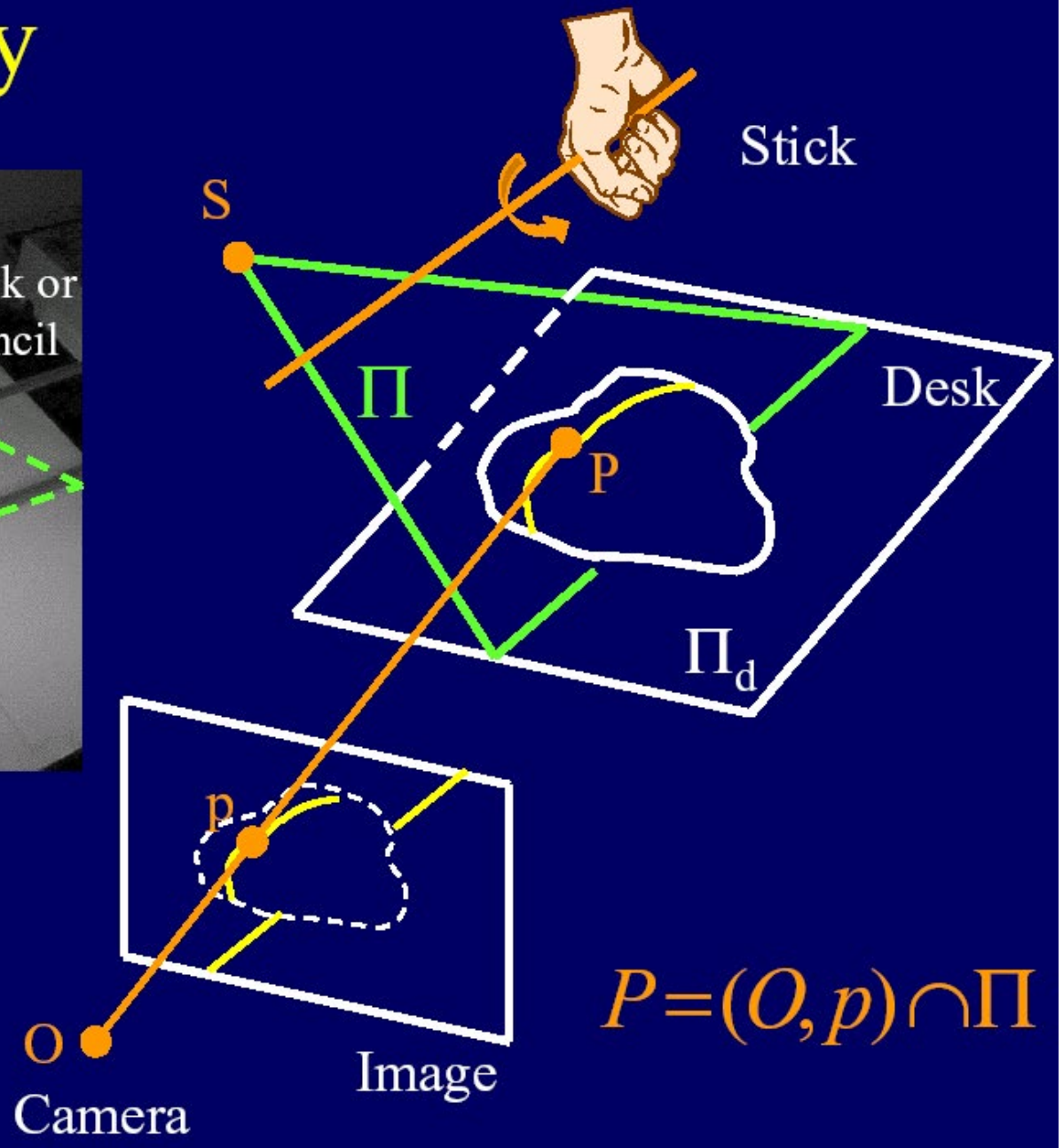
The idea



The geometry

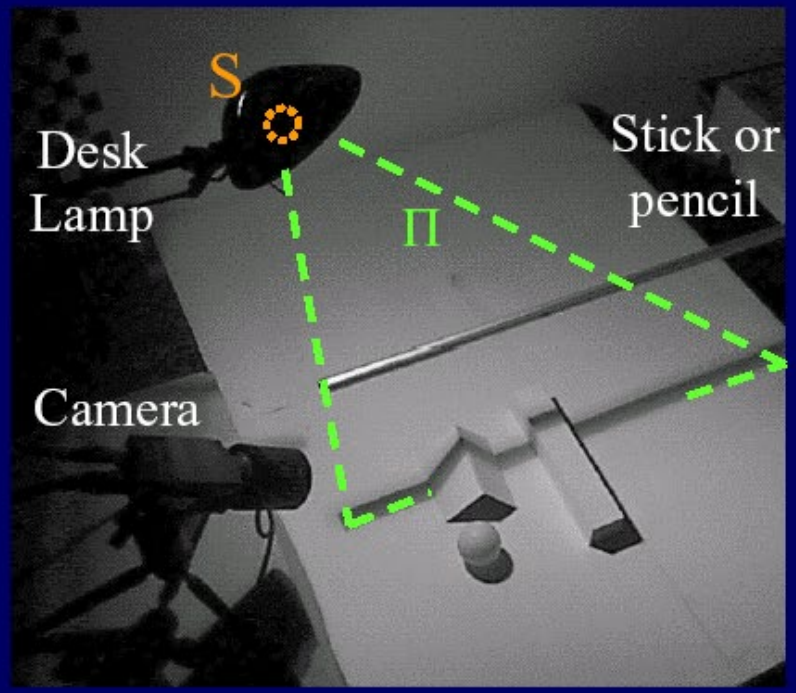


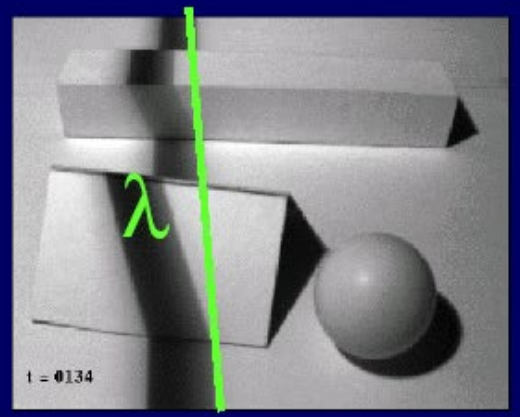


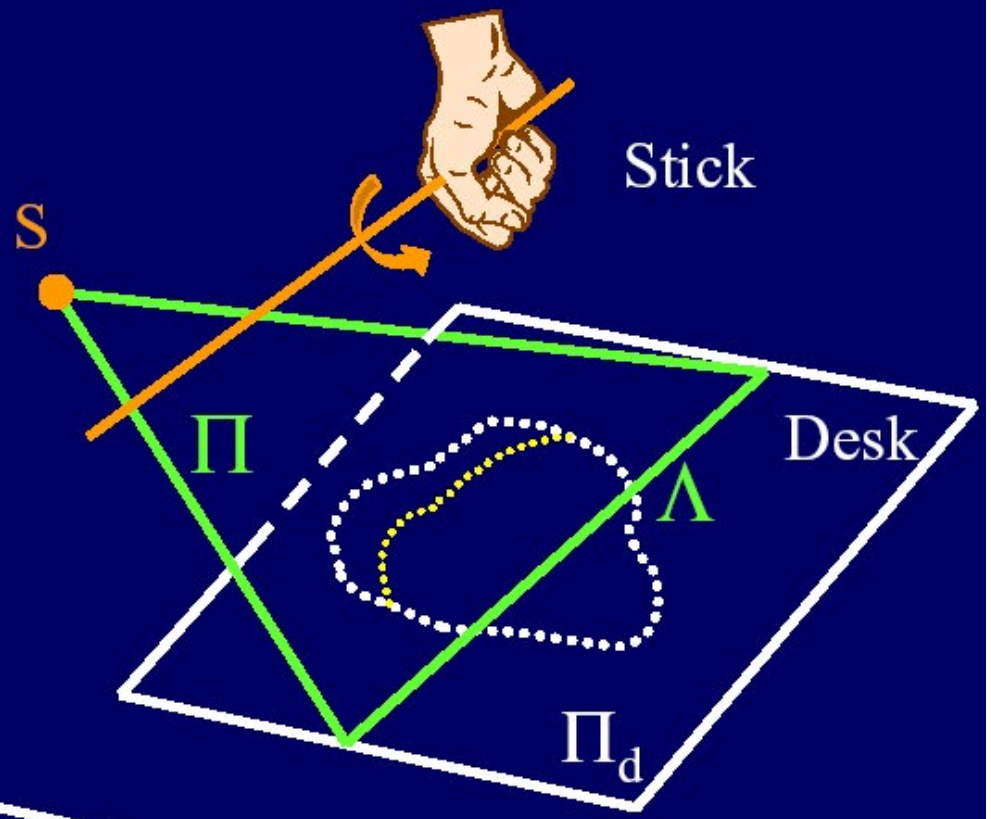


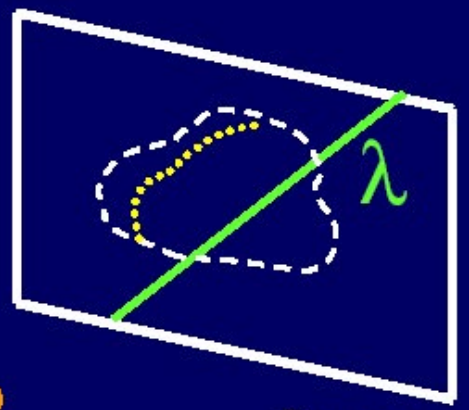
$$P=(O,p)\cap\Pi$$

The geometry







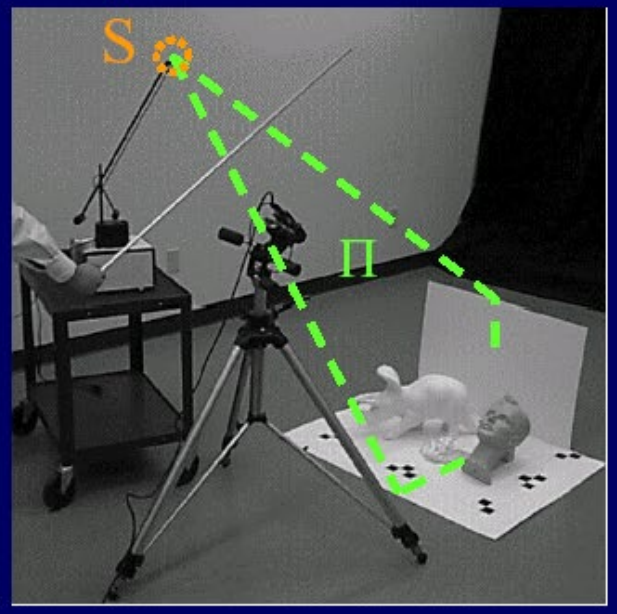


O
Camera

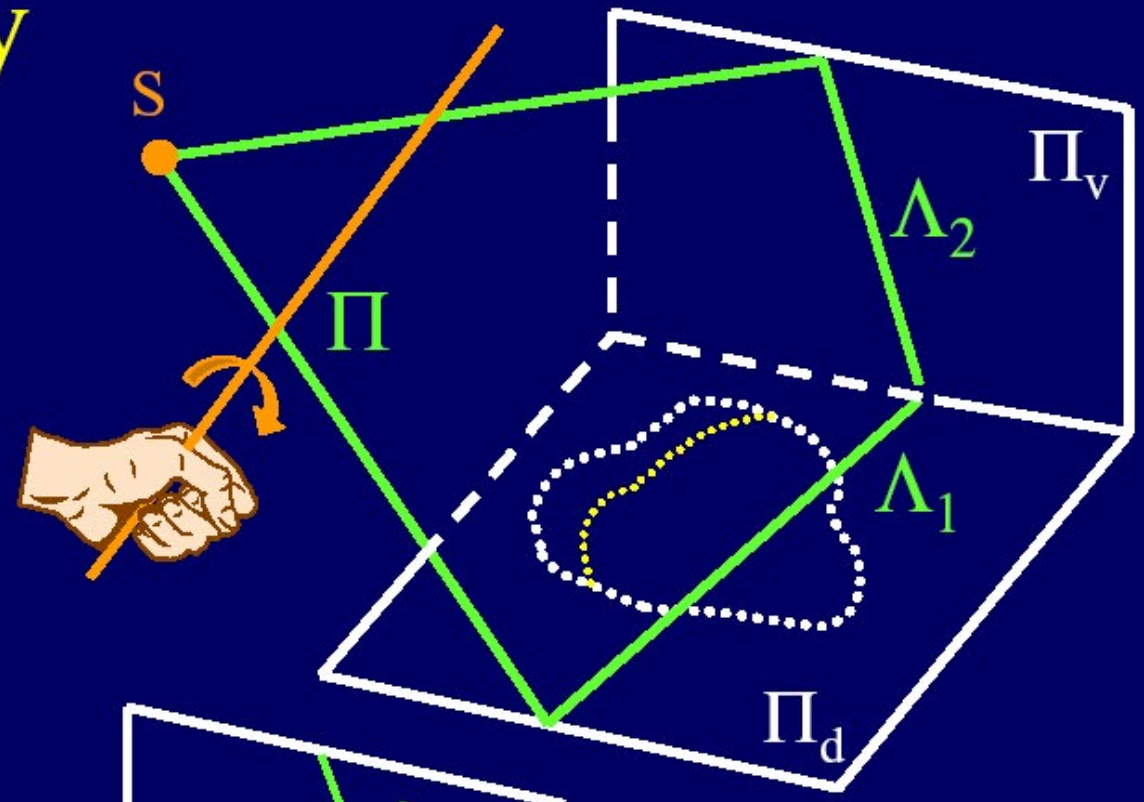
$\Lambda = (O, \lambda) \cap \Pi_d$

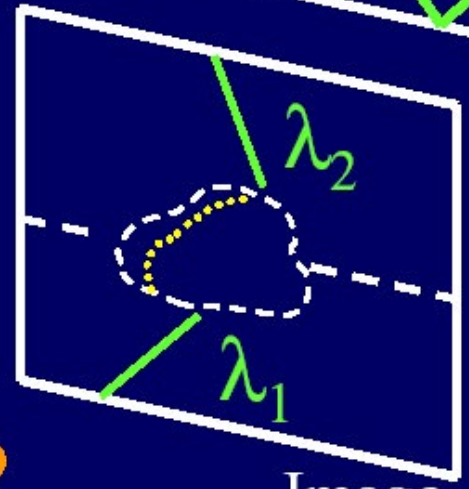
$\Pi = (S, \Lambda)$

The geometry









O
Camera

$\Lambda_1 = (O, \lambda_1) \cap \Pi_d$

$\Lambda_2 = (O, \lambda_2) \cap \Pi_v$

$\Pi = (\Lambda_1, \Lambda_2)$

Angel experiment







Accuracy: 0.1mm over 10cm → ~ 0.1% error

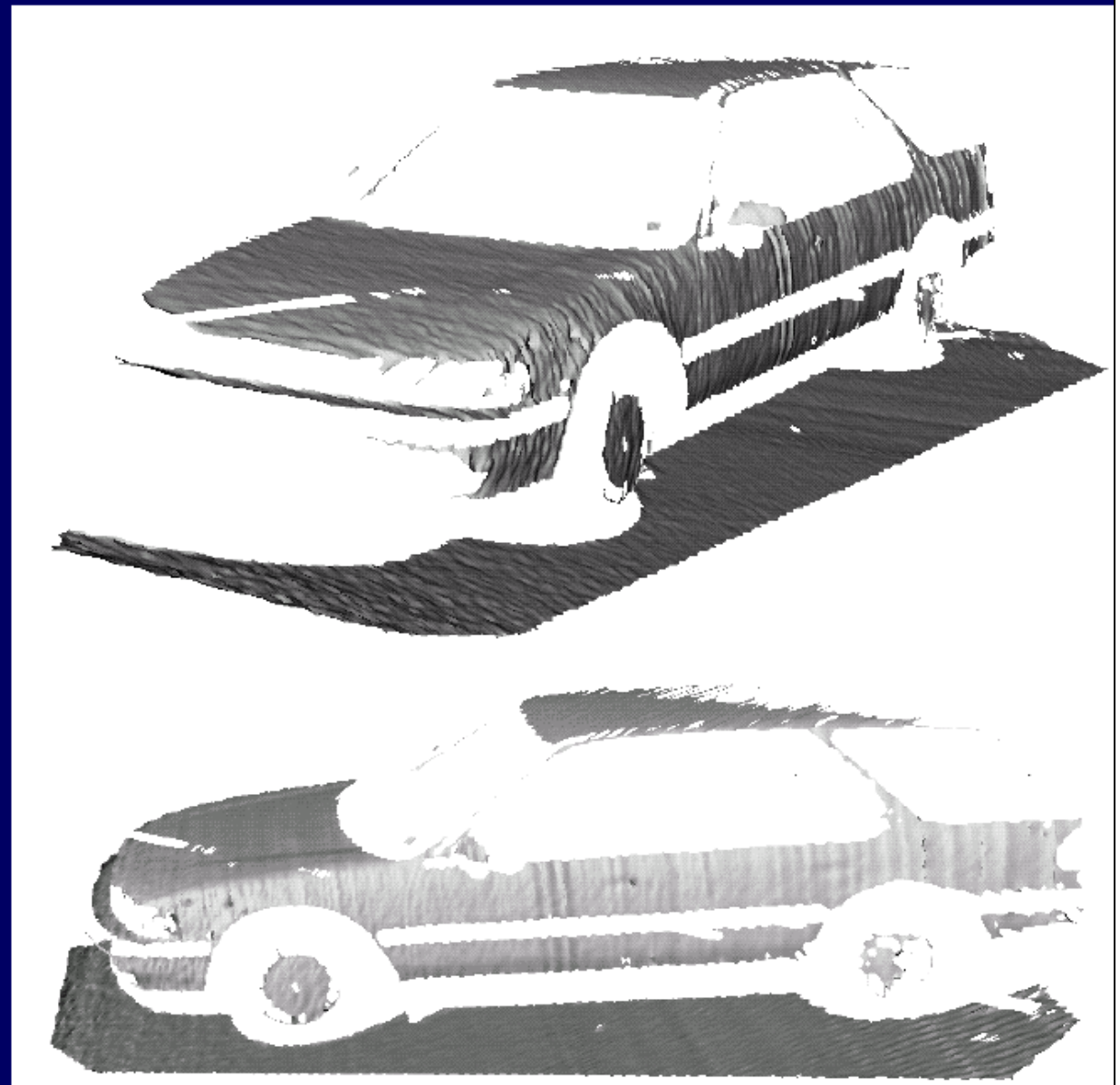
Scanning with the sun





Accuracy: 1cm over 2m

~ 0.5% error



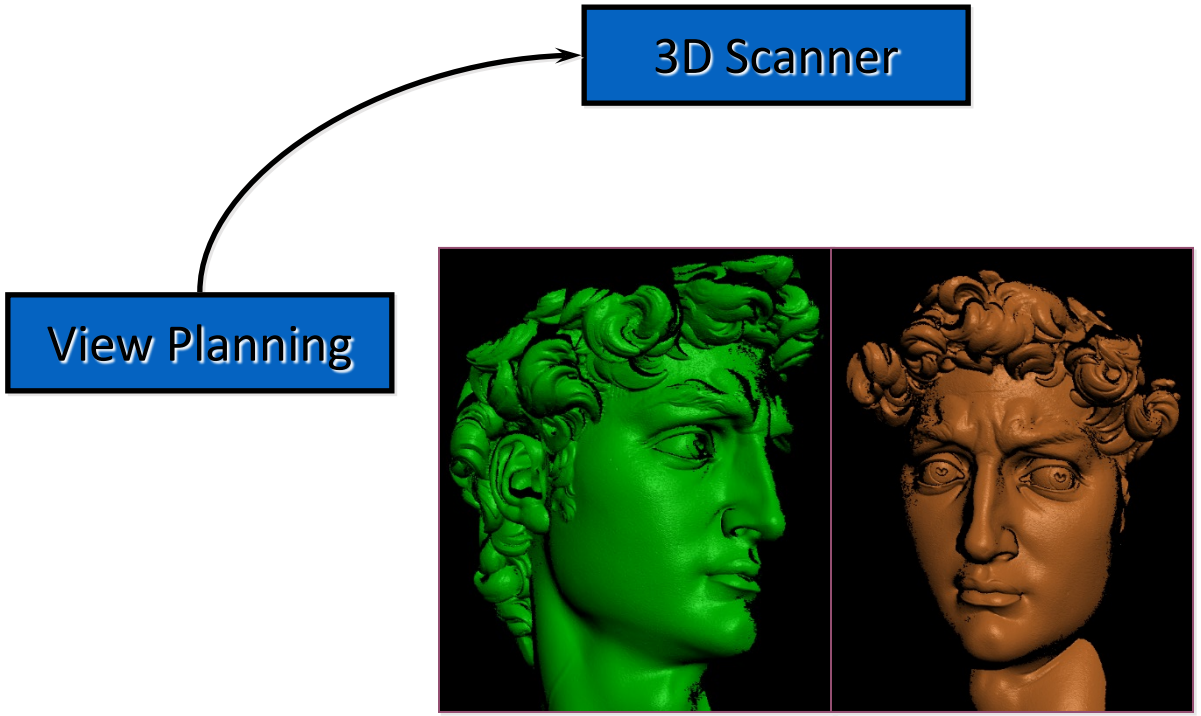
The 3D scanning pipeline

3D Model Acquisition Pipeline

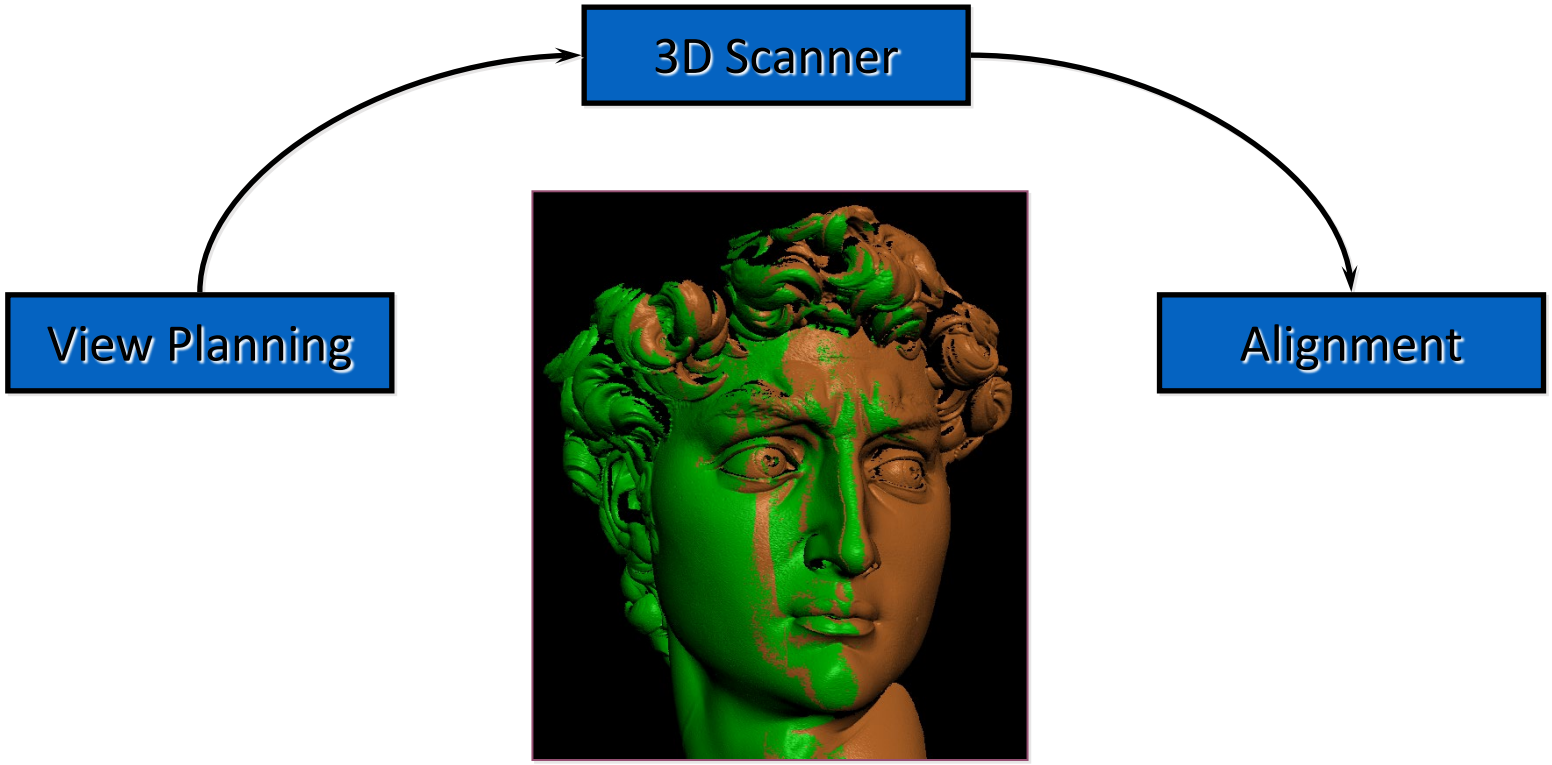
3D Scanner



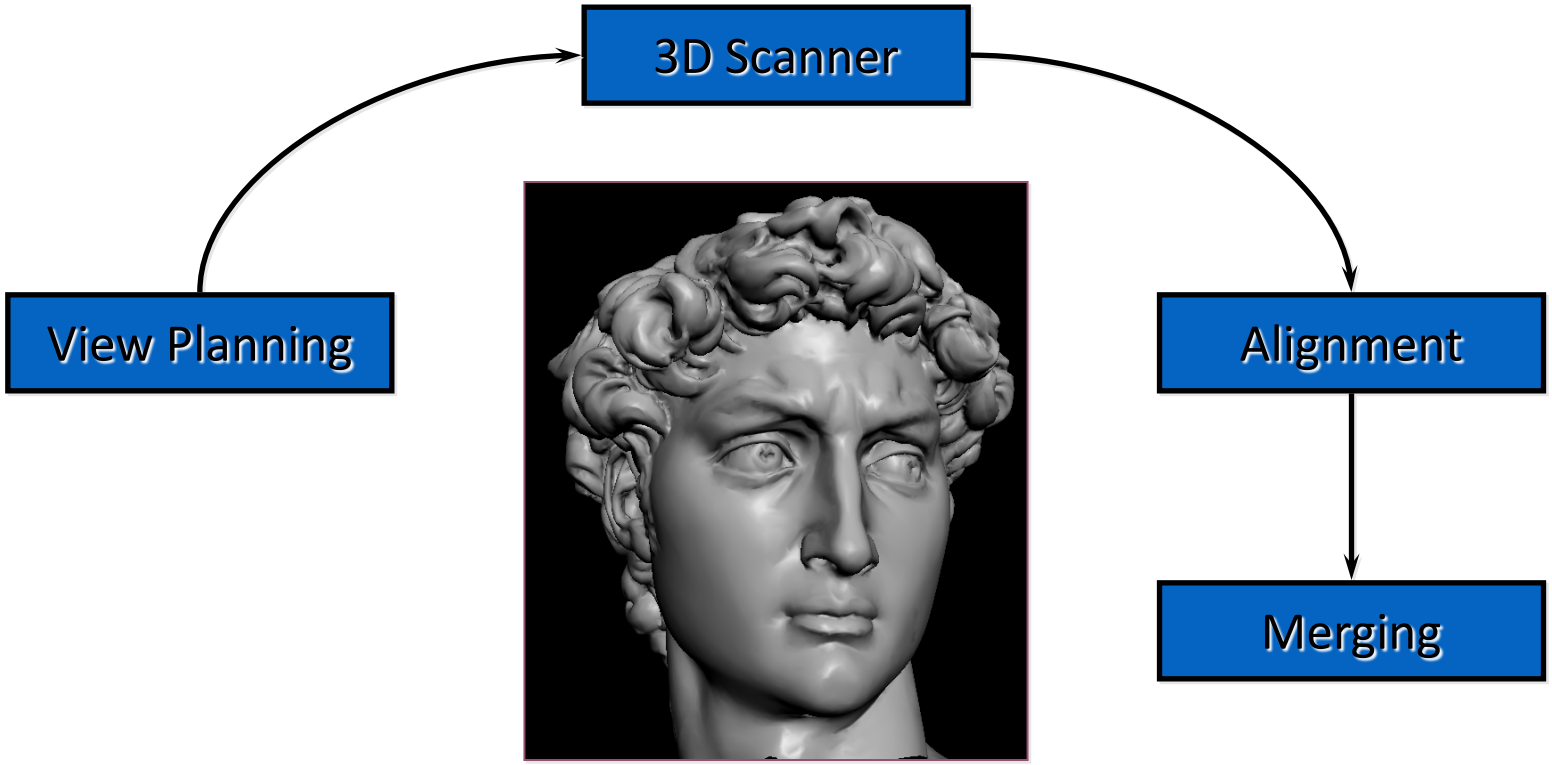
3D Model Acquisition Pipeline



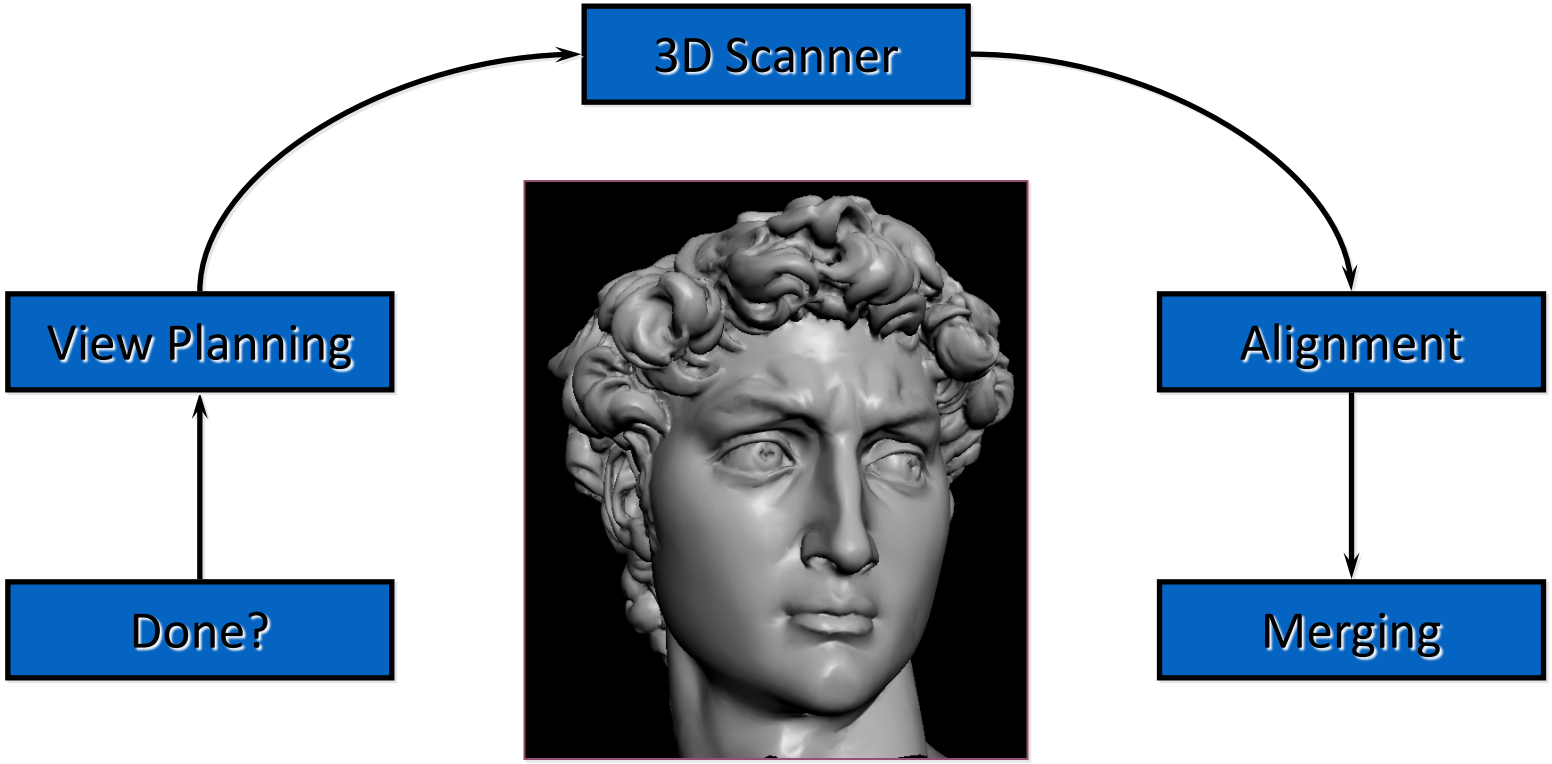
3D Model Acquisition Pipeline



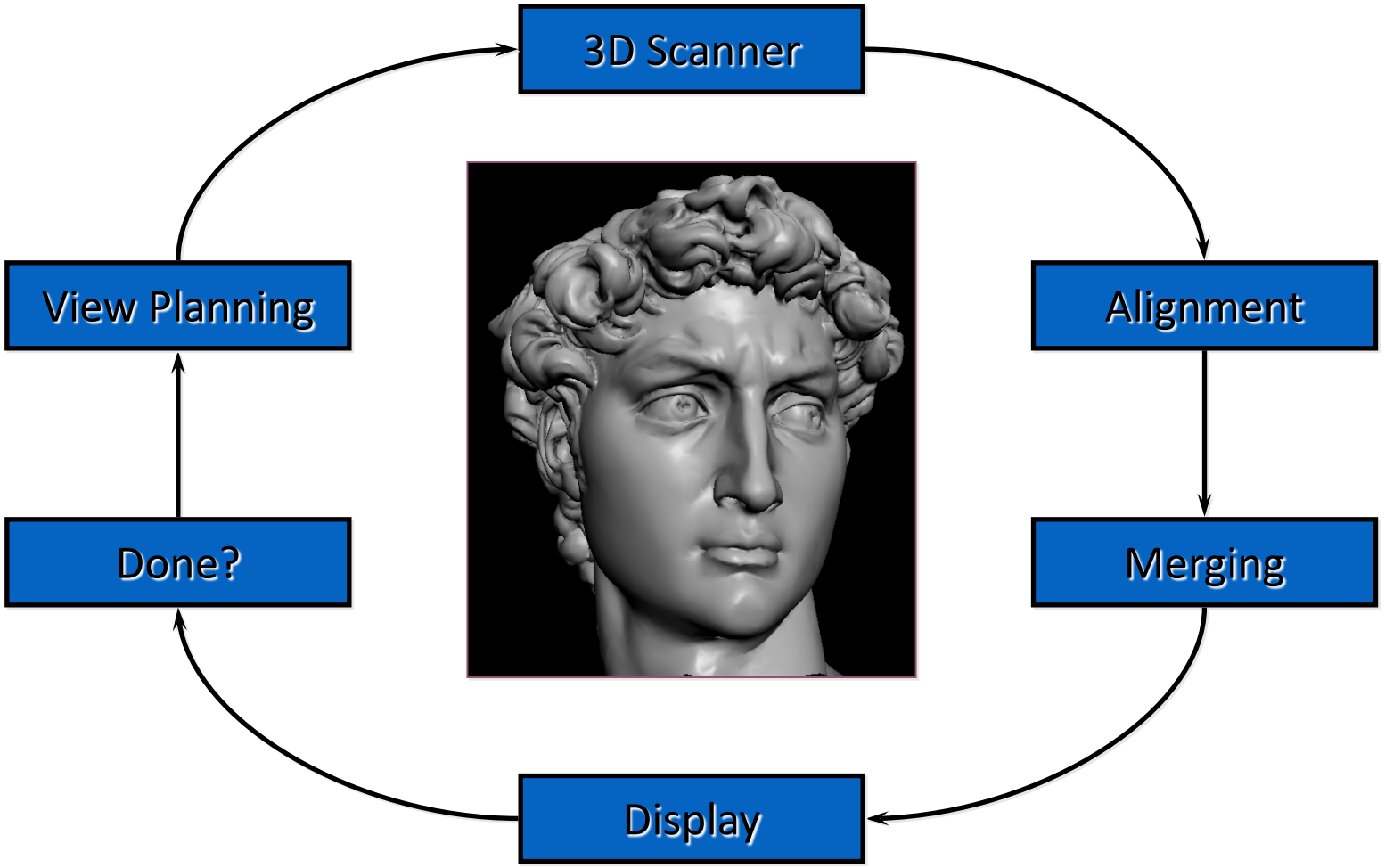
3D Model Acquisition Pipeline



3D Model Acquisition Pipeline



3D Model Acquisition Pipeline



References

Basic reading:

- Szeliski textbook, Sections 7.1, 11.1, 12.1.
- Boles et al., “Epipolar-plane image analysis: An approach to determining structure from motion,” IJCV 1987.
  - This classical paper introduces EPIs, and discusses how they can be used to infer depth.
- Lanman and Taubin, “Build Your Own 3D Scanner: Optical Triangulation for Beginners,” SIGGRAPH course 2009.
  - This very comprehensive course has everything you need to know about 3D scanning using structured light, including details on how to build your own.
- Bouguet and Perona, “3D Photography Using Shadows in Dual-Space Geometry,” IJCV 1999.
  - This paper introduces the idea of using shadows to do structured light 3D scanning, and shows an implementation using just a camera, desk lamp, and a stick.

Additional reading:

- Gupta et al., “A Practical Approach to 3D Scanning in the Presence of Interreflections, Subsurface Scattering and Defocus,” IJCV 2013.
  - This paper has a very detailed treatment of standard patterns used for structured light, problems arising due to global illumination, and robust patterns for dealing with these patterns.
- Barron et al., “Fast bilateral-space stereo for synthetic defocus,” CVPR 2015.
- Barron and Poole, “The fast bilateral solver,” ECCV 2016.
  - The above two papers show how to combine edge-aware filtering (and bilateral filtering in particular) with disparity matching for robust stereo. The first paper also shows how the resulting depth maps can be used to create synthetic defocus blur.
- Wanner and Goldluecke, “Globally Consistent Depth Labeling of 4D Light Fields,” CVPR 2012.
- Kim et al., “Scene reconstruction from high spatio-angular resolution light fields,” SIGGRAPH 2013.
  - These two papers show detailed systems for using EPIs to extract depth.
- Levin et al., “Understanding camera trade-offs through a Bayesian analysis of light field projections,” ECCV 2008.
  - This paper uses EPIs to show how different types of imaging systems (pinhole cameras, plenoptic cameras, stereo pairs, lens-based systems, and so on) relate to each other, and analyze their pros and cons for 3D imaging.