Two-view geometry, stereo, and disparity

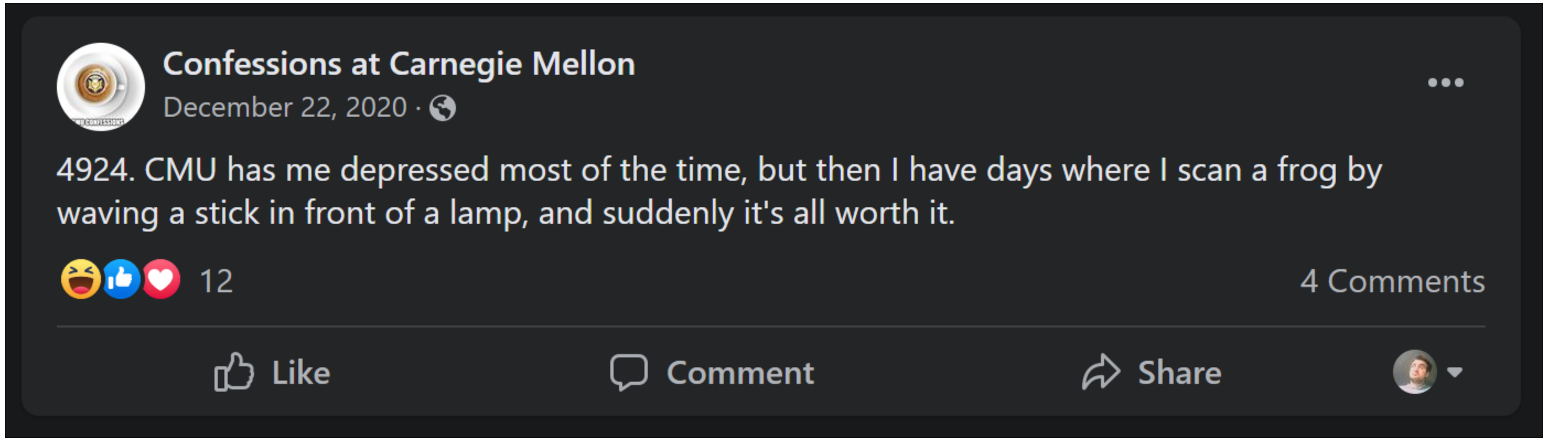
15-463, 15-663, 15-862
Computational Photography
Fall 2023, Lecture 16

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

Course announcements

- Homework assignment 6 posted, due December 8.
  - Start early: Capturing structured light stereo is challenging.

- Final project proposals due today.



Overview of today’s lecture

- Triangulation.
- Epipolar geometry.
- Revisiting triangulation.
- Disparity.
- Revisiting lightfields.
- Structured light.
- Some notes on focusing.

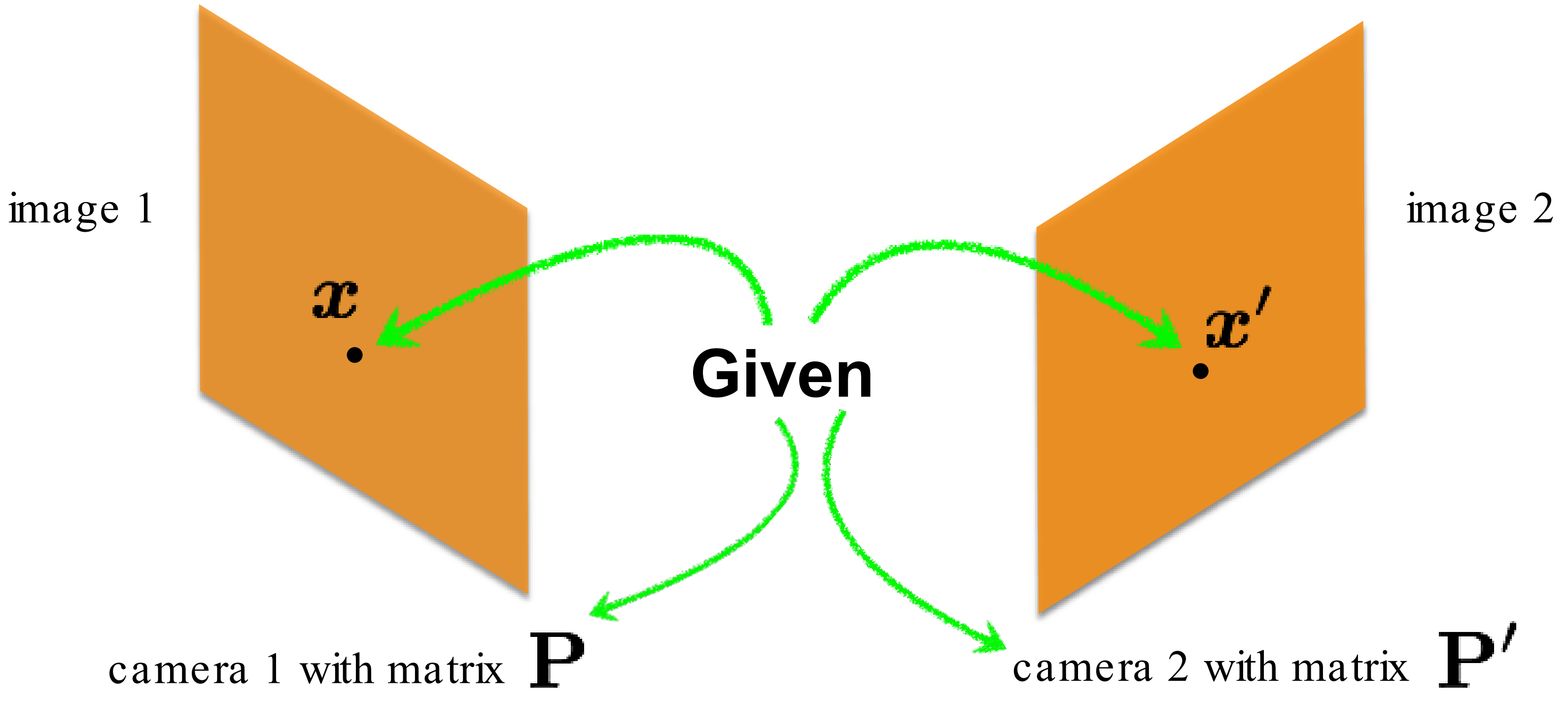
Slide credits

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).

Triangulation

Triangulation



Triangulation

Which 3D points map
to $\mathbf{x}$?

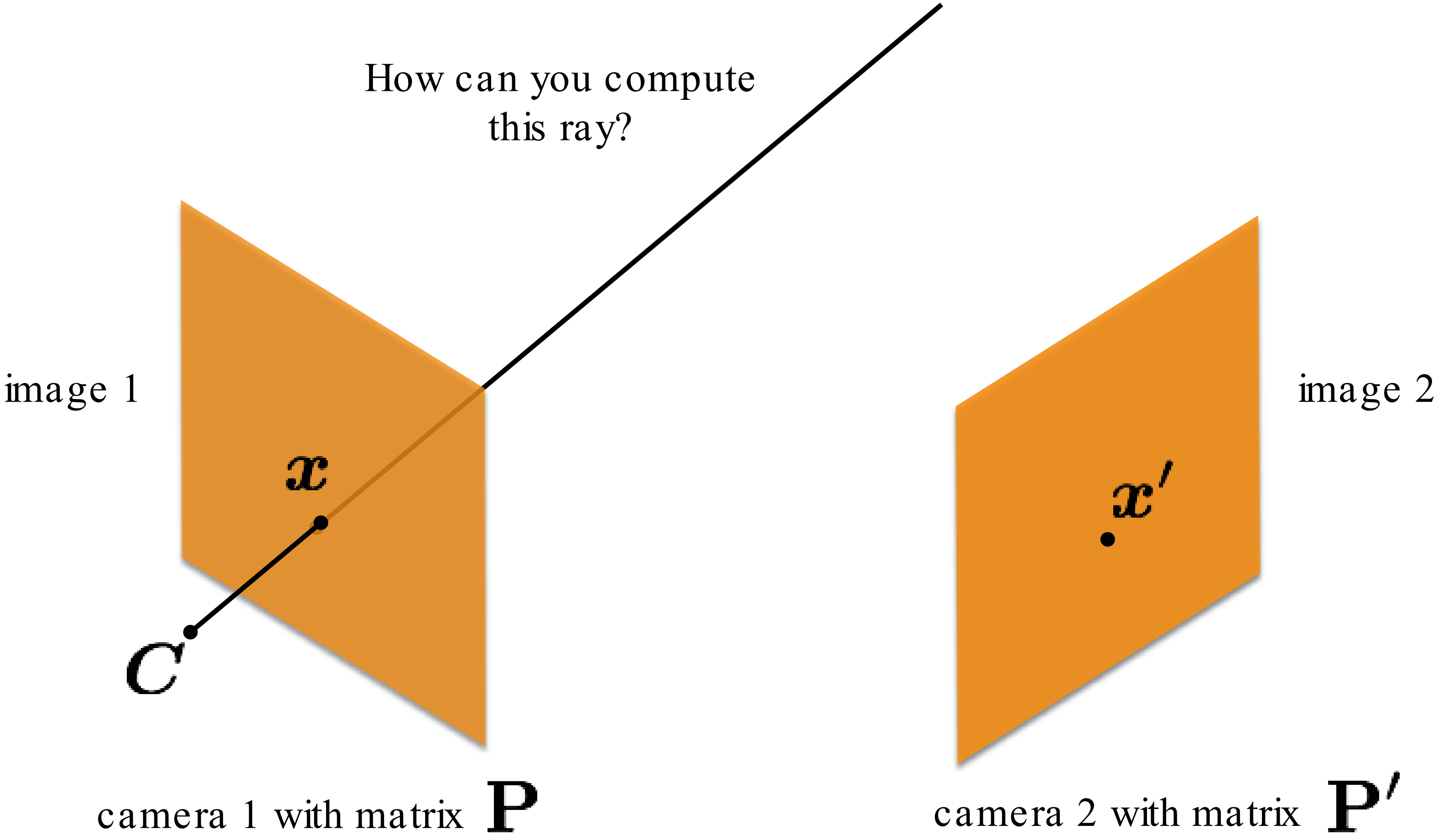
image 1

image 2

camera 1 with matrix $\mathbf{P}$

camera 2 with matrix $\mathbf{P}'$

Triangulation



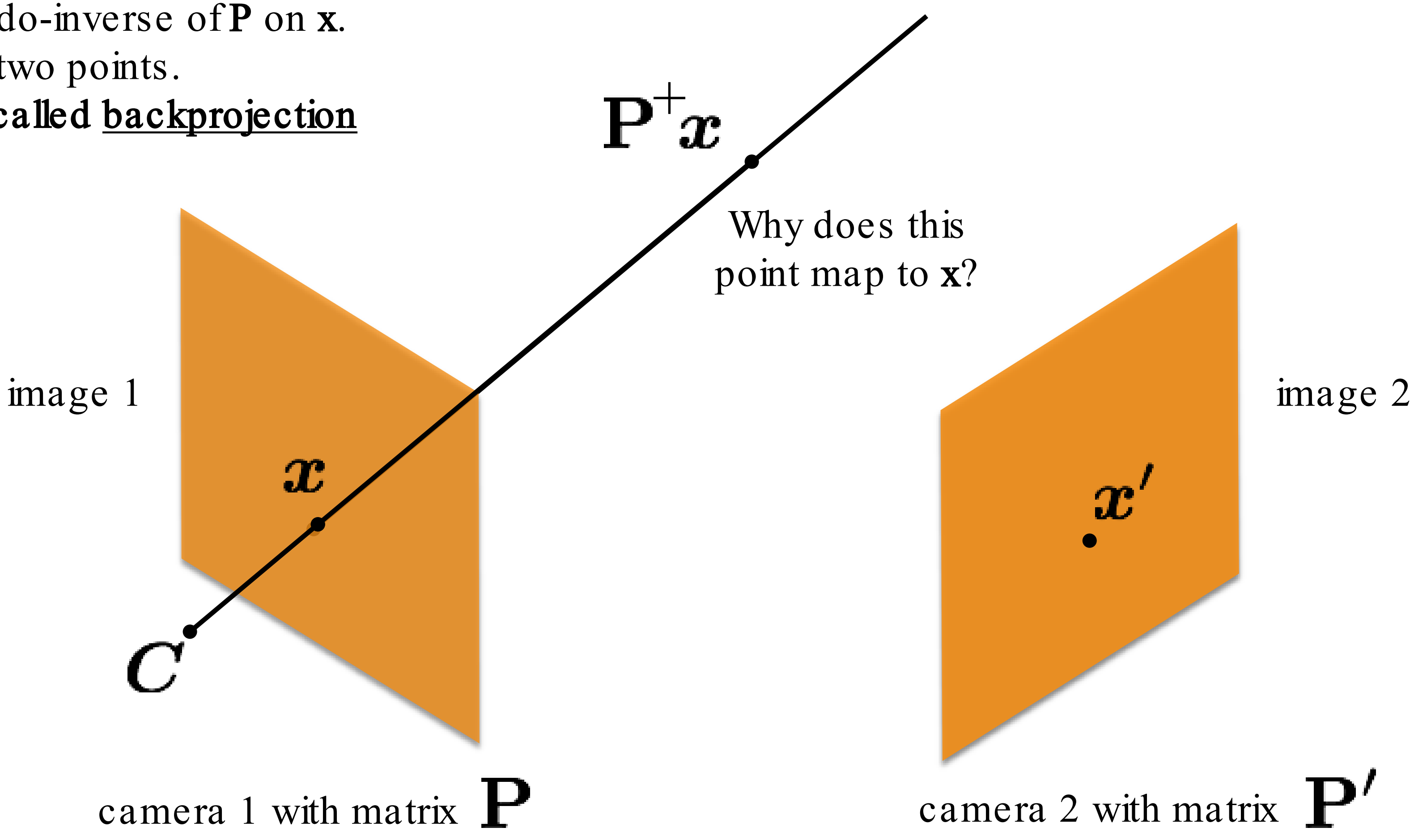
Triangulation

Create two points on the ray:

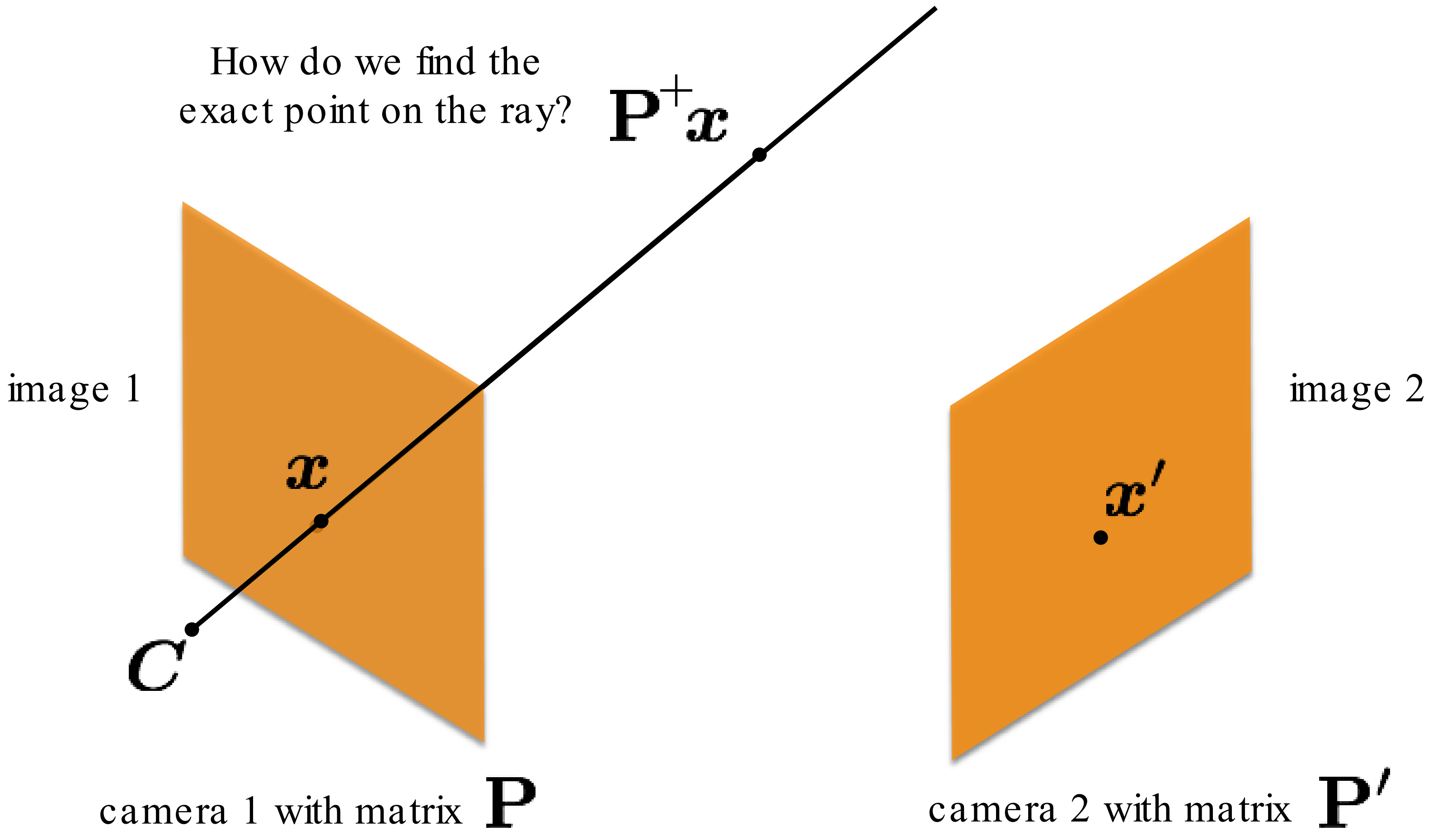
1) find the camera center; and
2) apply the pseudo-inverse of **P** on **x**.

Then connect the two points.

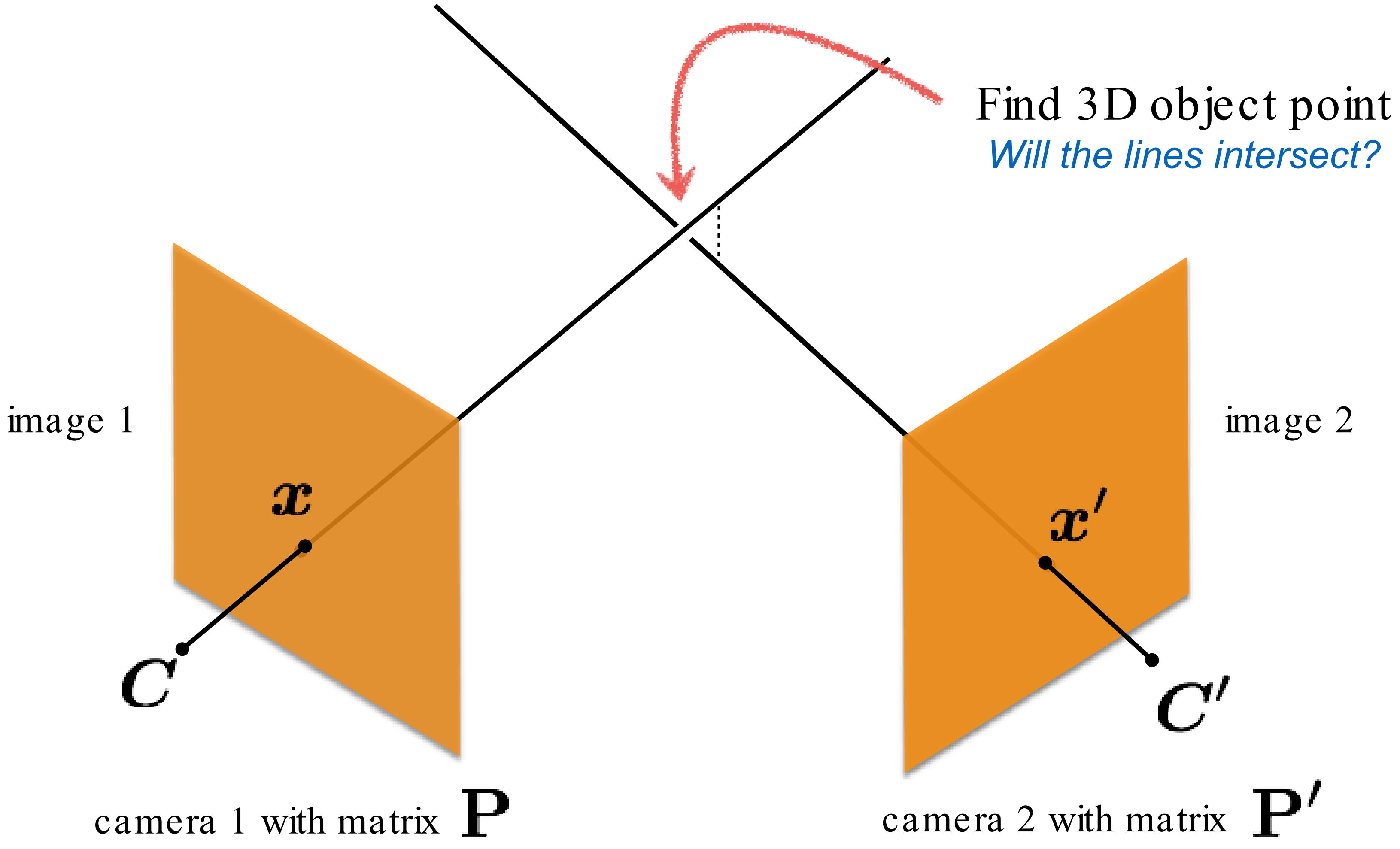
This procedure is called <u>backprojection</u>



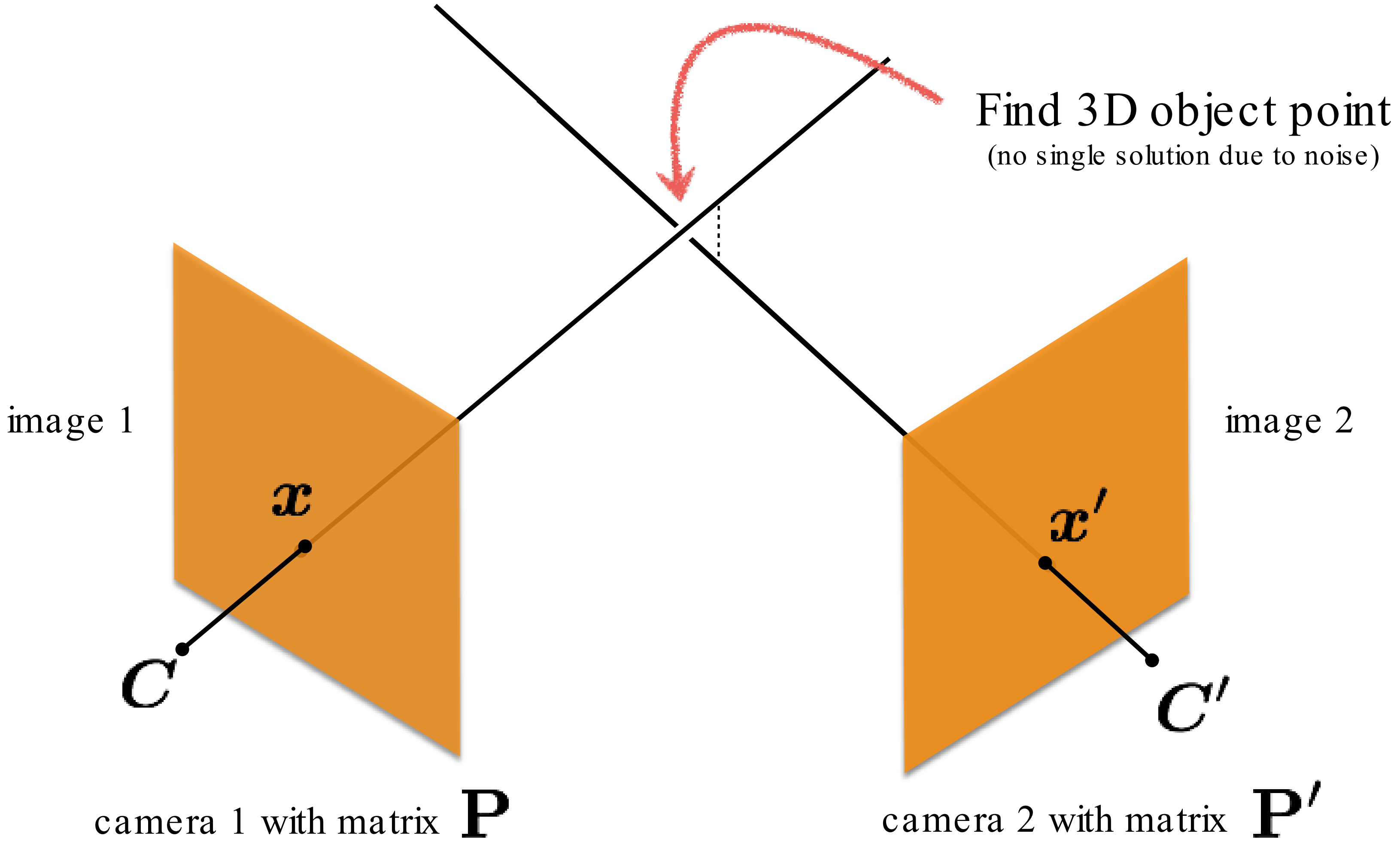
Triangulation



Triangulation



Triangulation



Triangulation

Given a set of (noisy) matched points

$$\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}$$

and camera matrices

$$\mathbf{P}, \mathbf{P}'$$

Estimate the 3D point

$$\mathbf{X}$$

$$\underbrace{\mathbf{x}}_{\text{known}} = \underbrace{\mathbf{P}}_{\text{known}}\boldsymbol{X}$$

*Can we compute **X** from a single correspondence **x**?*

$$\mathbf{x} = \mathbf{P}\boldsymbol{X}$$

(homogeneous coordinate)

This is a similarity relation because it involves homogeneous coordinates

$$\mathbf{x} = \alpha\mathbf{P}\boldsymbol{X}$$

(homorogeneous coordinate)

Same ray direction but differs by a scale factor

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

How do we solve for unknowns in a similarity relation?

$$\mathbf{x} = \mathbf{P}\boldsymbol{X}$$

(homogeneous coordinate)

Also, this is a similarity relation because it involves homogeneous coordinates

$$\mathbf{x} = \alpha\mathbf{P}\boldsymbol{X}$$

(inhomogeneous coordinate)

Same ray direction but differs by a scale factor

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

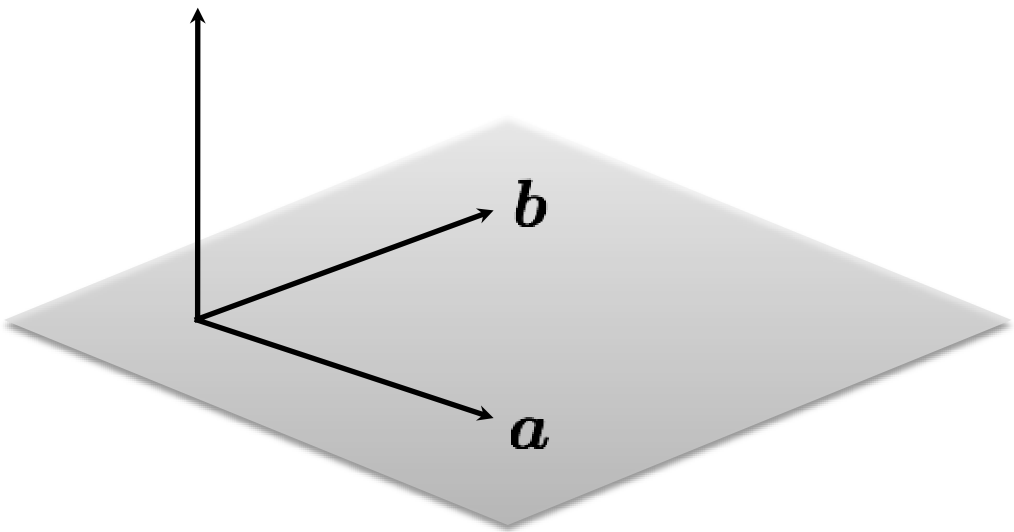
How do we solve for unknowns in a similarity relation?

Linear algebra reminder: cross product

Vector (cross) product

takes two vectors and returns a vector perpendicular to both

$$\boldsymbol{c} = \boldsymbol{a} \times \boldsymbol{b}$$



$$\boldsymbol{c} \cdot \boldsymbol{a} = 0 \qquad \boldsymbol{c} \cdot \boldsymbol{b} = 0$$

$$\boldsymbol{a} \times \boldsymbol{b} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix}$$

cross product of two vectors in the same direction is zero vector

$$\boldsymbol{a} \times \boldsymbol{a} = 0$$

remember this!!!

Linear algebra reminder: cross product

Cross product

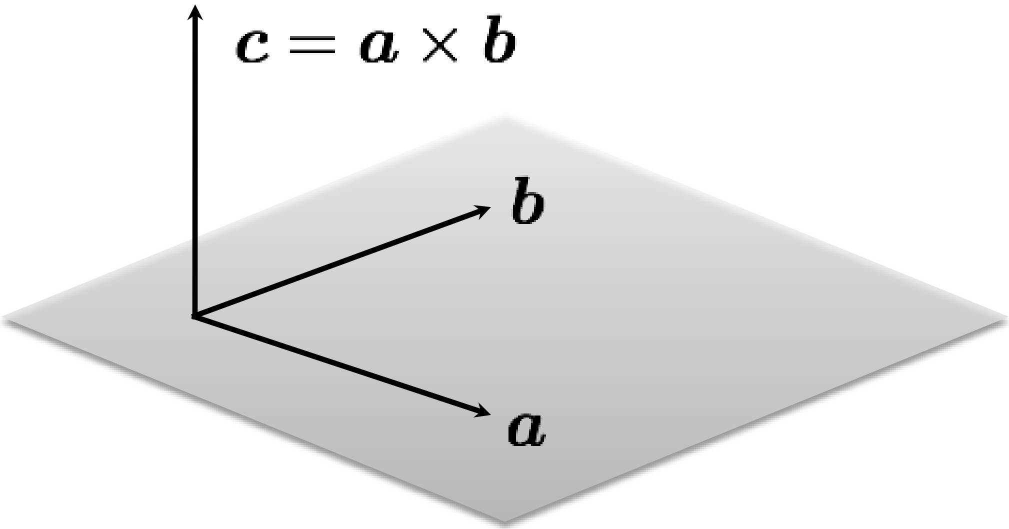
$$\boldsymbol{a} \times \boldsymbol{b} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix}$$

Can also be written as a matrix multiplication

$$\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}]_\times \boldsymbol{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Skew symmetric

Compare with: dot product



$$\boldsymbol{c} \cdot \boldsymbol{a} = 0 \qquad \boldsymbol{c} \cdot \boldsymbol{b} = 0$$

dot product of two orthogonal vectors is (scalar) zero

Back to triangulation

$$\mathbf{x} = \alpha \mathbf{P} \boldsymbol{X}$$

Same direction but differs by a scale factor

How can we rewrite this using vector products?

$$\mathbf{x} = \alpha \mathbf{P} \boldsymbol{X}$$

Same direction but differs by a scale factor

$$\mathbf{x} \times \mathbf{P} \boldsymbol{X} = \mathbf{0}$$

Cross product of two vectors of same direction is zero
(this equality removes the scale factor)

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Do the same after first expanding out the camera matrix and points

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} \text{---} & \boldsymbol{p}_1^\top & \text{---} \\ \text{---} & \boldsymbol{p}_2^\top & \text{---} \\ \text{---} & \boldsymbol{p}_3^\top & \text{---} \end{bmatrix} \begin{bmatrix} | \\ \boldsymbol{X} \\ | \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} \boldsymbol{p}_1^\top \boldsymbol{X} \\ \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_3^\top \boldsymbol{X} \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \times \begin{bmatrix} \boldsymbol{p}_1^\top \boldsymbol{X} \\ \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_3^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} y\boldsymbol{p}_3^\top \boldsymbol{X} - \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_1^\top \boldsymbol{X} - x\boldsymbol{p}_3^\top \boldsymbol{X} \\ x\boldsymbol{p}_2^\top \boldsymbol{X} - y\boldsymbol{p}_1^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Using the fact that the cross product should be zero

$$\mathbf{x} \times \mathbf{P}\boldsymbol{X} = \mathbf{0}$$

$$\begin{bmatrix} y\boldsymbol{p}_3^\top \boldsymbol{X} - \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_1^\top \boldsymbol{X} - x\boldsymbol{p}_3^\top \boldsymbol{X} \\ x\boldsymbol{p}_2^\top \boldsymbol{X} - y\boldsymbol{p}_1^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Third line is a linear combination of the first and second lines.
(x times the first line plus y times the second line)

One 2D to 3D point correspondence give you [redacted] equations

Using the fact that the cross product should be zero

$$\mathbf{x} \times \mathbf{P}\boldsymbol{X} = \mathbf{0}$$

$$\begin{bmatrix} y\boldsymbol{p}_3^\top \boldsymbol{X} - \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_1^\top \boldsymbol{X} - x\boldsymbol{p}_3^\top \boldsymbol{X} \\ x\boldsymbol{p}_2^\top \boldsymbol{X} - y\boldsymbol{p}_1^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Third line is a linear combination of the first and second lines.
(x times the first line plus y times the second line)

One 2D to 3D point correspondence give you 2 equations

$$\begin{bmatrix} y\boldsymbol{p}_3^\top \boldsymbol{X} - \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_1^\top \boldsymbol{X} - x\boldsymbol{p}_3^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Remove third row, and rearrange as system on unknowns

$$\begin{bmatrix} y\boldsymbol{p}_3^\top - \boldsymbol{p}_2^\top \\ \boldsymbol{p}_1^\top - x\boldsymbol{p}_3^\top \end{bmatrix} \boldsymbol{X} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathbf{A}_i \boldsymbol{X} = \mathbf{0}$$

Now we can make a system of linear equations
(two lines for each 2D point correspondence)

Concatenate the 2D points from both images

Two rows from camera one

Two rows from camera two

$$\begin{bmatrix} y\boldsymbol{p}_3^\top - \boldsymbol{p}_2^\top \\ \boldsymbol{p}_1^\top - x\boldsymbol{p}_3^\top \\ y'\boldsymbol{p}_3'^\top - \boldsymbol{p}_2'^\top \\ \boldsymbol{p}_1'^\top - x'\boldsymbol{p}_3'^\top \end{bmatrix} \boldsymbol{X} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

sanity check! dimensions?

$$\mathbf{A}\boldsymbol{X} = \mathbf{0}$$

How do we solve homogeneous linear system?

Concatenate the 2D points from both images

$$\begin{bmatrix} y\boldsymbol{p}_3^\top - \boldsymbol{p}_2^\top \\ \boldsymbol{p}_1^\top - x\boldsymbol{p}_3^\top \\ y'\boldsymbol{p}_3'^\top - \boldsymbol{p}_2'^\top \\ \boldsymbol{p}_1'^\top - x'\boldsymbol{p}_3'^\top \end{bmatrix} \boldsymbol{X} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{A}\boldsymbol{X} = \mathbf{0}$$

How do we solve homogeneous linear system?

S V D !

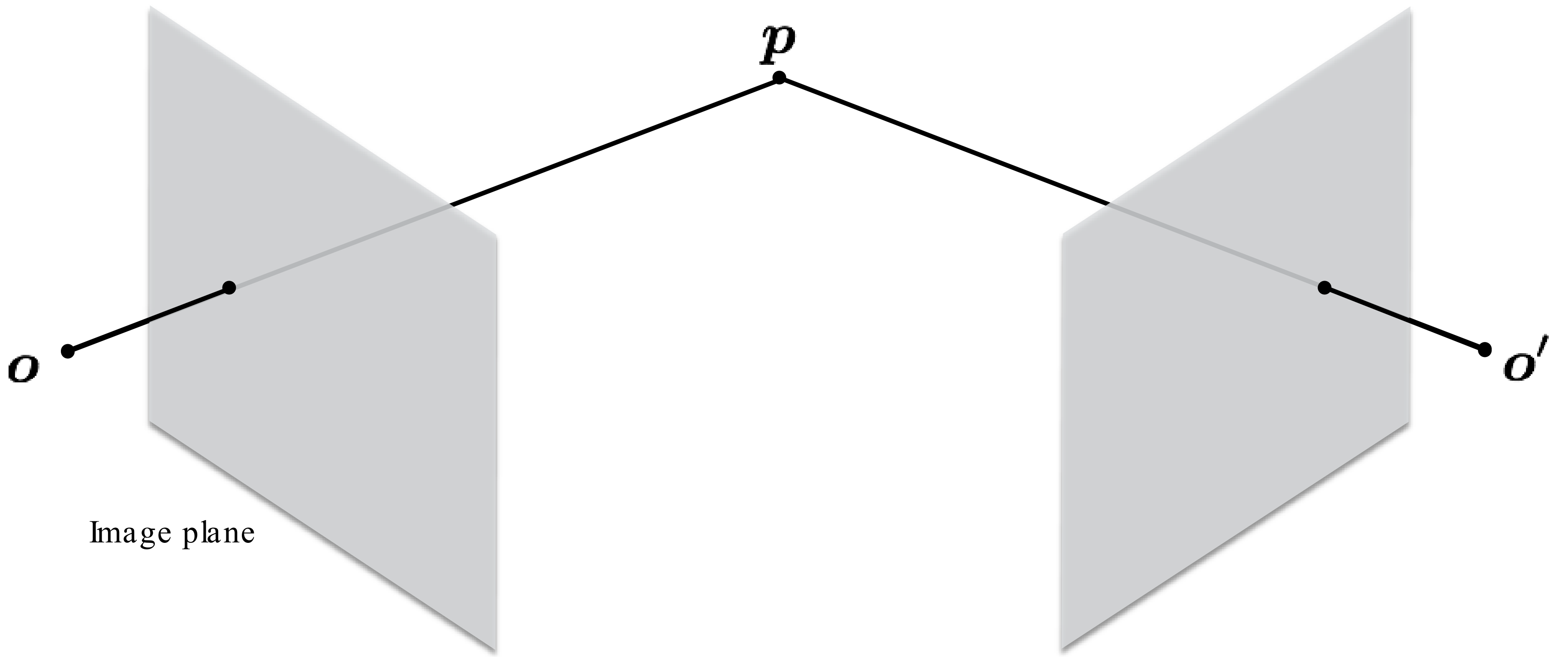
How would you reconstruct 3D points?

Left image

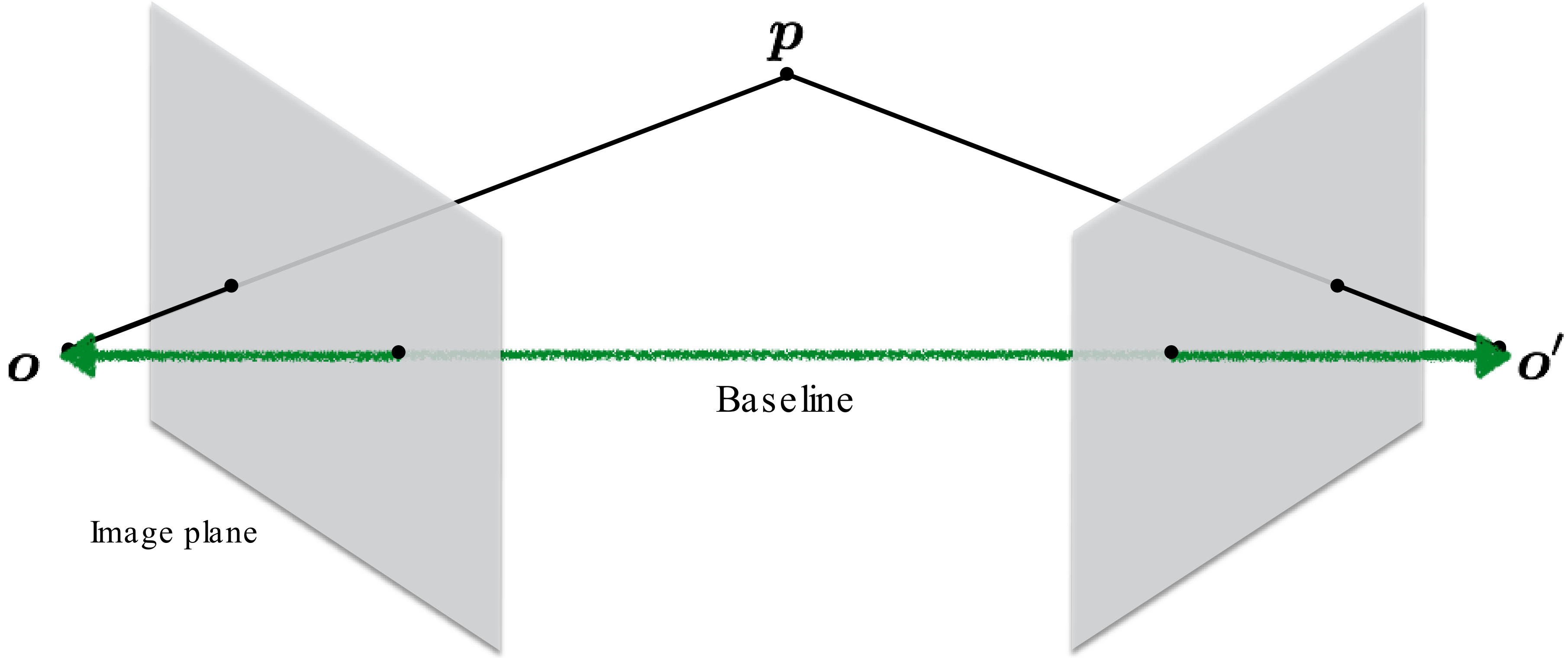
Right image

Epipolar geometry

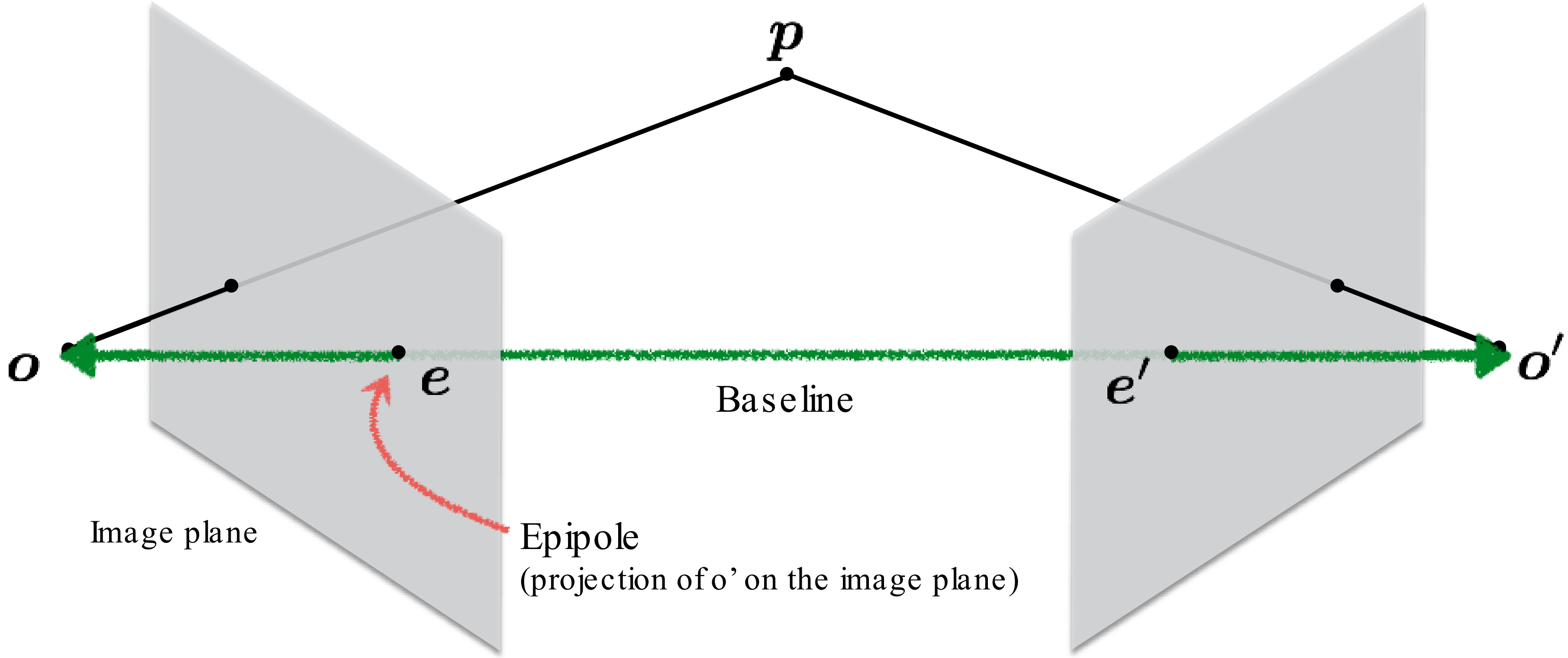
Epipolar geometry



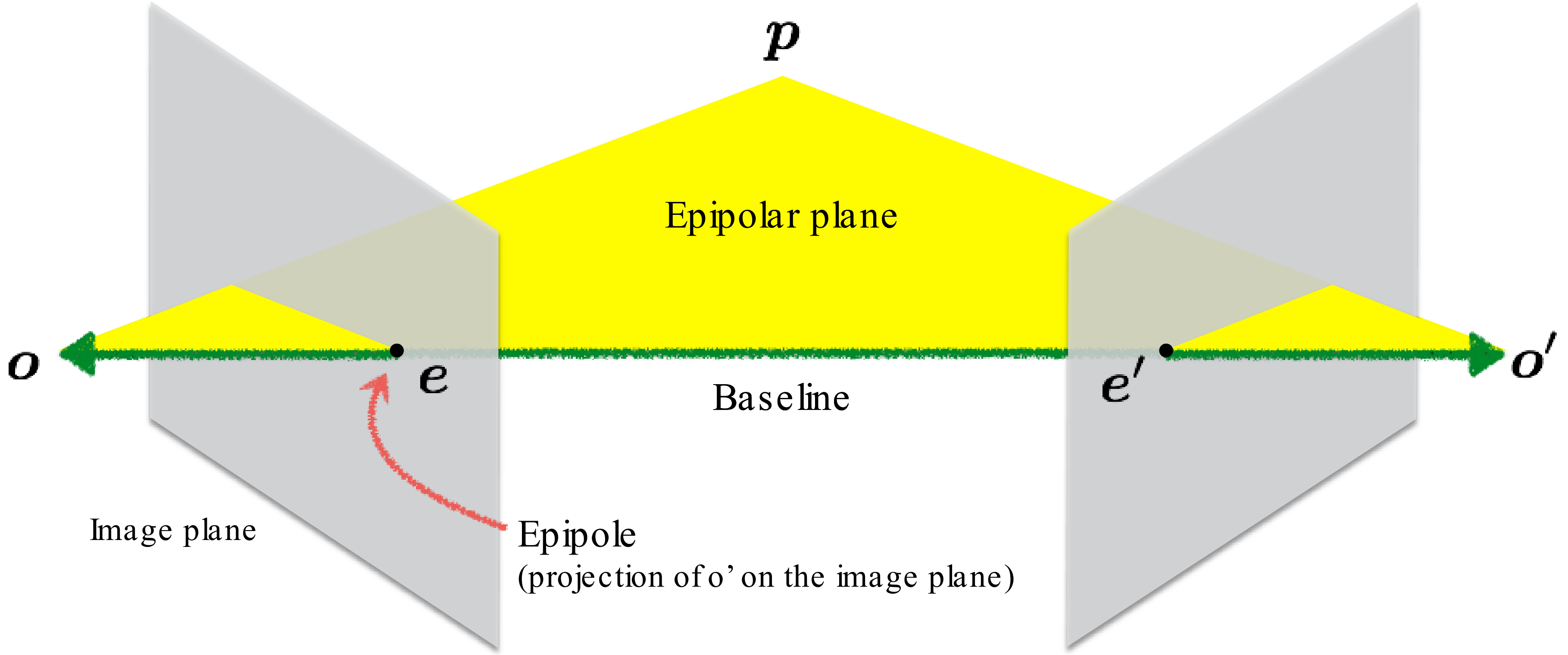
Epipolar geometry



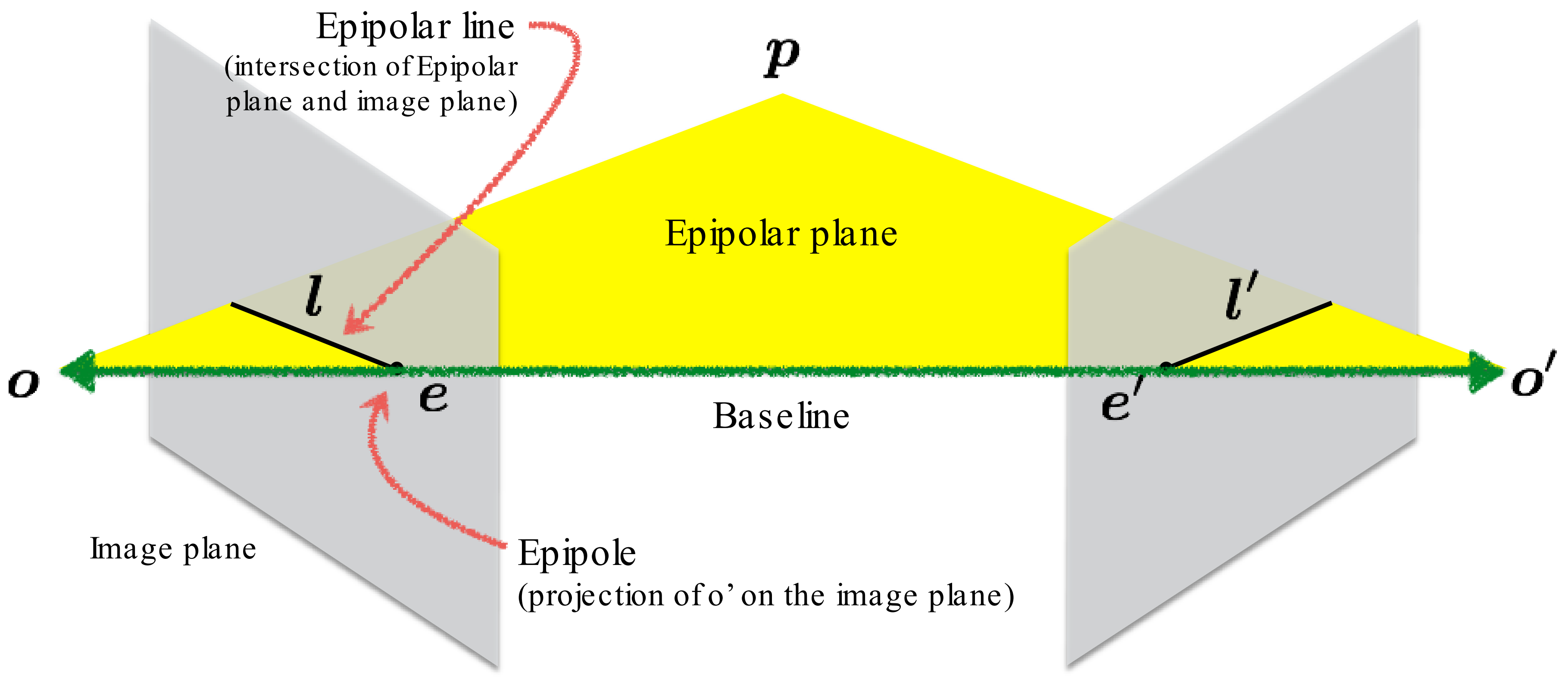
Epipolar geometry



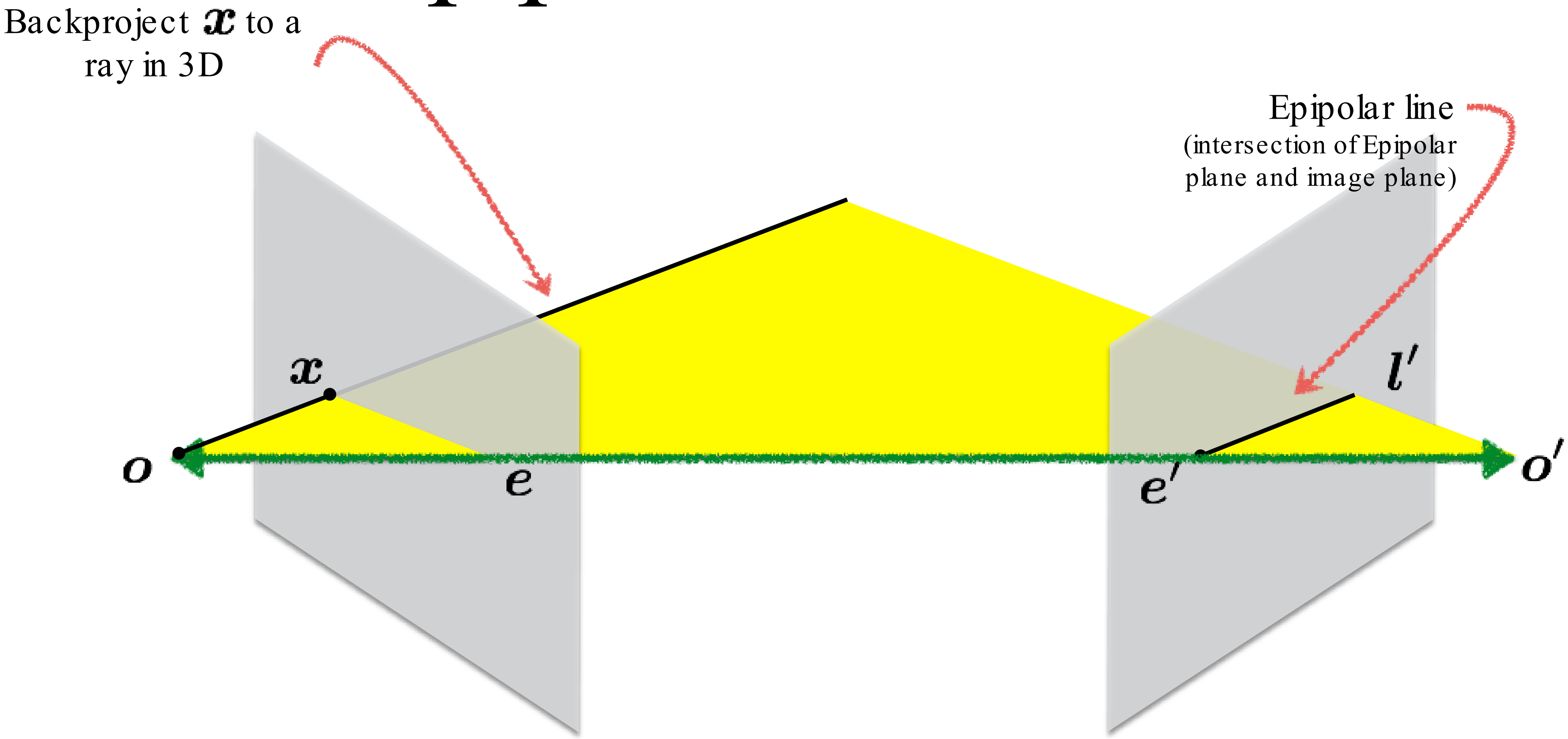
Epipolar geometry



Epipolar geometry

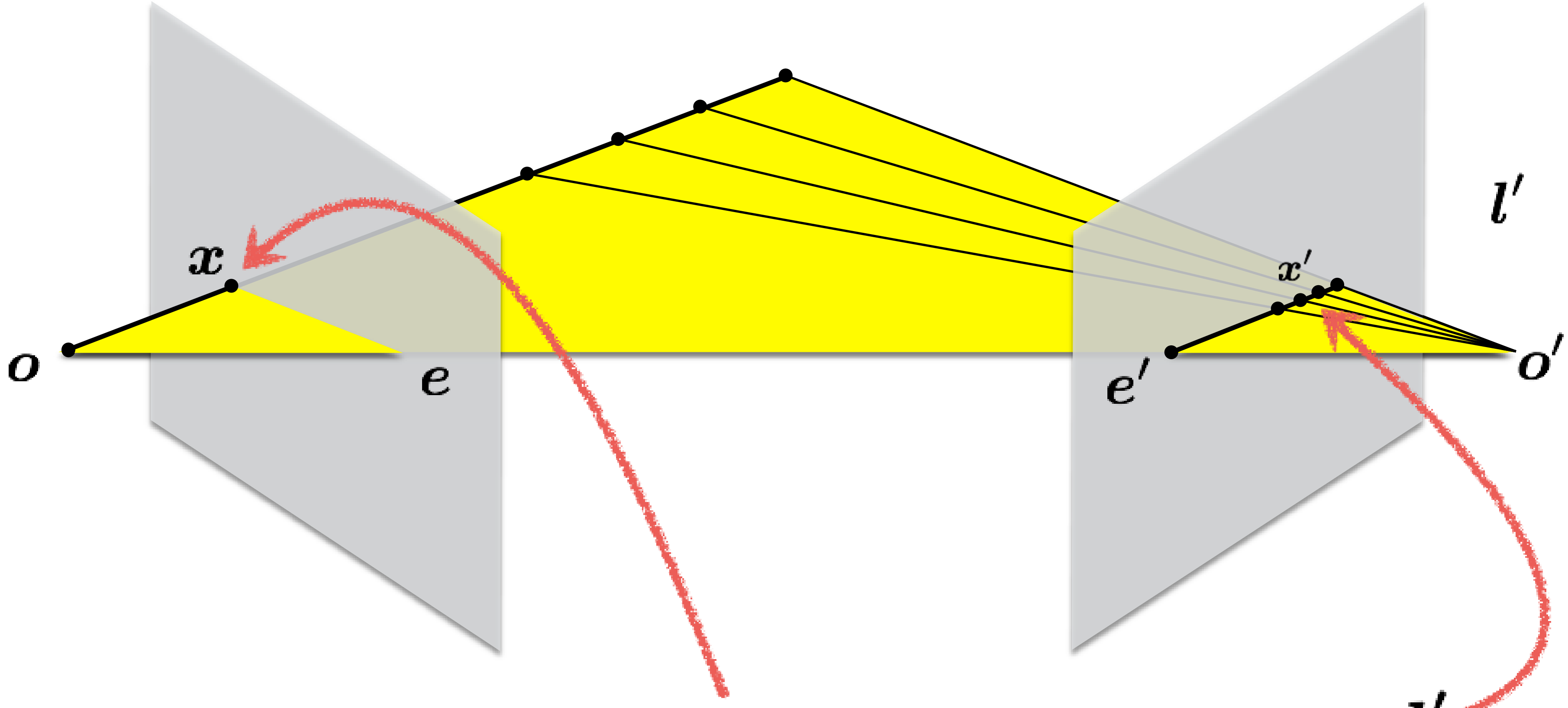


Epipolar constraint

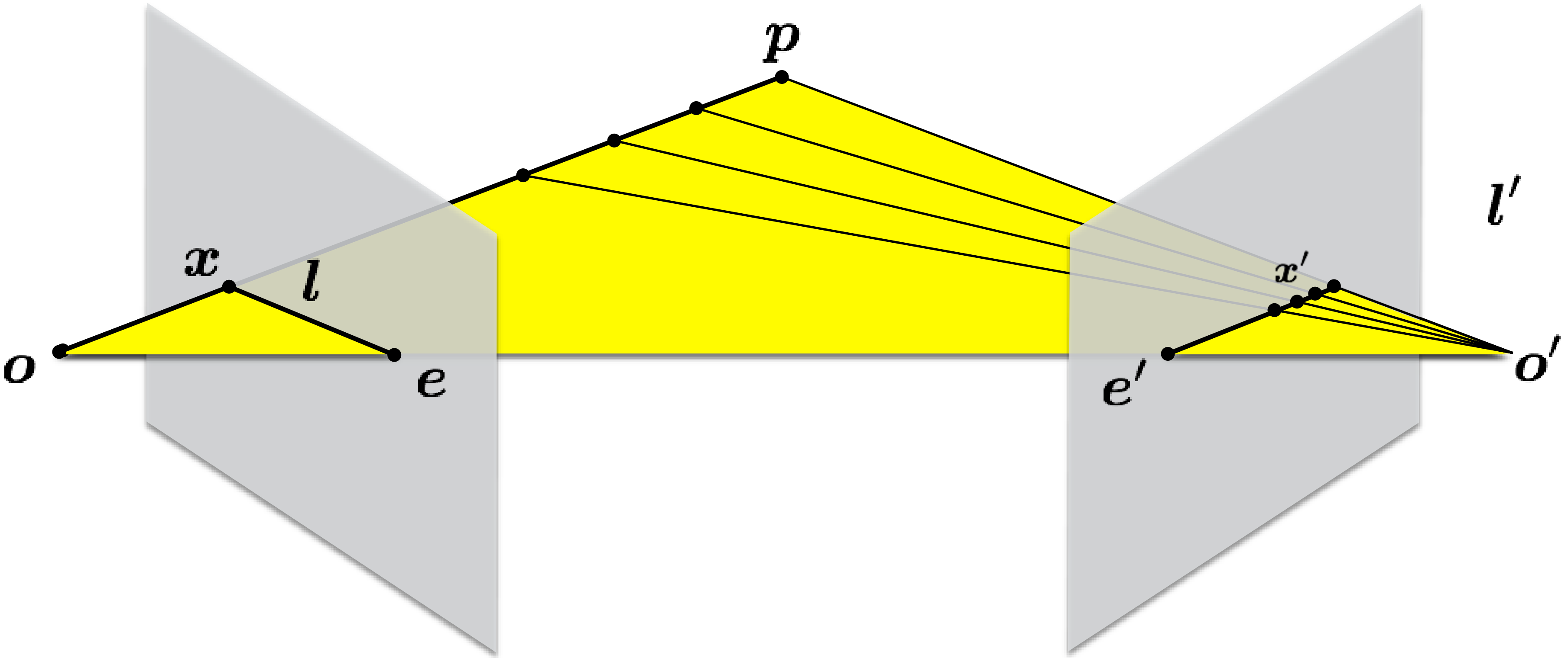


Another way to construct the epipolar plane, this time given x

Epipolar constraint



Potential matches for x lie on the epipolar line l'



The point **x** (left image) maps to a ____________ in the right image

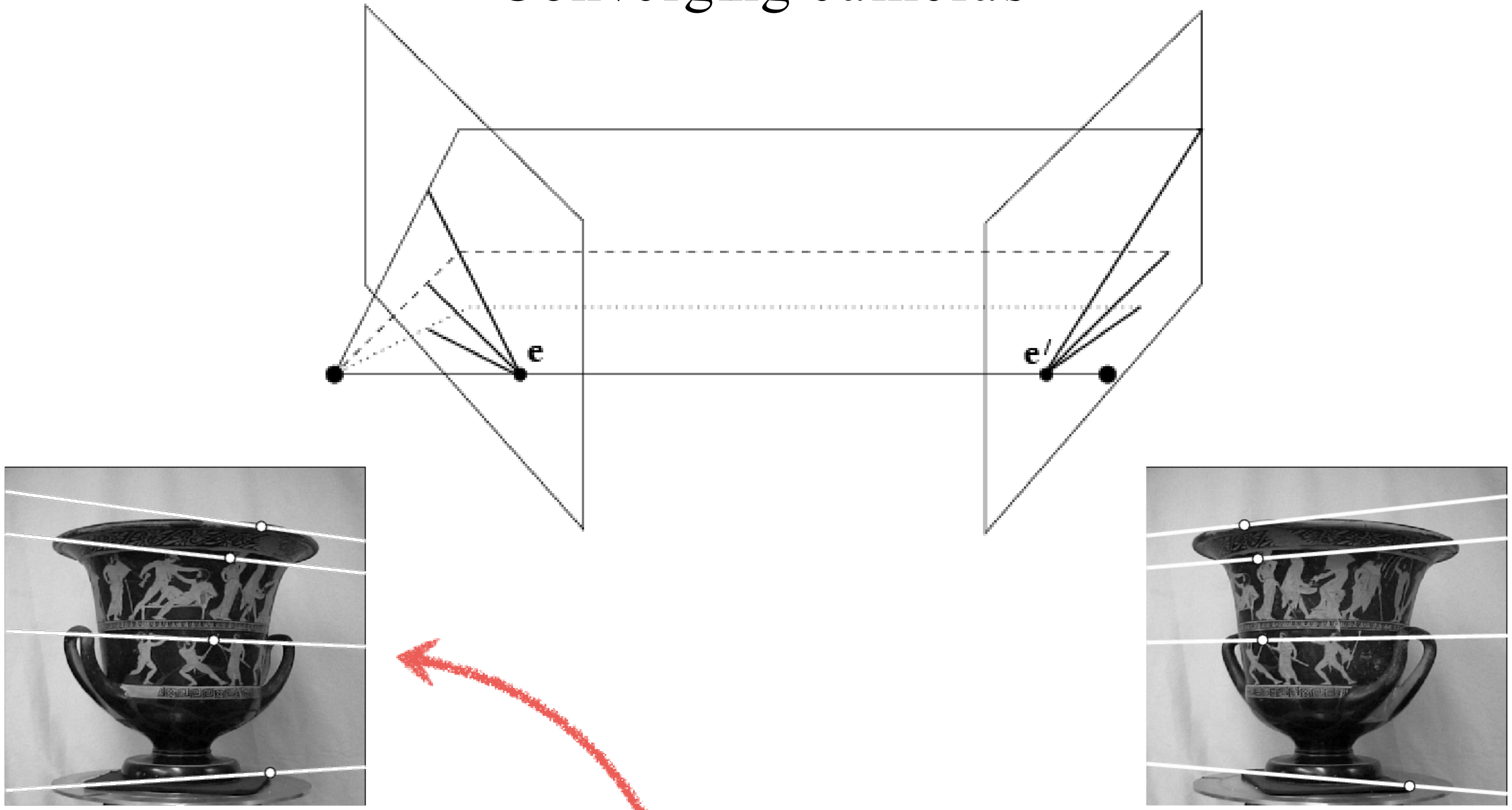
The baseline connects the ____________ and _____________

An epipolar line (left image) maps to a ___________ in the right image

An epipole **e** is a projection of the _______________ on the image plane

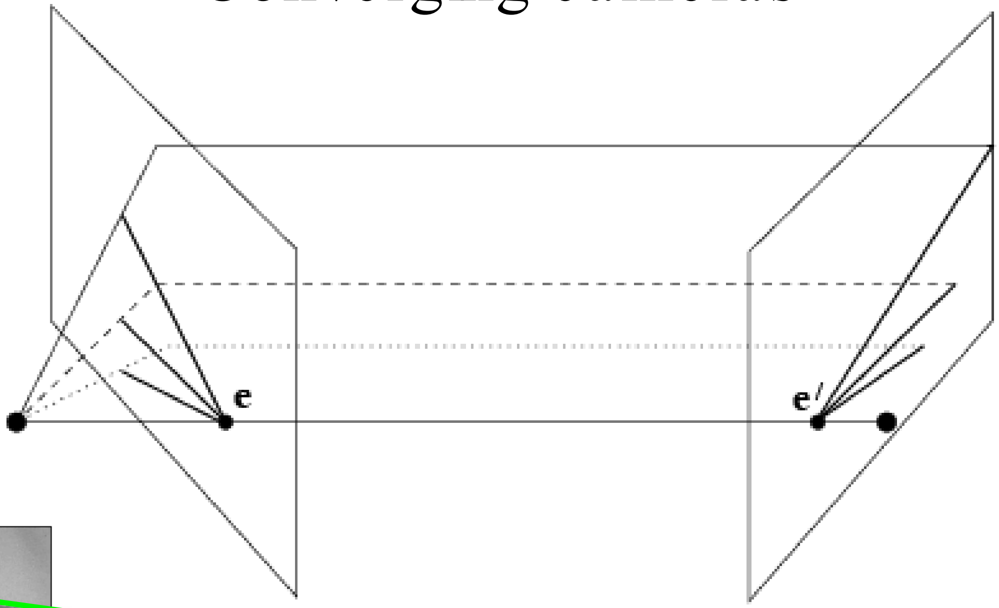
All epipolar lines in an image intersect at the _______________

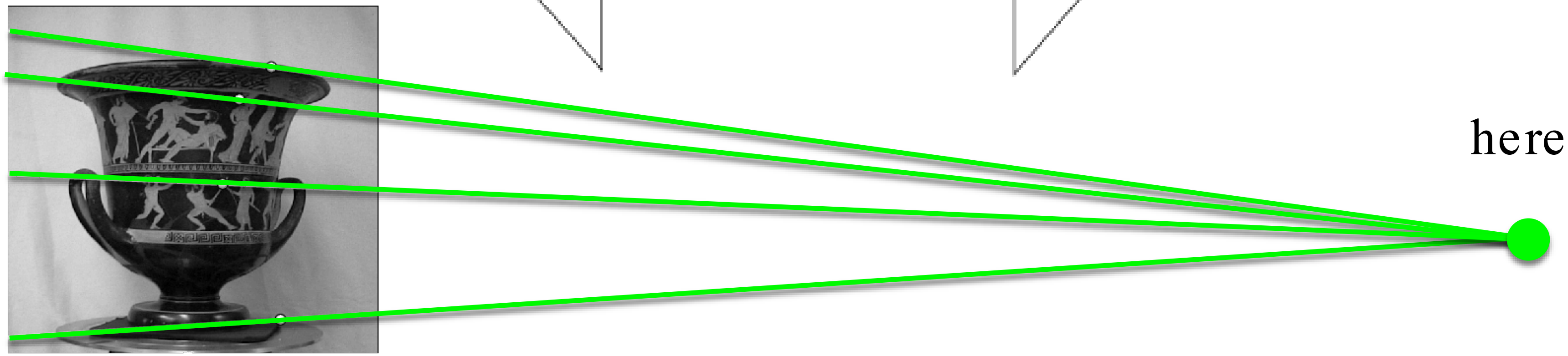
Converging cameras



Where is the epipole in this image?

Converging cameras

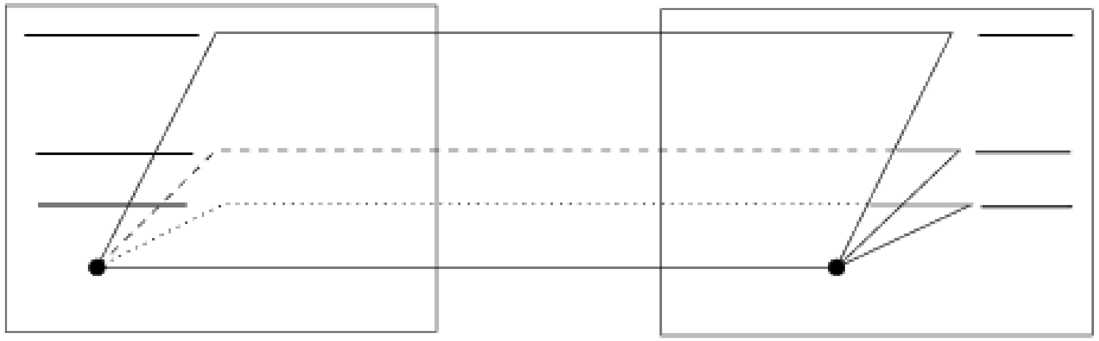


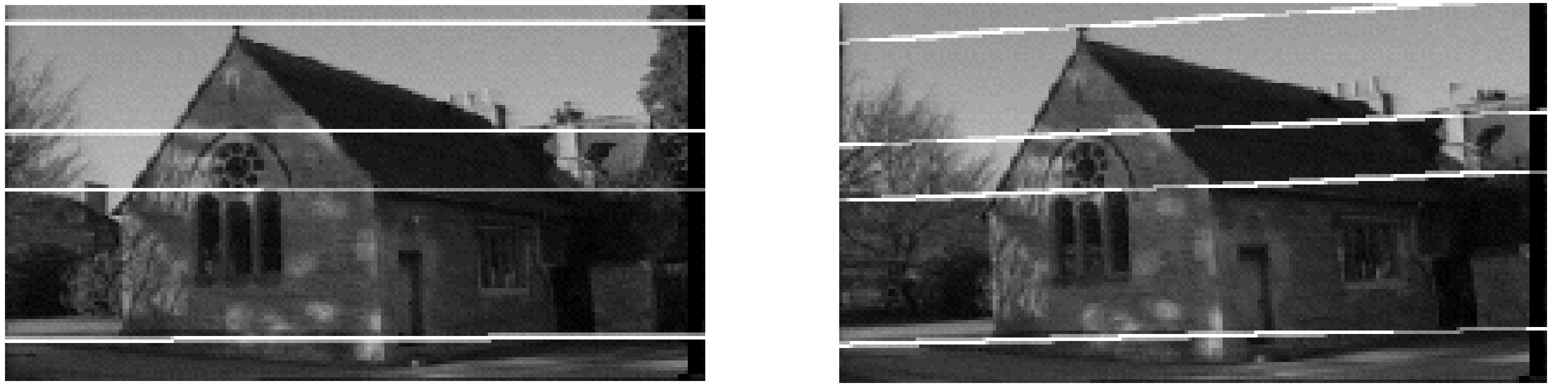


Where is the epipole in this image?

It's not always in the image

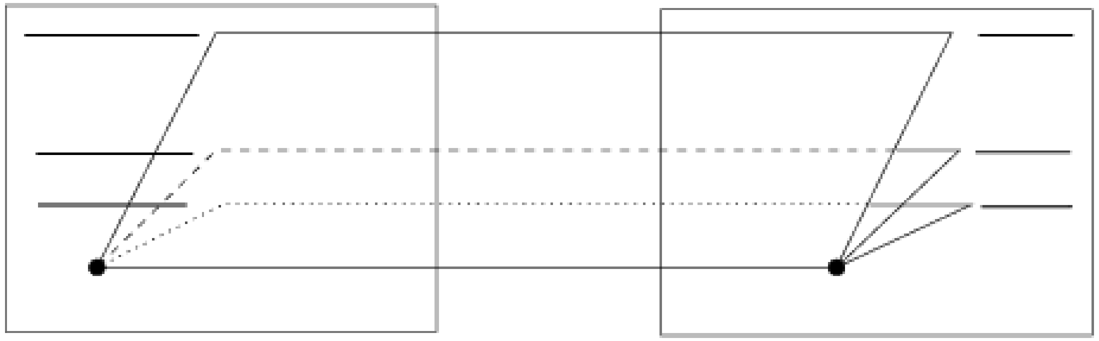
Parallel cameras

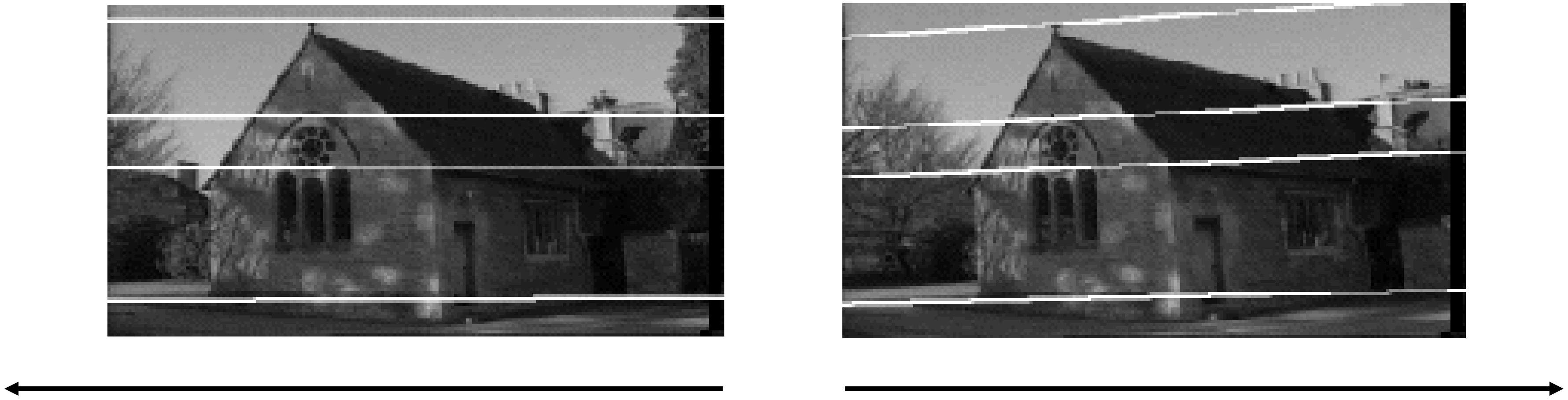




Where is the epipole?

Parallel cameras

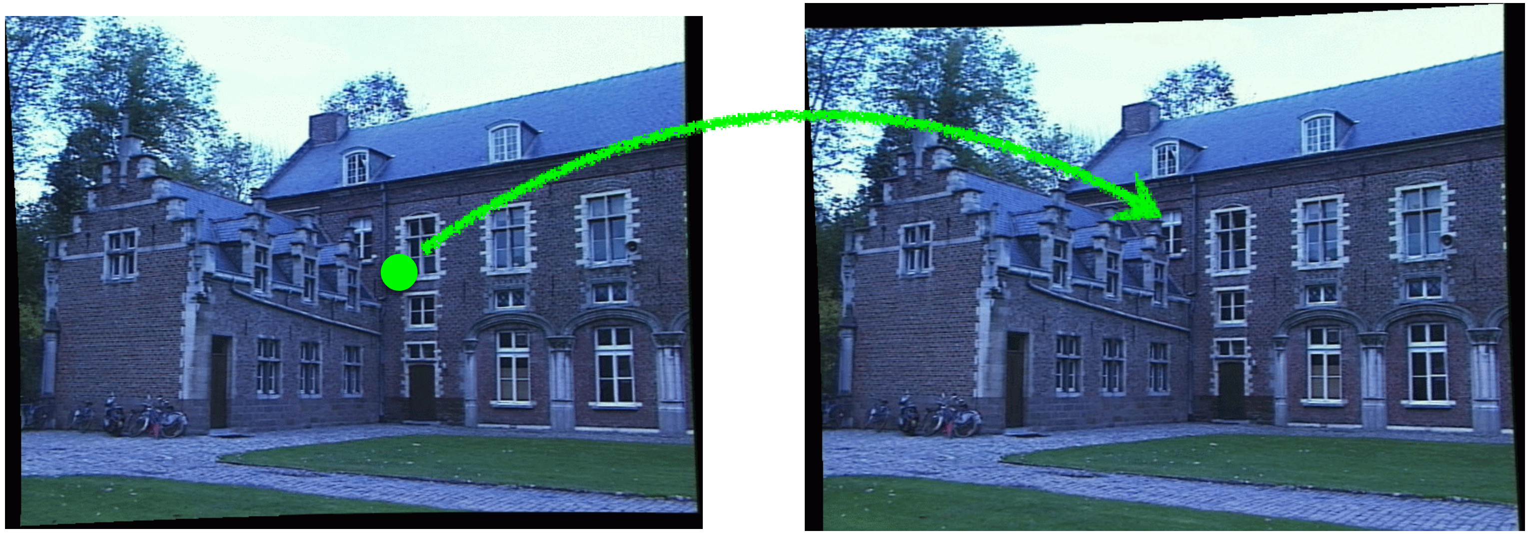




epipole at infinity

The epipolar constraint is an important concept for stereo vision

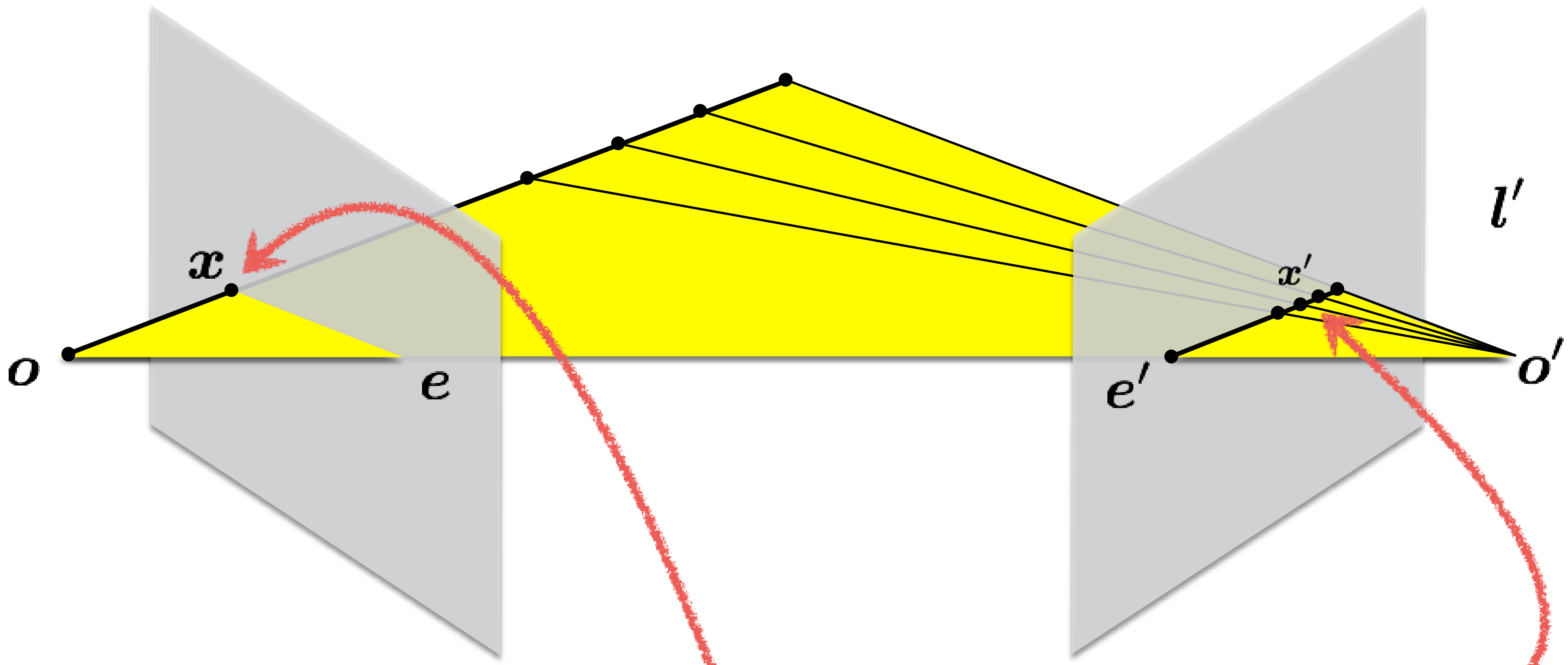
Task: Match point in left image to point in right image



Left image Right image

How would you do it?

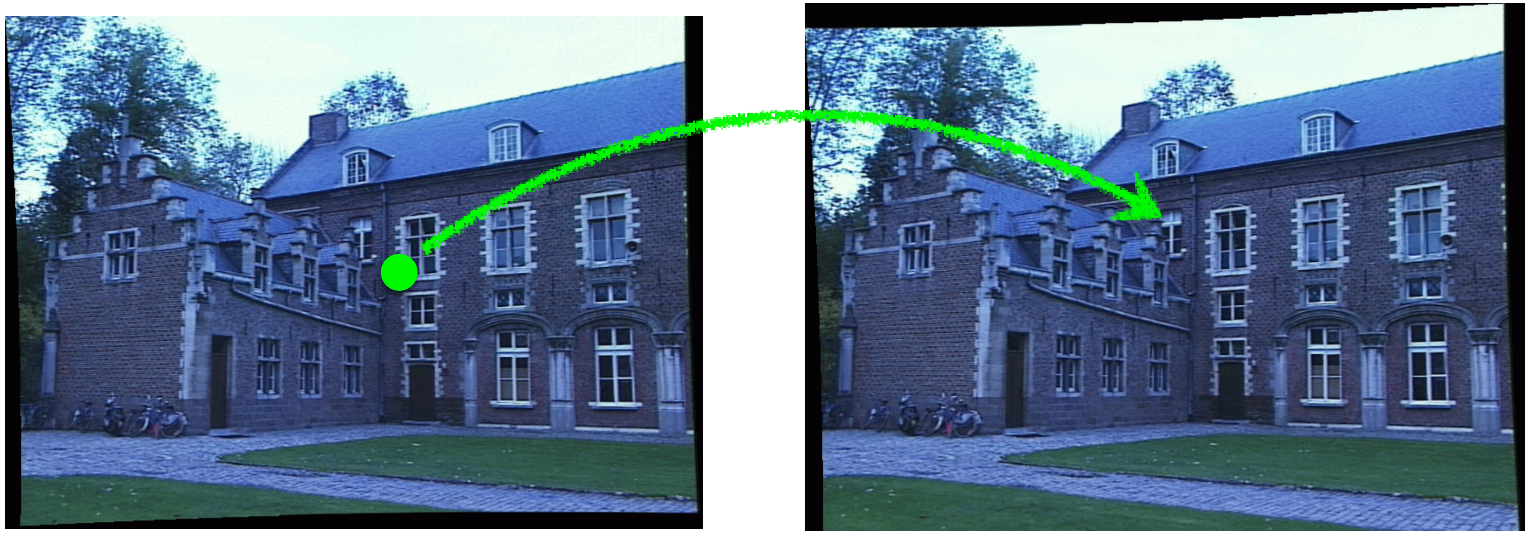
Epipolar constraint



Potential matches for $\boldsymbol{x}$ lie on the epipolar line $\boldsymbol{l}'$

The epipolar constraint is an important concept for stereo vision

Task: Match point in left image to point in right image



Left image

Right image

Want to avoid search over entire image

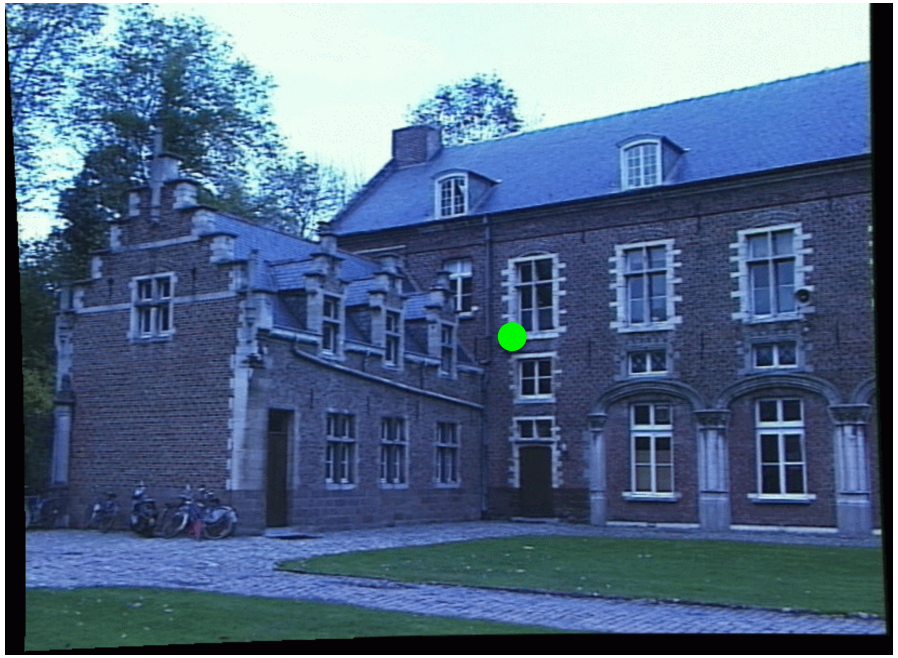
Epipolar constraint reduces search to a single line

How would you reconstruct 3D points?

Left image

Right image

How would you reconstruct 3D points?

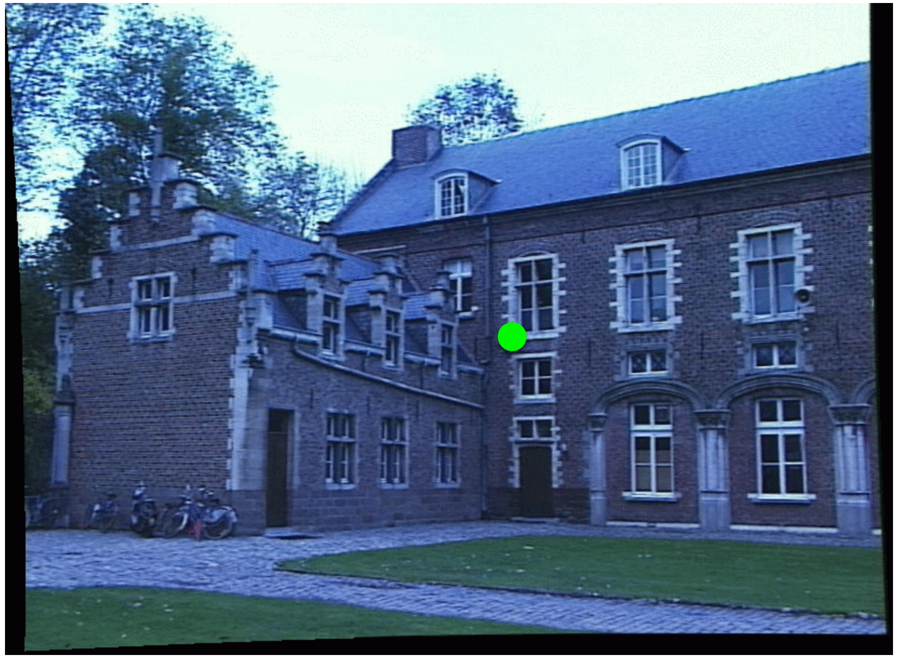


Left image

Right image

1. Select point in one image

How would you reconstruct 3D points?

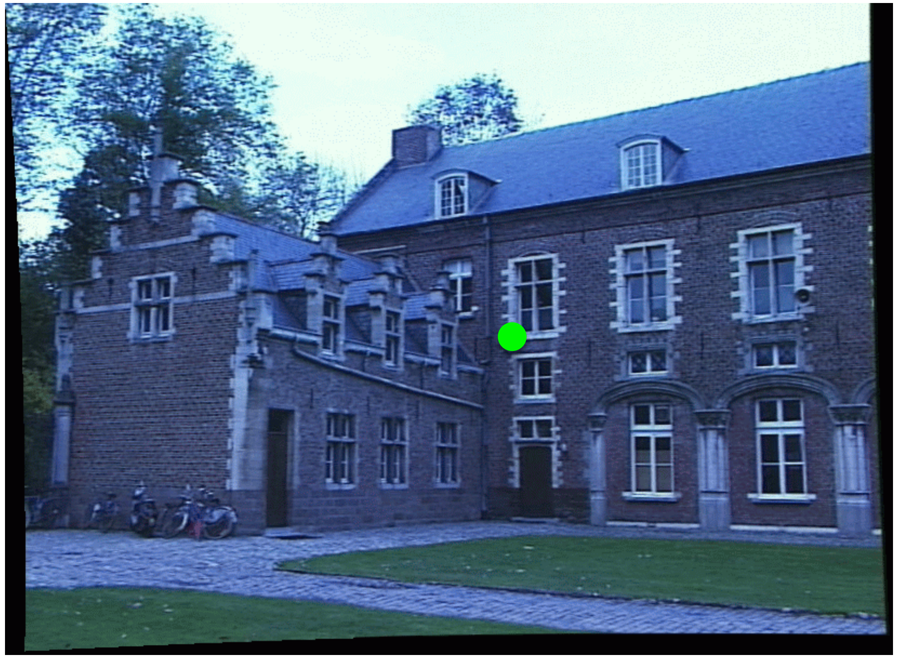


Left image

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?

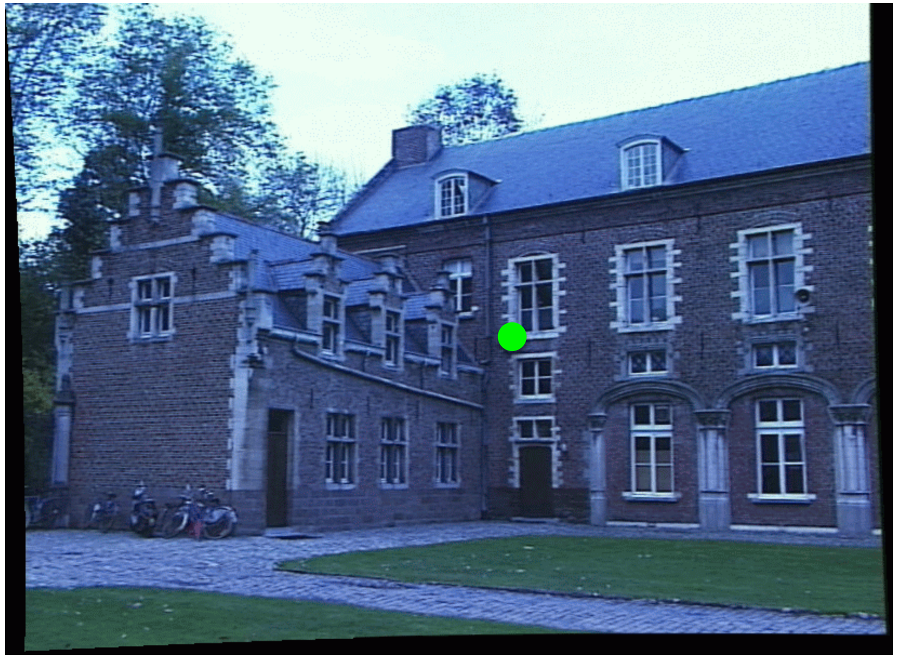


Left image

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?



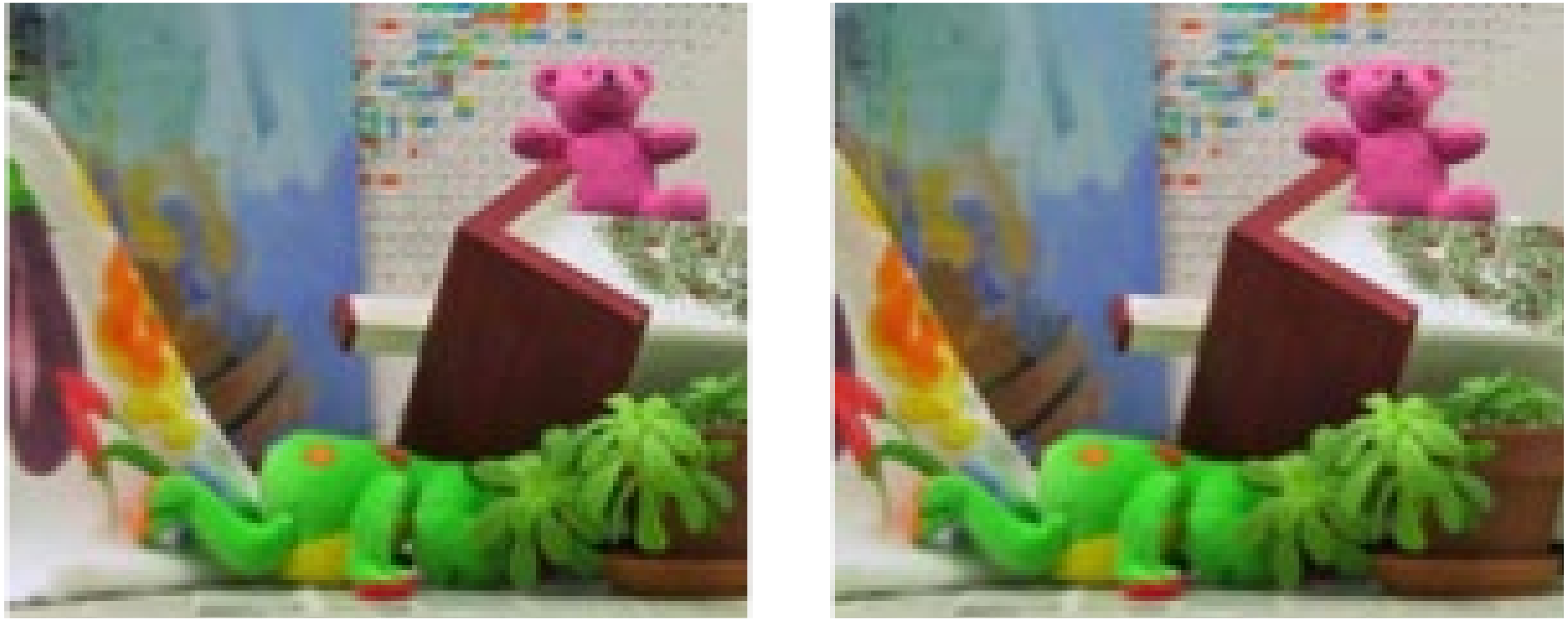
Left image

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?)

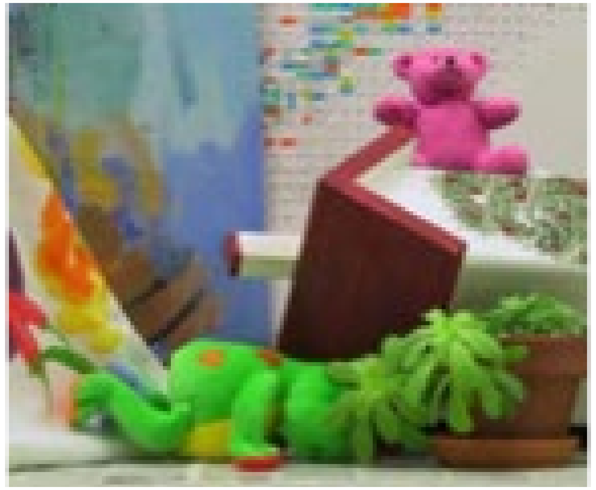
Stereo rectification

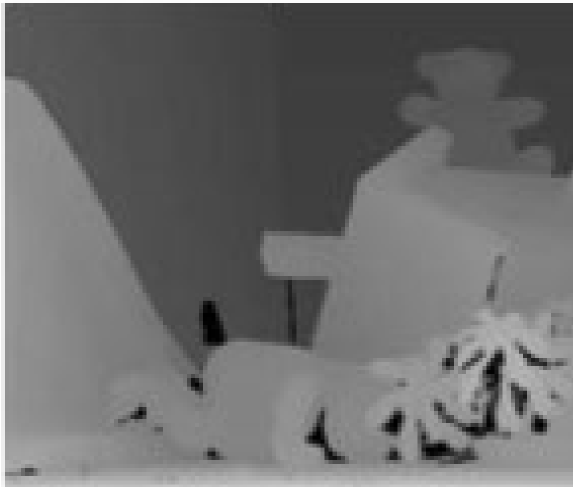
What's different between these two images?



Objects that are close move more or less?

The amount of horizontal movement is inversely proportional to …

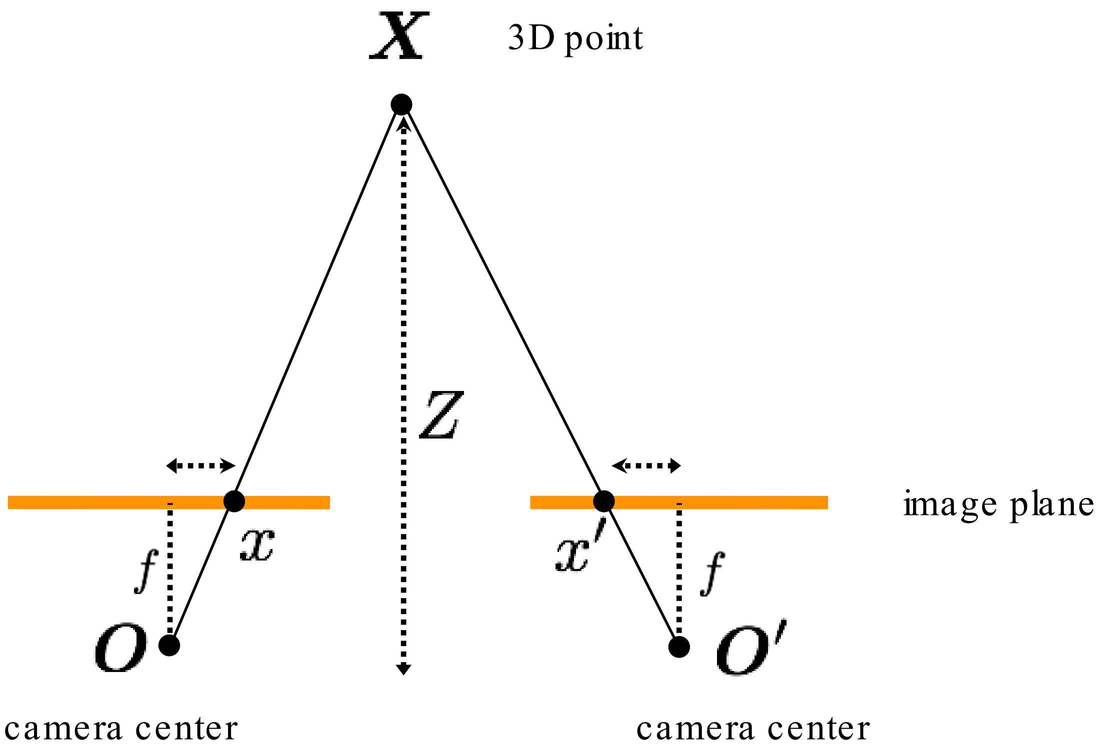




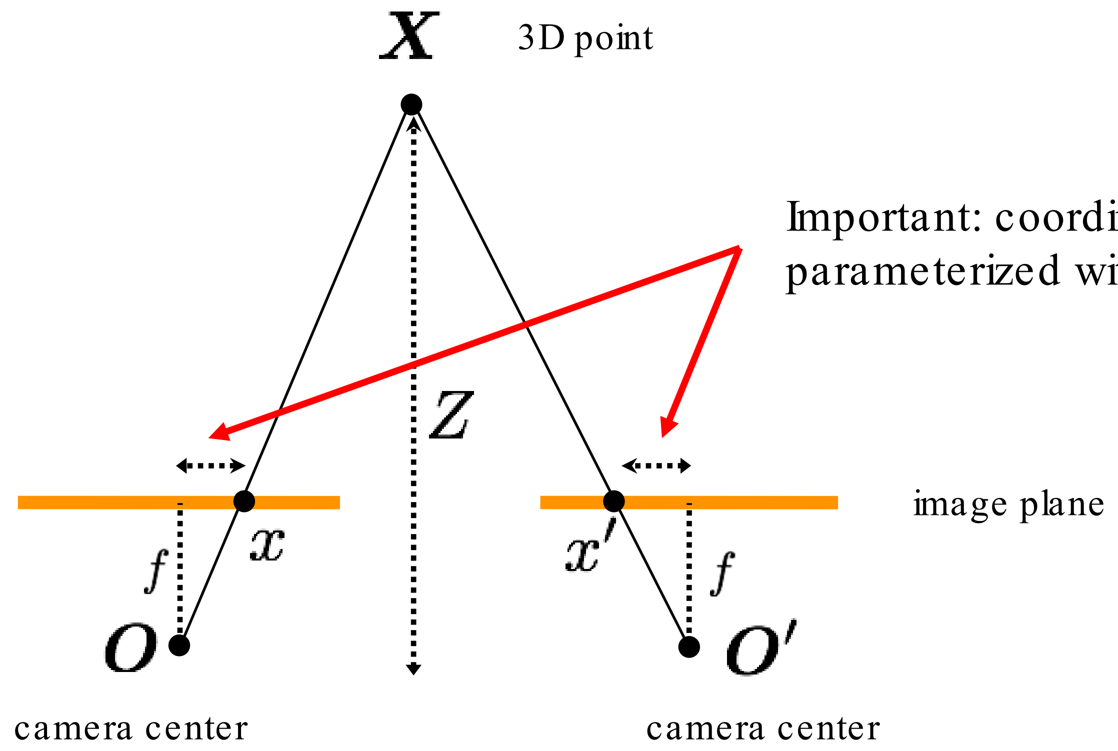
The amount of horizontal movement is inversely proportional to …

… the distance from the camera.

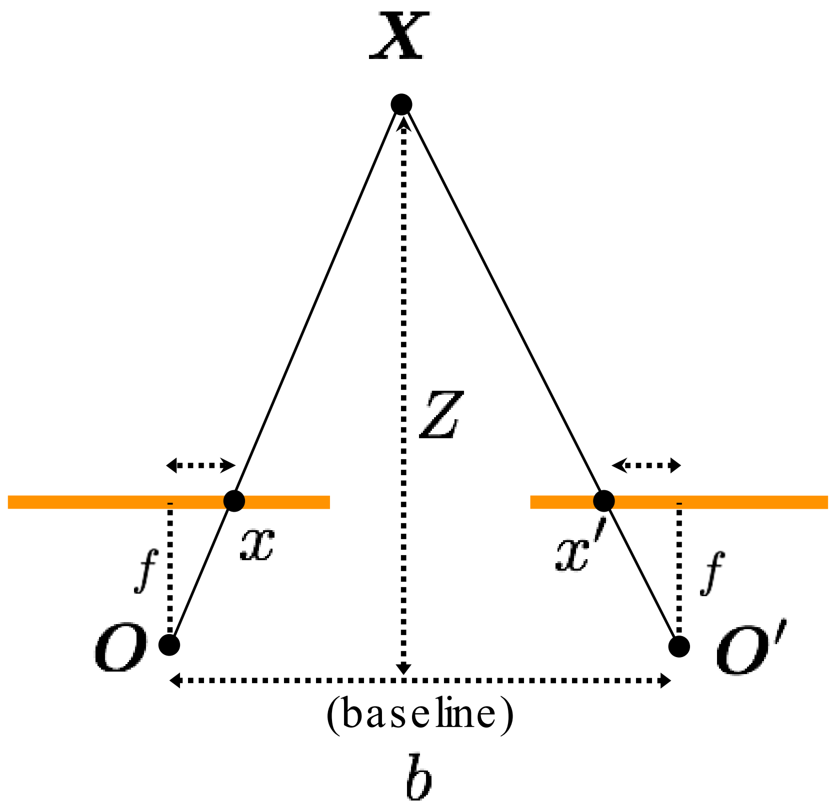
More formally…



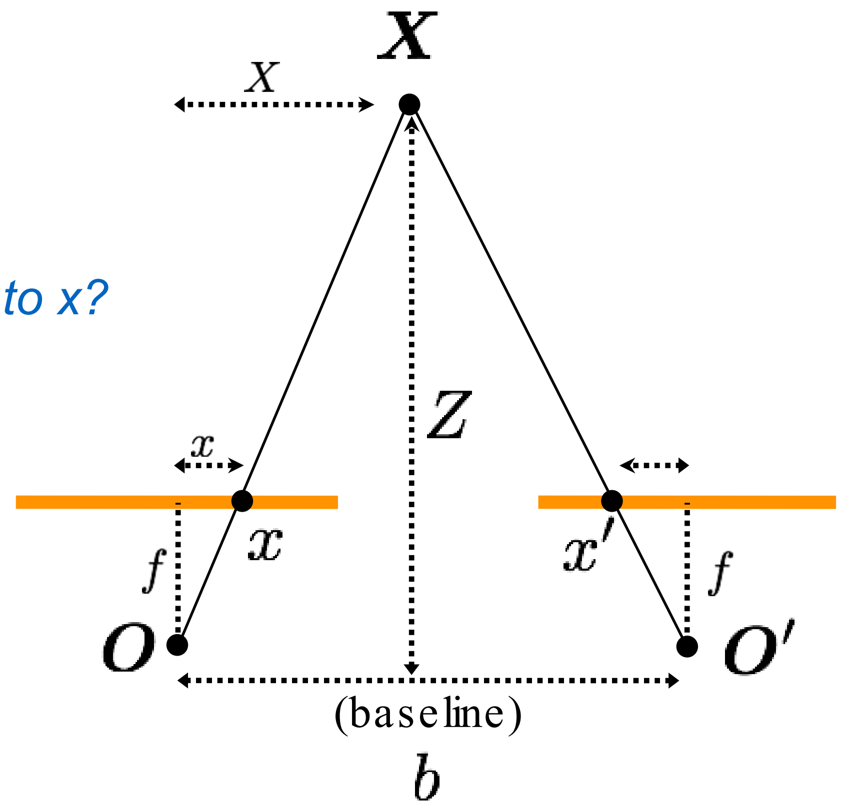
X
3D point
Z
image plane
x
x'
f
f
O
O'
camera center
camera center



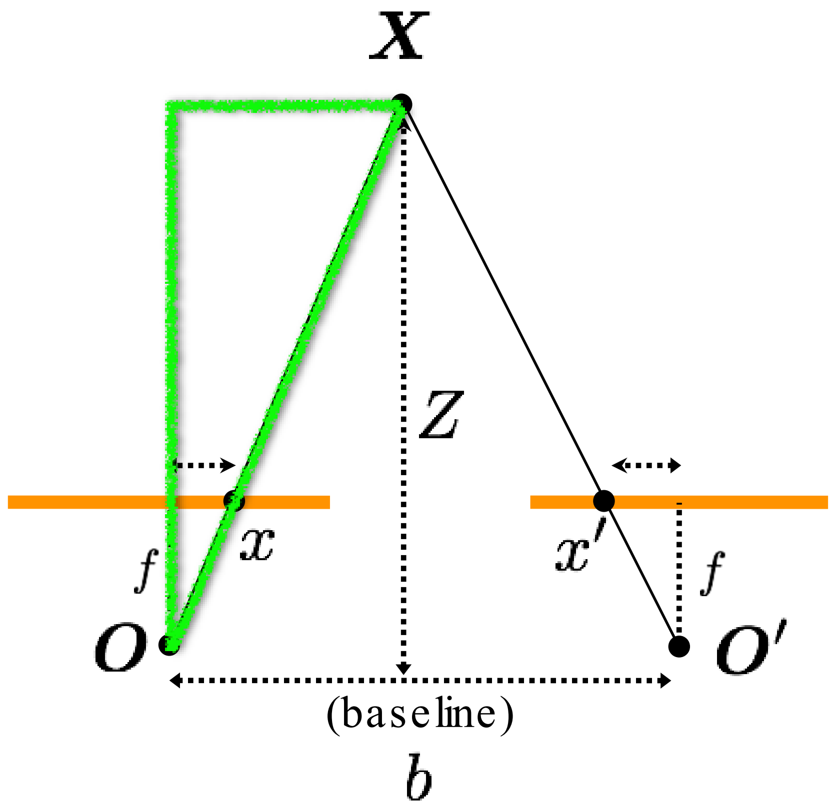
Important: coordinates **x** and **x'** are parameterized with respect to image center.


X
Z
x
x'
f
f
O
O'
(baseline)
b

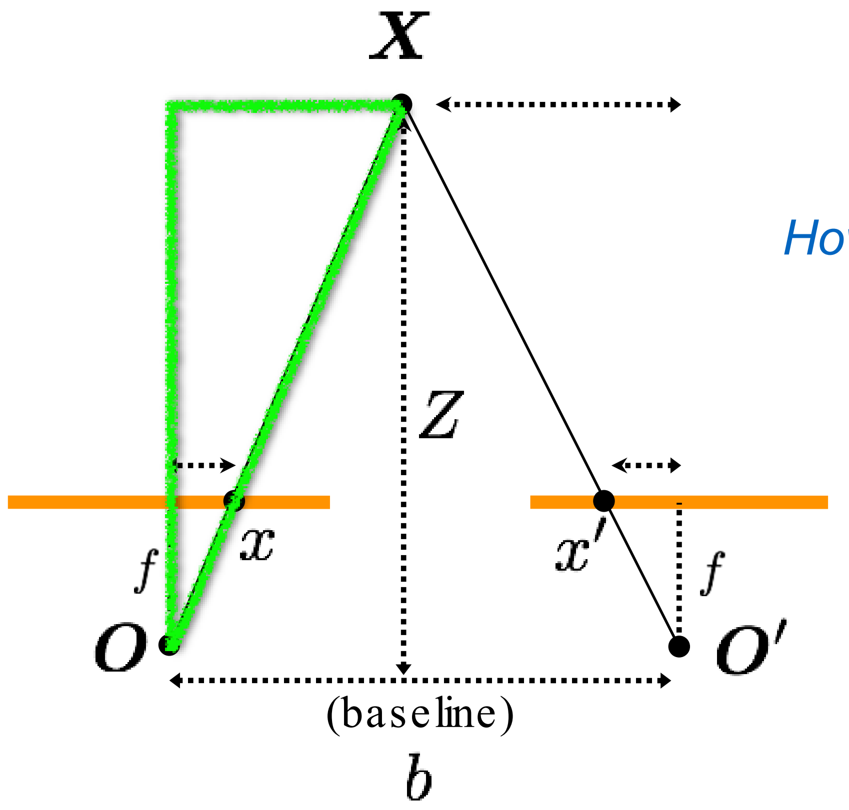
How is X related to x?



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

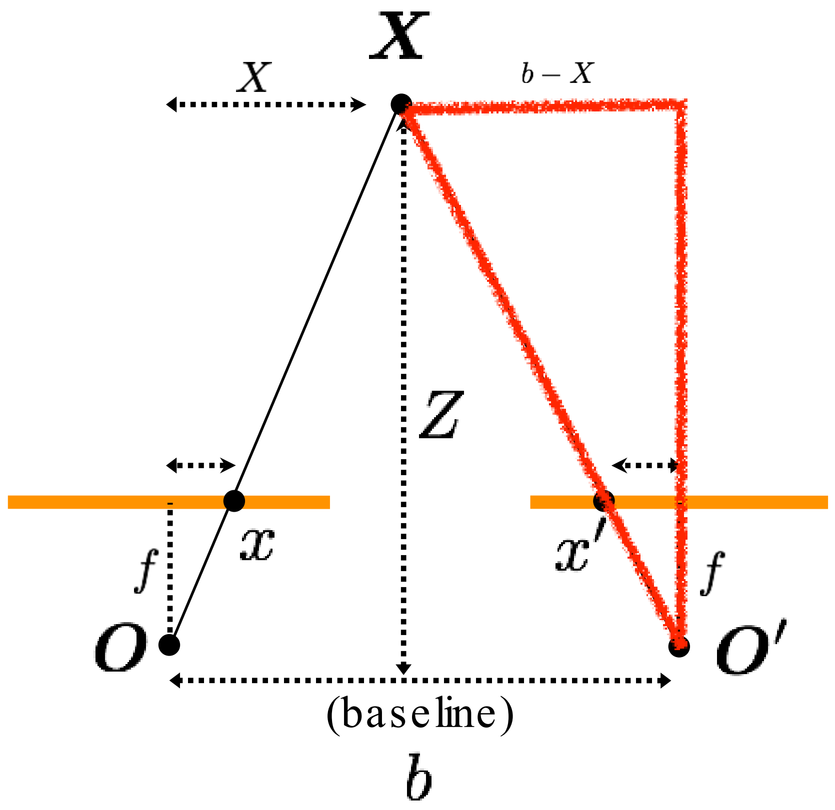


$$\frac{X}{Z} = \frac{x}{f}$$



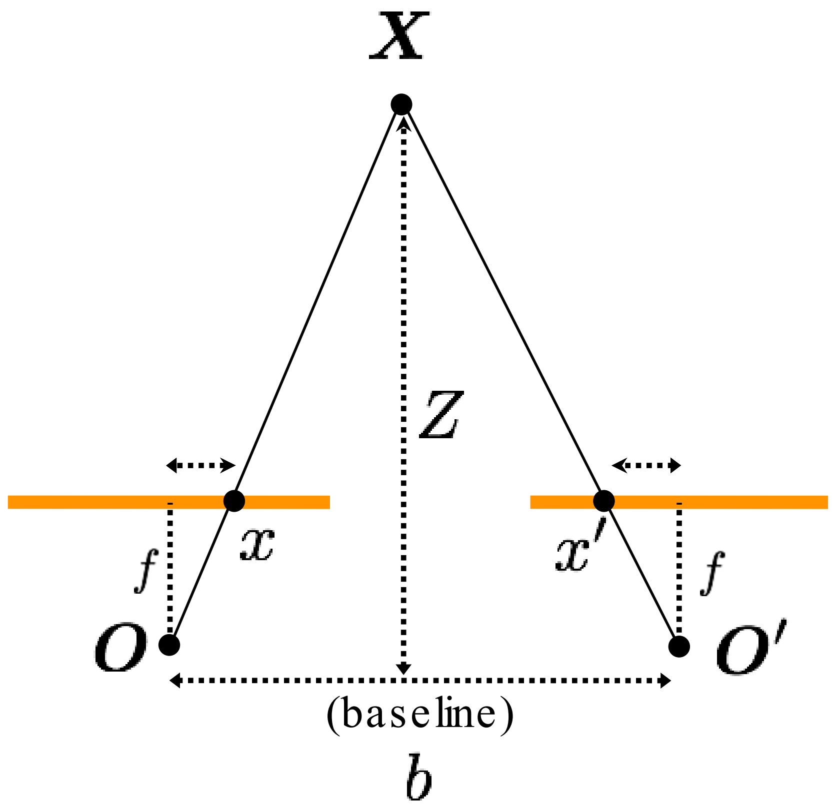
How is X related to x'?

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



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

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



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

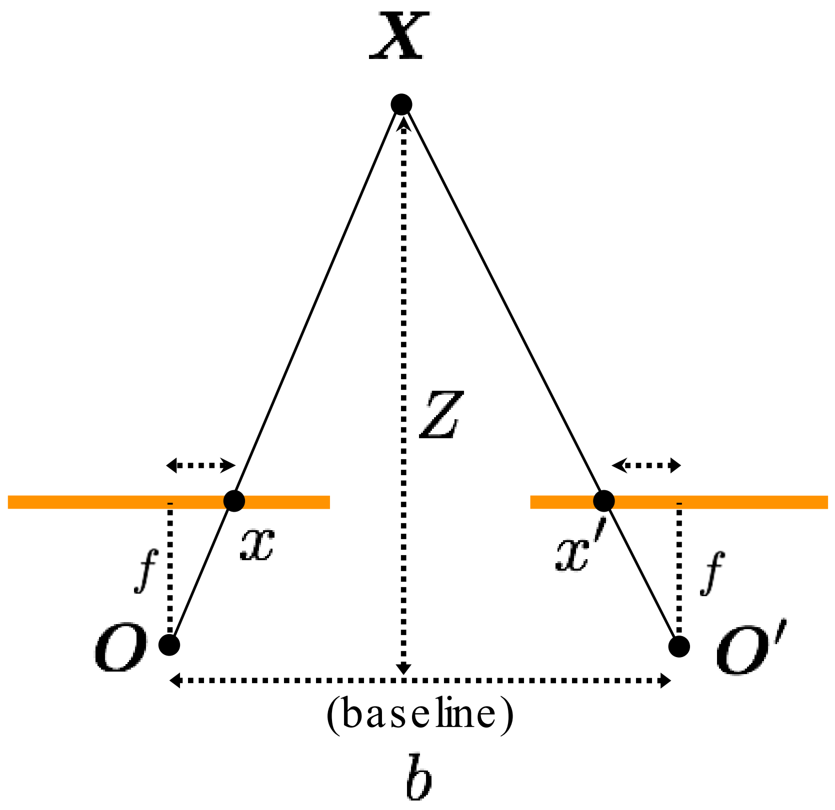
Disparity

$$d = x - x'$$

(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

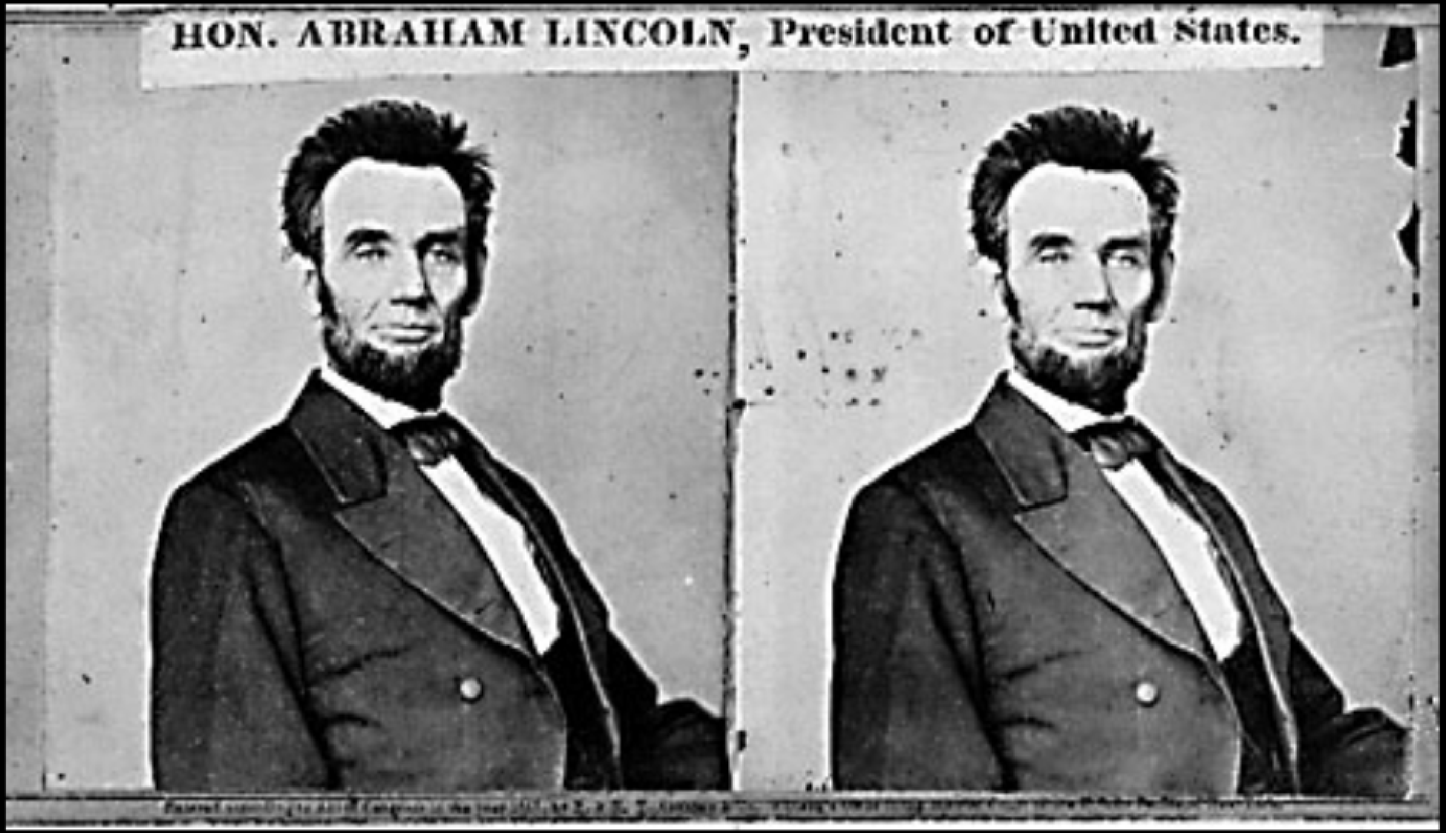
Nomad robot searches for meteorites in Antartica
http://www.frc.ri.cmu.edu/projects/meteorobot/index.html

Subaru
Eyesight system

Pre-collision
braking

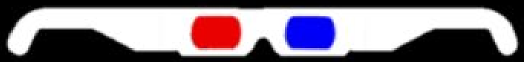
What other vision system uses disparity for depth sensing?

Stereoscopes: A 19th Century Pastime

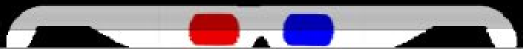

HON. ABRAHAM LINCOLN, President of United States.

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

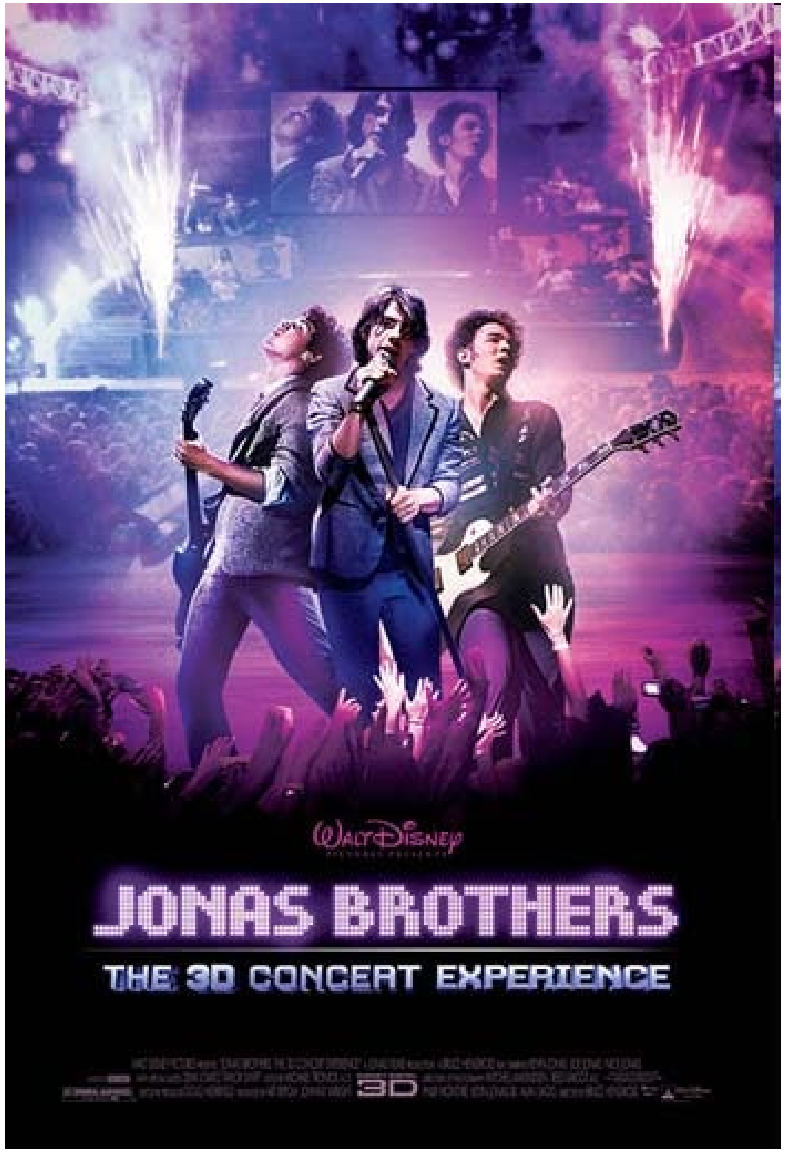
Teesta suspension bridge-Darjeeling, India



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

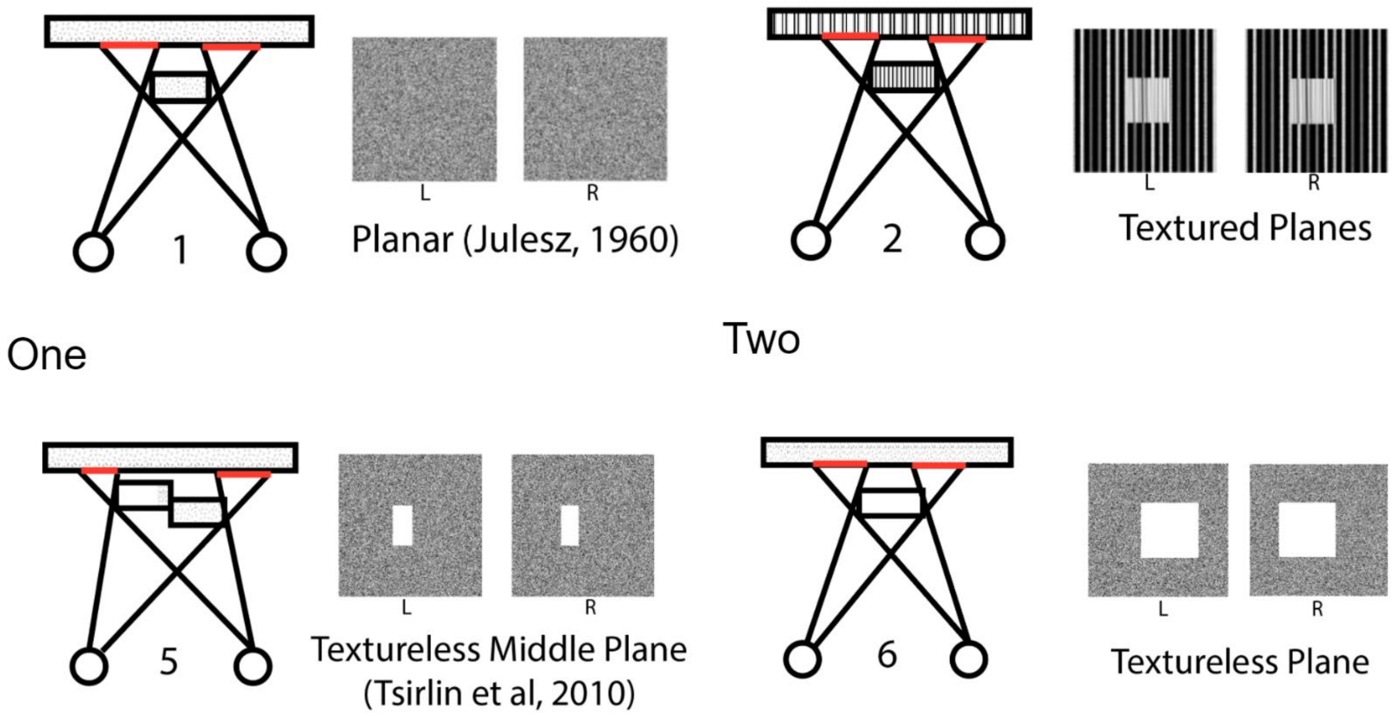


This is how 3D movies work



Simple stereoscope

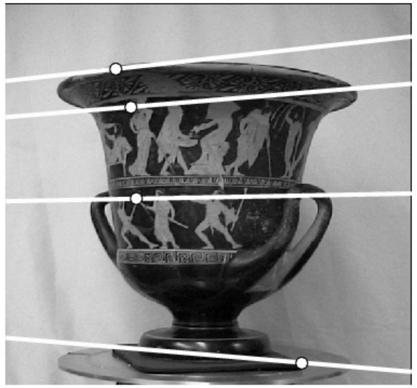
Google cardboard



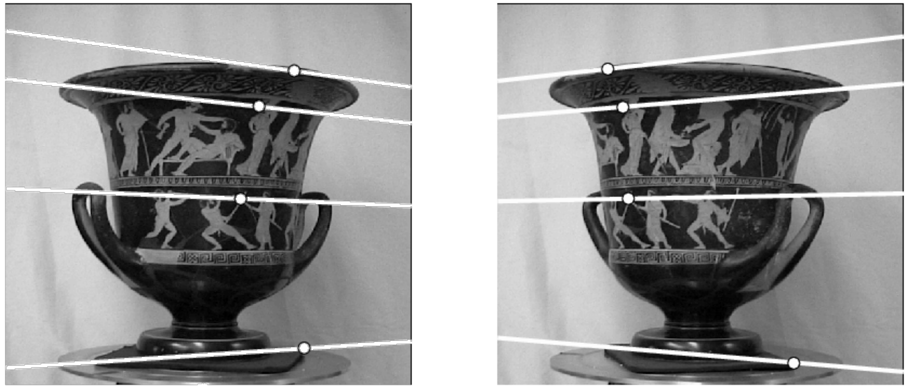
Fun patterns: random dot stereograms

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

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

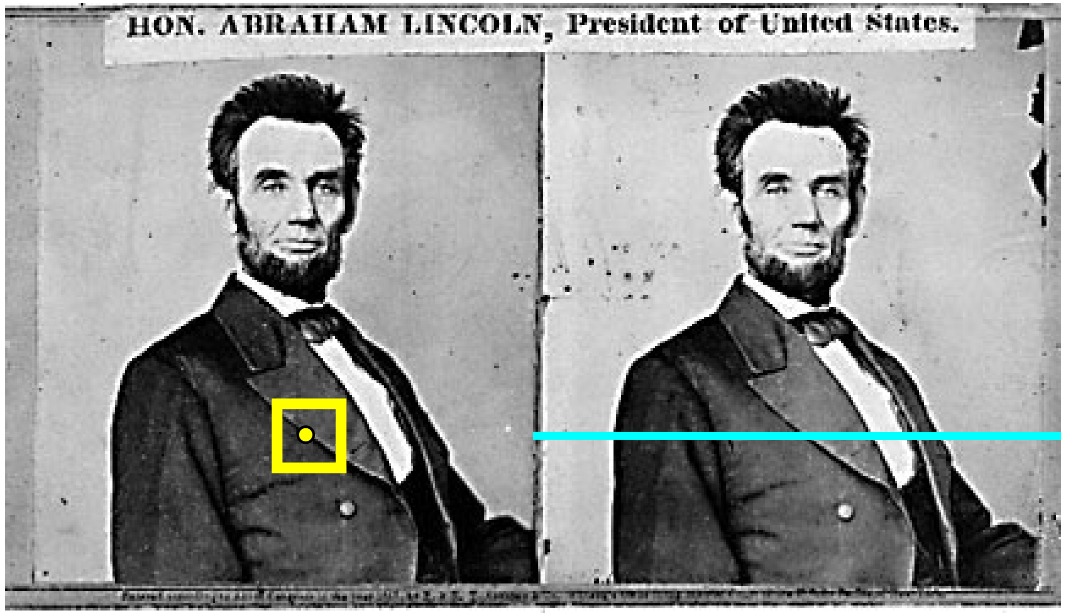


So can I compute depth using disparity from any two images of the same object?

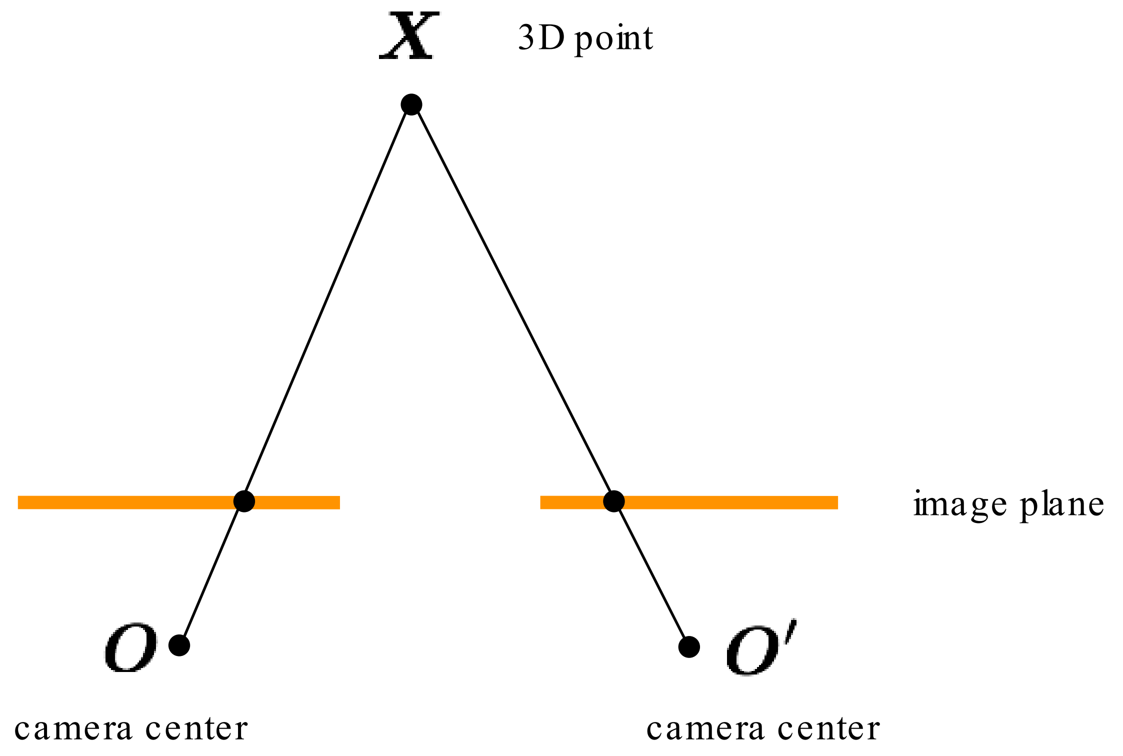


1. Need sufficient baseline

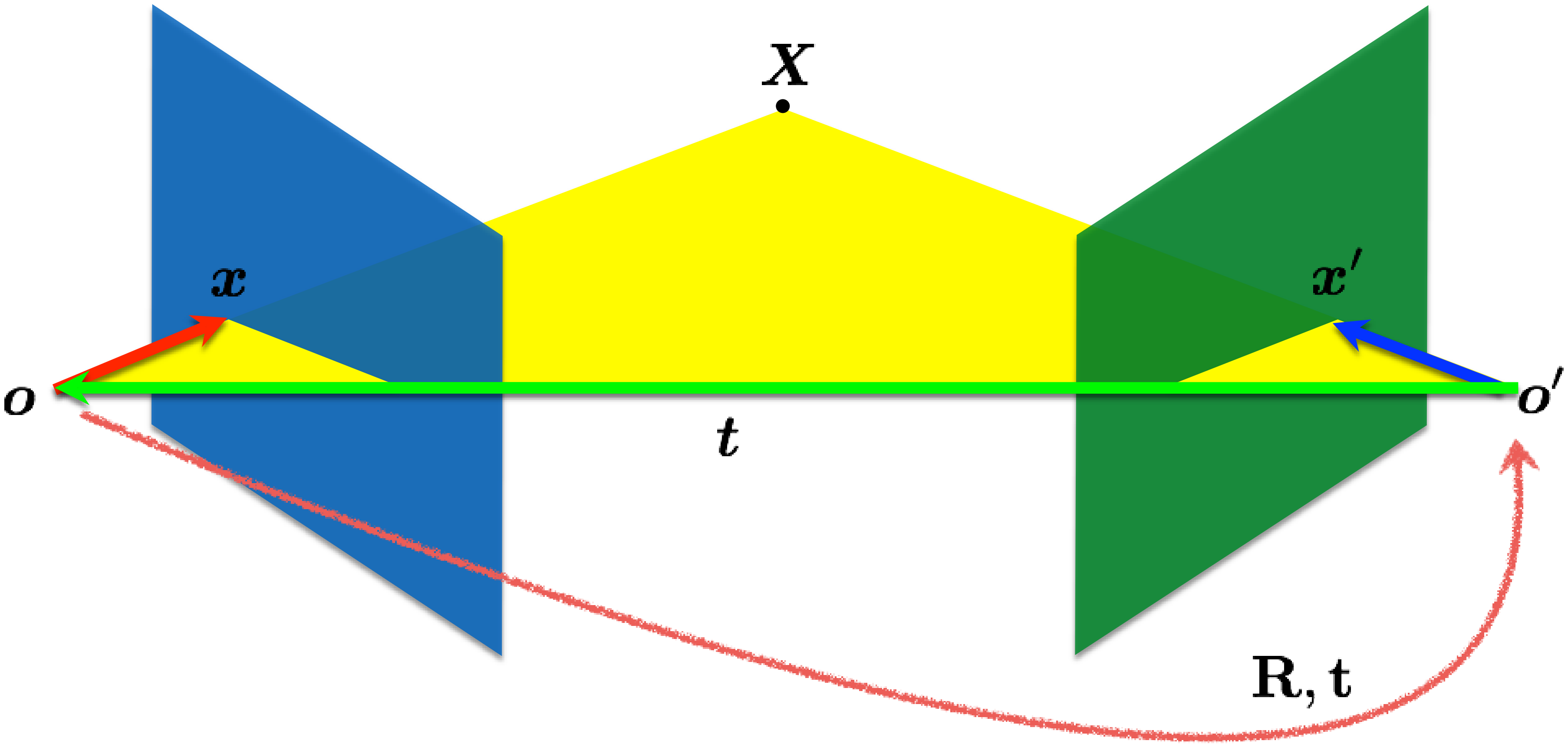
2. Images need to be 'rectified' first (make epipolar lines horizontal)



How can you make the epipolar lines horizontal?

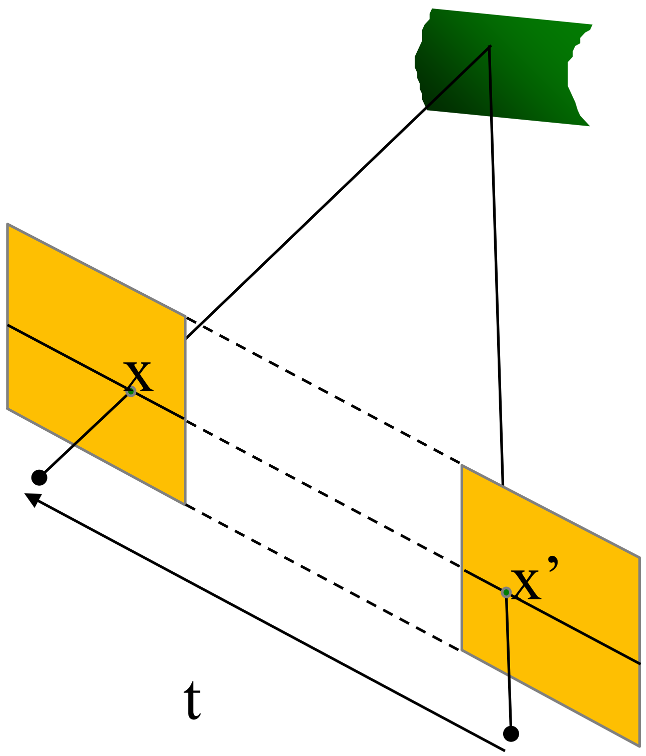


What's special about these two cameras?

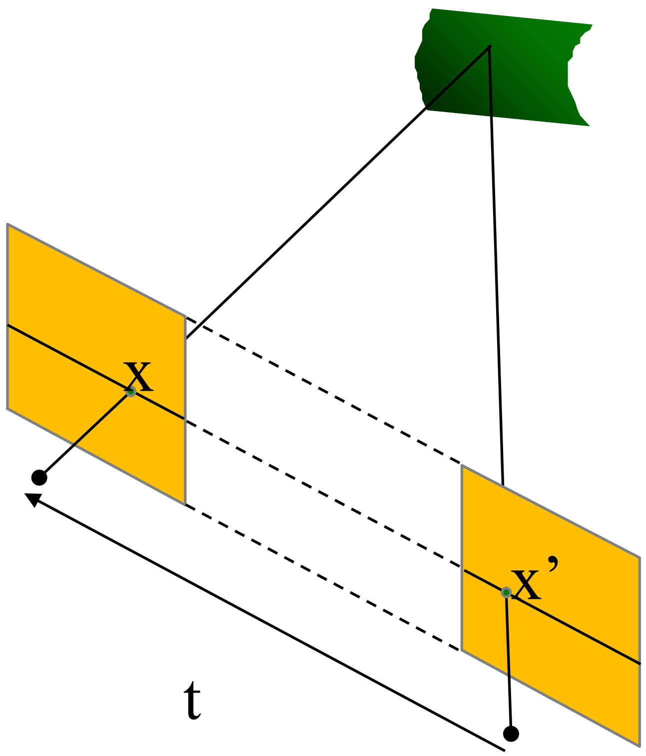


$$x' = \mathbf{R}(x - t)$$

When are epipolar lines horizontal?

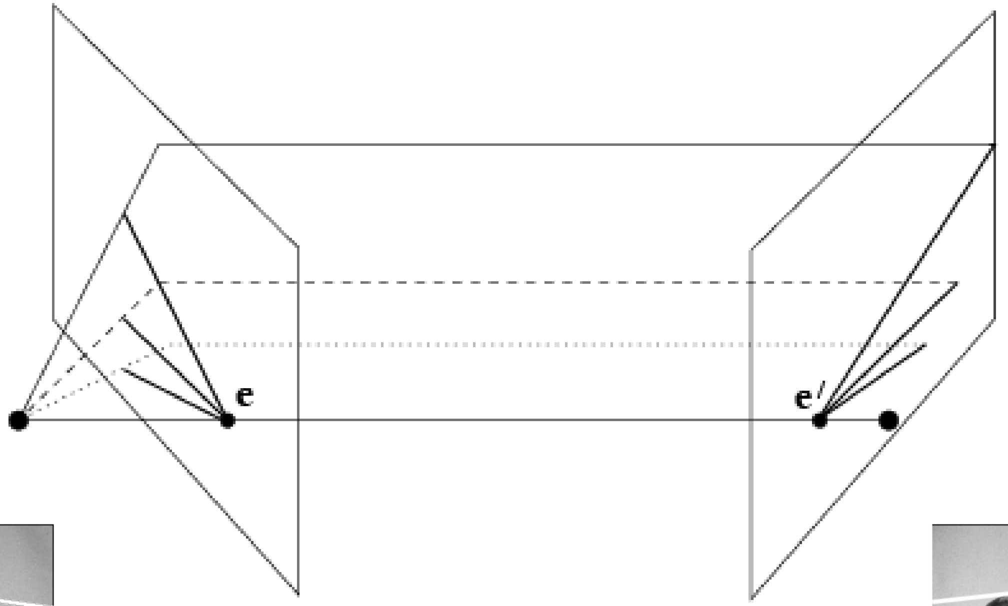


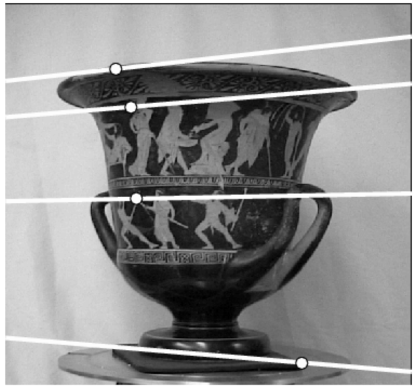
When are epipolar lines horizontal?



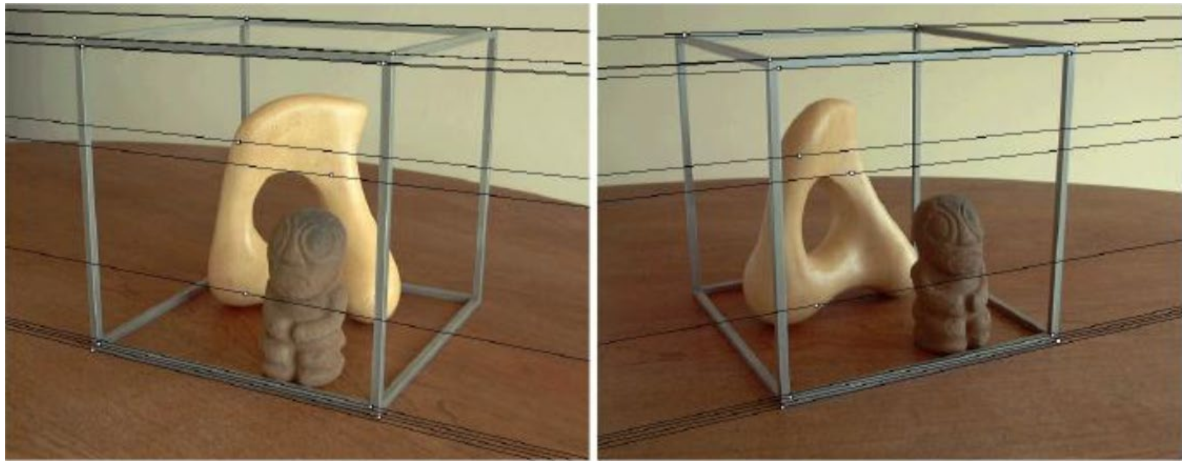
When this relationship holds:

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

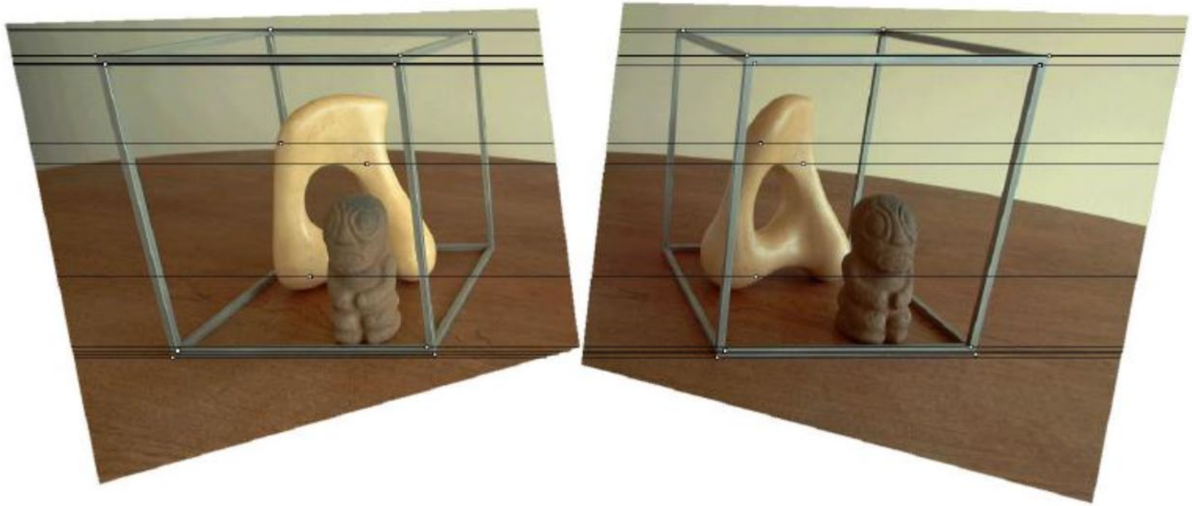


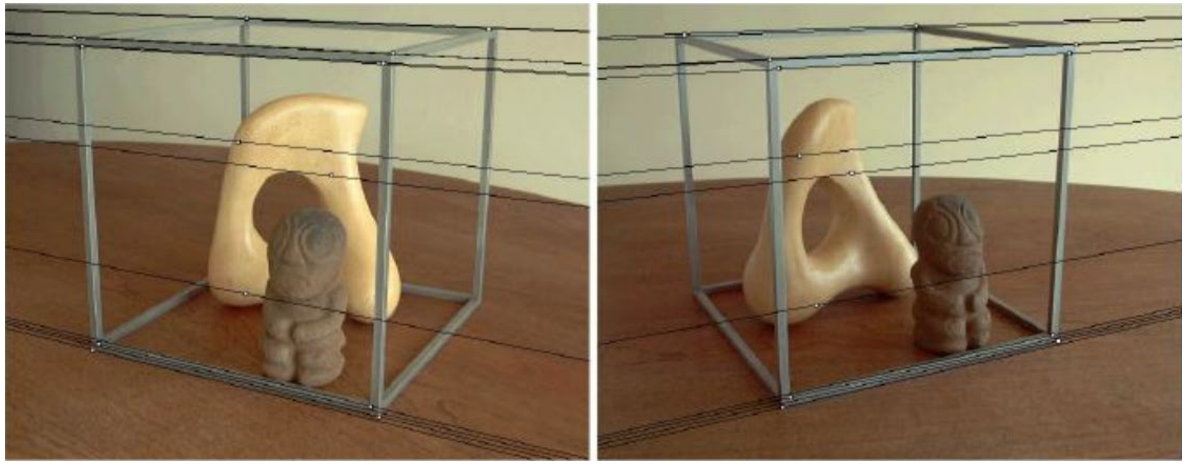


It’s hard to make the image planes exactly parallel

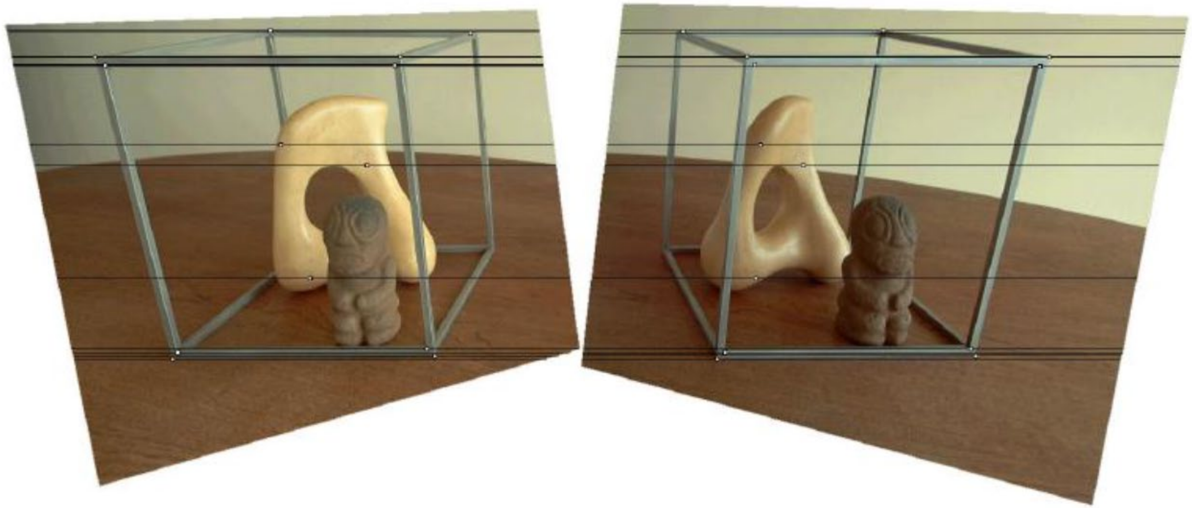


How can you make the epipolar lines horizontal?



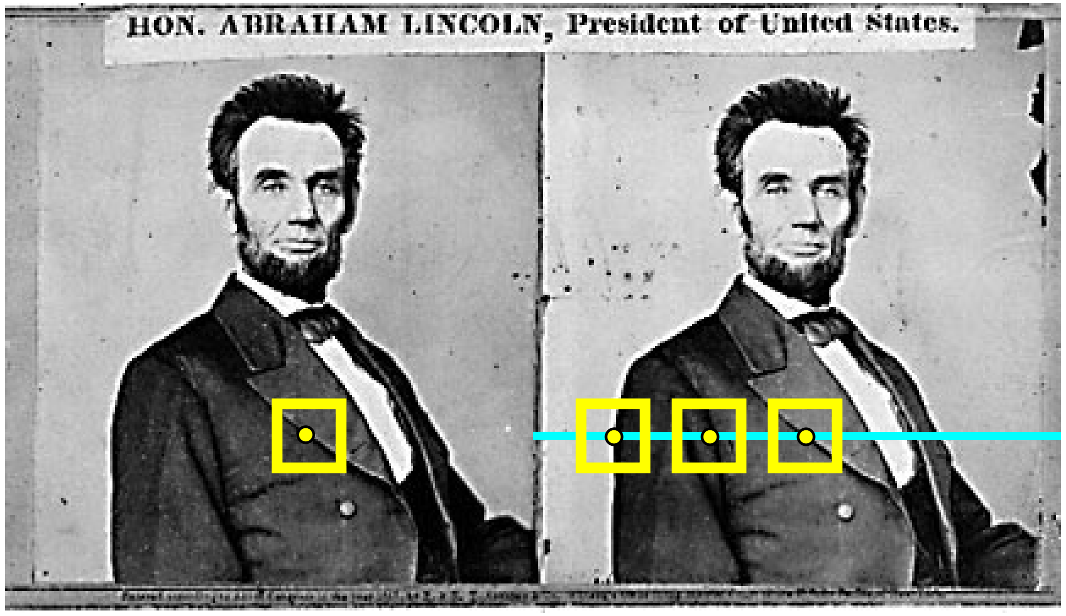


Use stereo rectification



Stereo matching

Depth Estimation via Stereo Matching



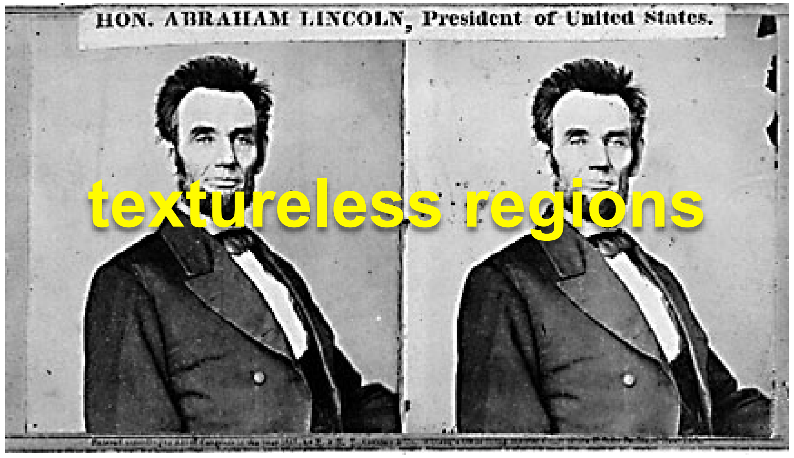
1. Rectify images
   (make epipolar lines horizontal)
2. For each pixel
   a. Find epipolar line
   b. Scan line for best match
   c. Compute depth from disparity

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

How would you do this?

When are correspondences difficult?

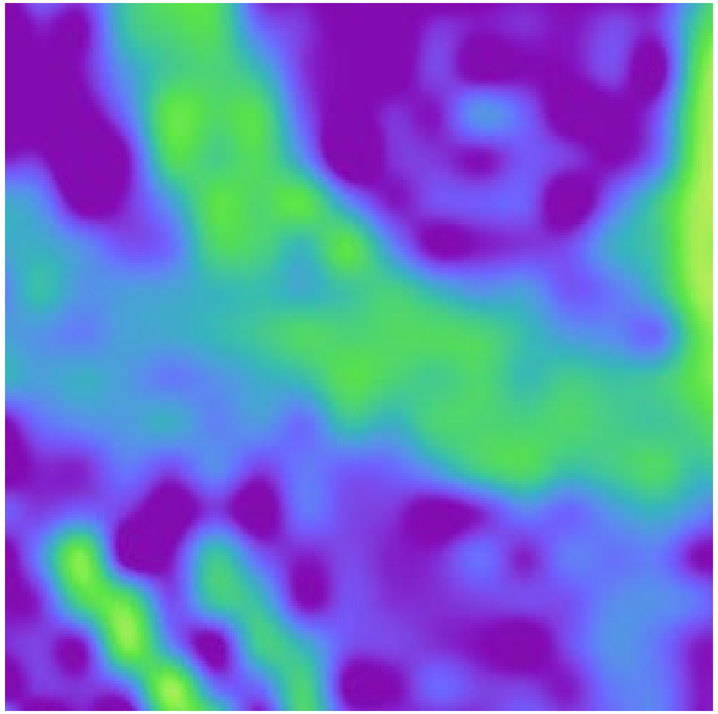
When are correspondences difficult?



Depth discontinuities

What is the problem here?

One of two input images



Depth from disparity

Groundtruth depth

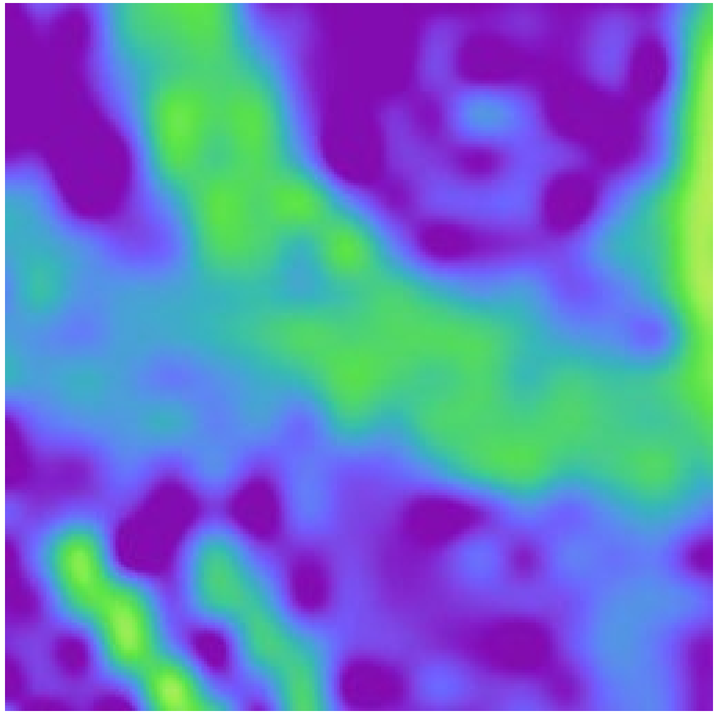
Depth discontinuities

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

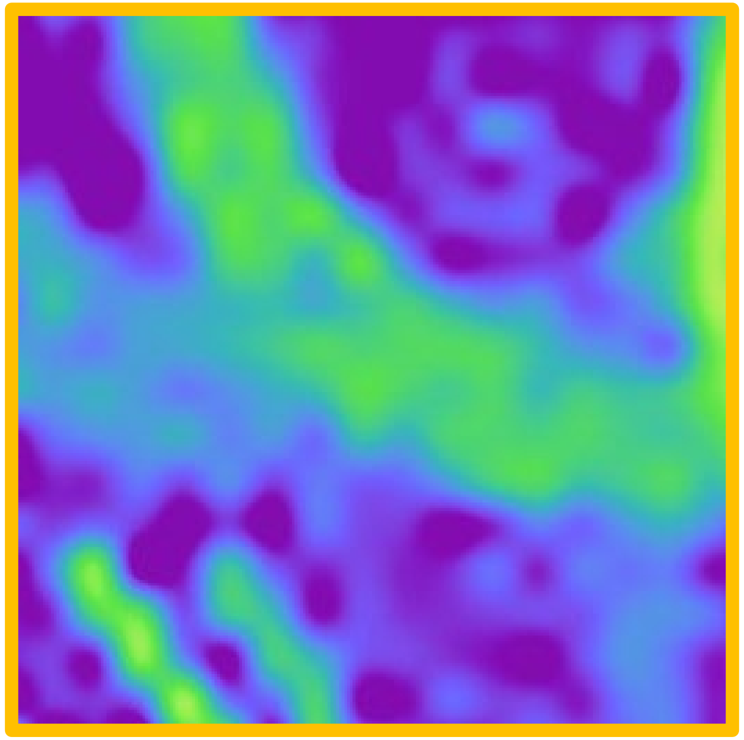
Groundtruth depth

Edge-aware depth denoising

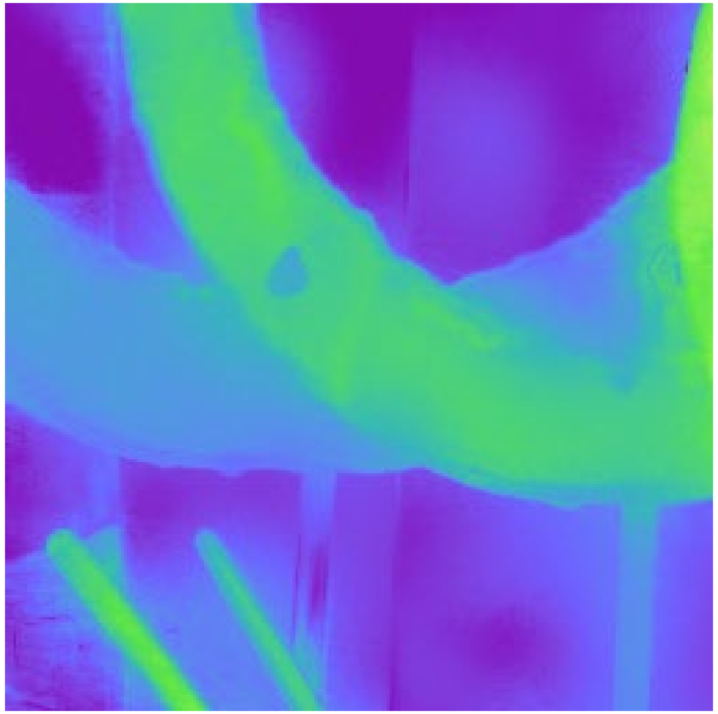
$$A_{p(col)} = \frac{1}{k(p(col))} \sum_{p' \in \Omega} g_d(|p - p'|) \, g_r\left(\boldsymbol{F}_{p(col)} - \boldsymbol{F}_{p'(col)}\right) A_{p'(col)}$$

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

One of two input images



Depth from disparity

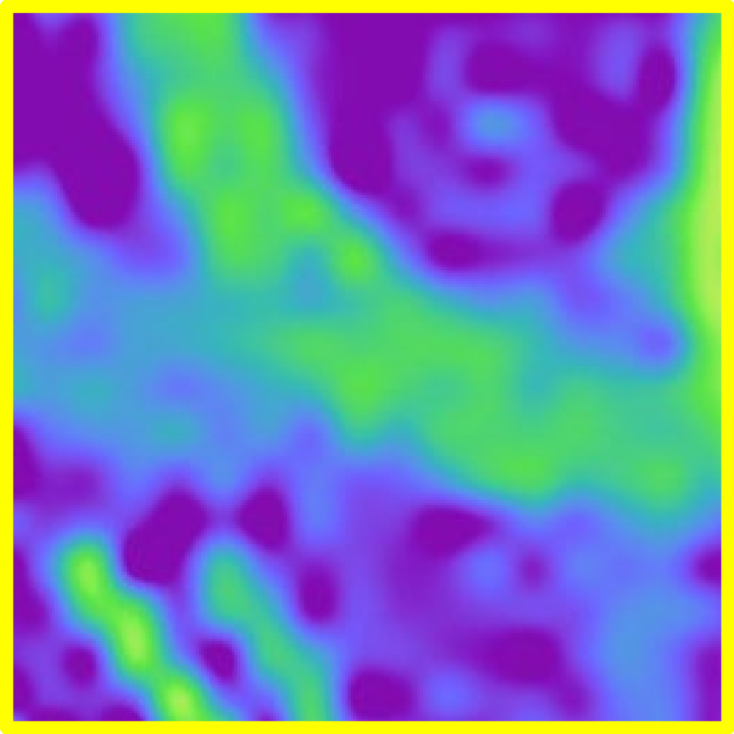


Guided filtering

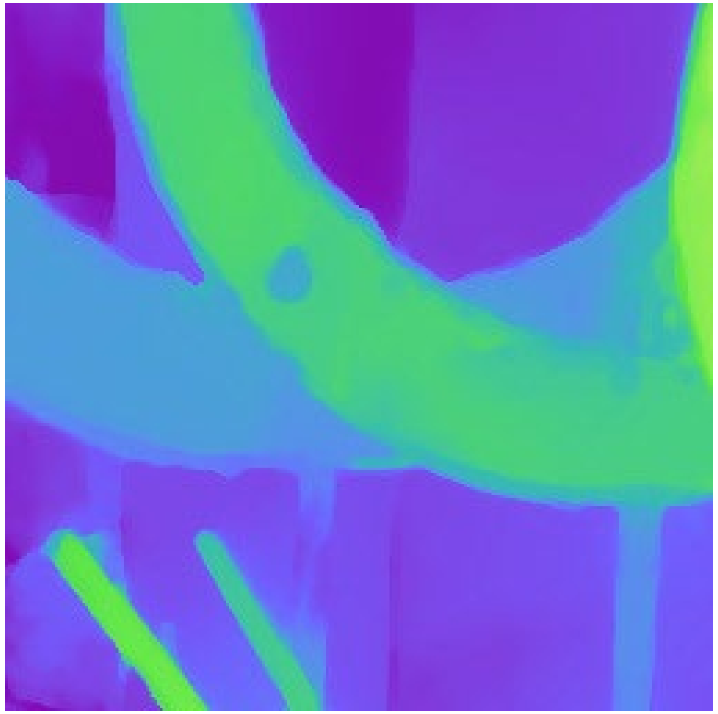
Fast bilateral solver

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

One of two input images



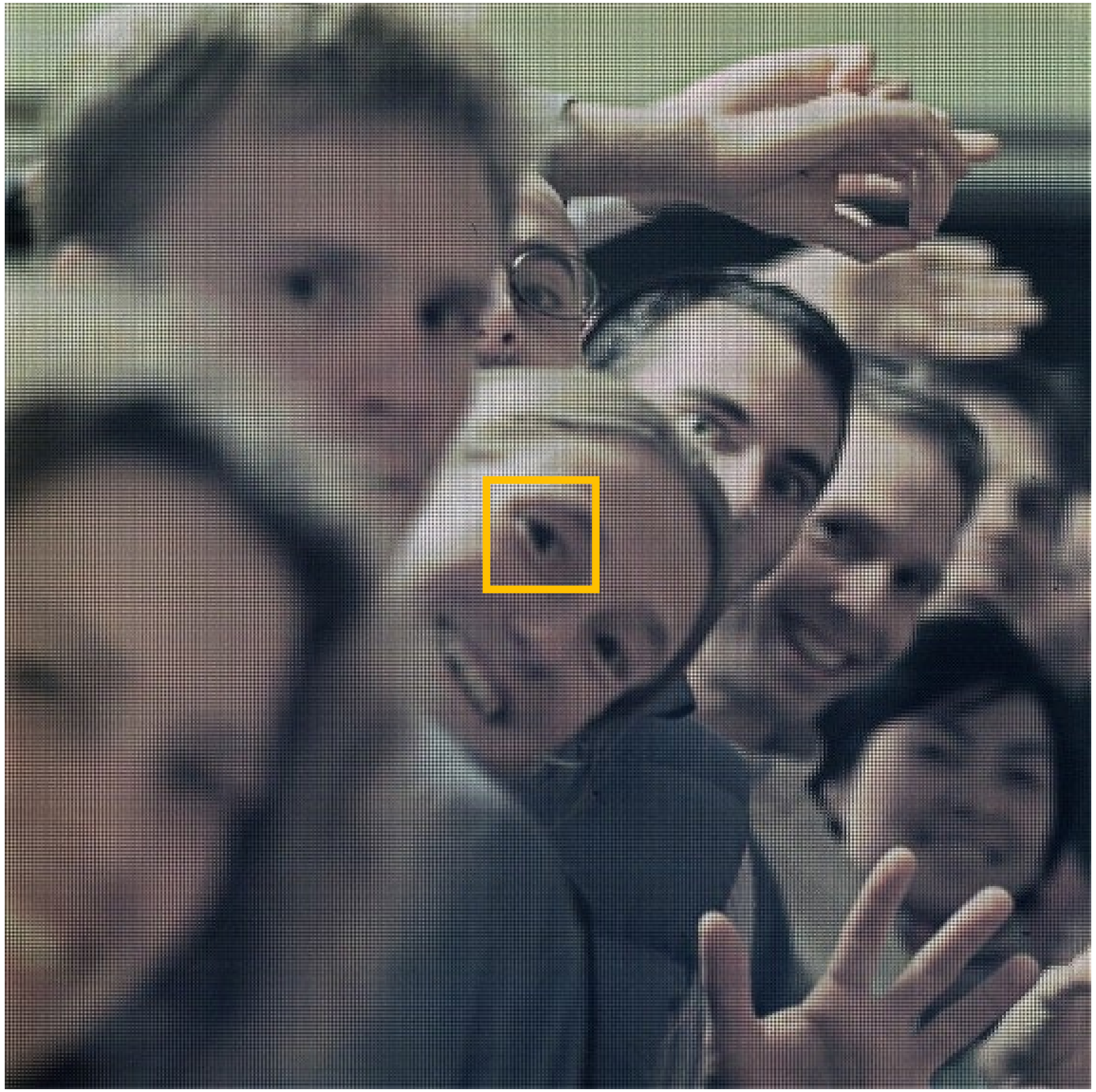
Depth from disparity

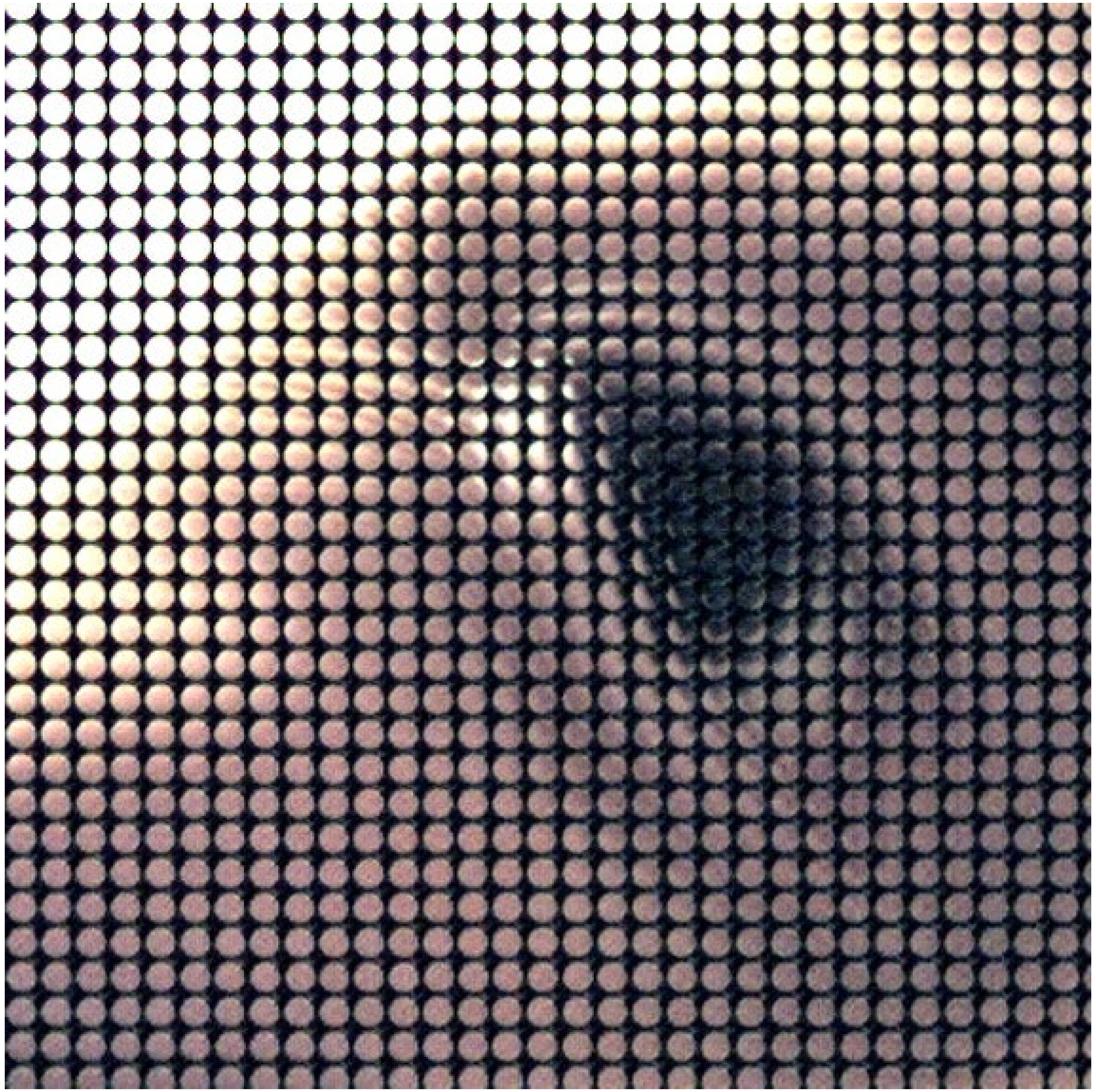


Bilateral stereo matching

Disparity and lightfields

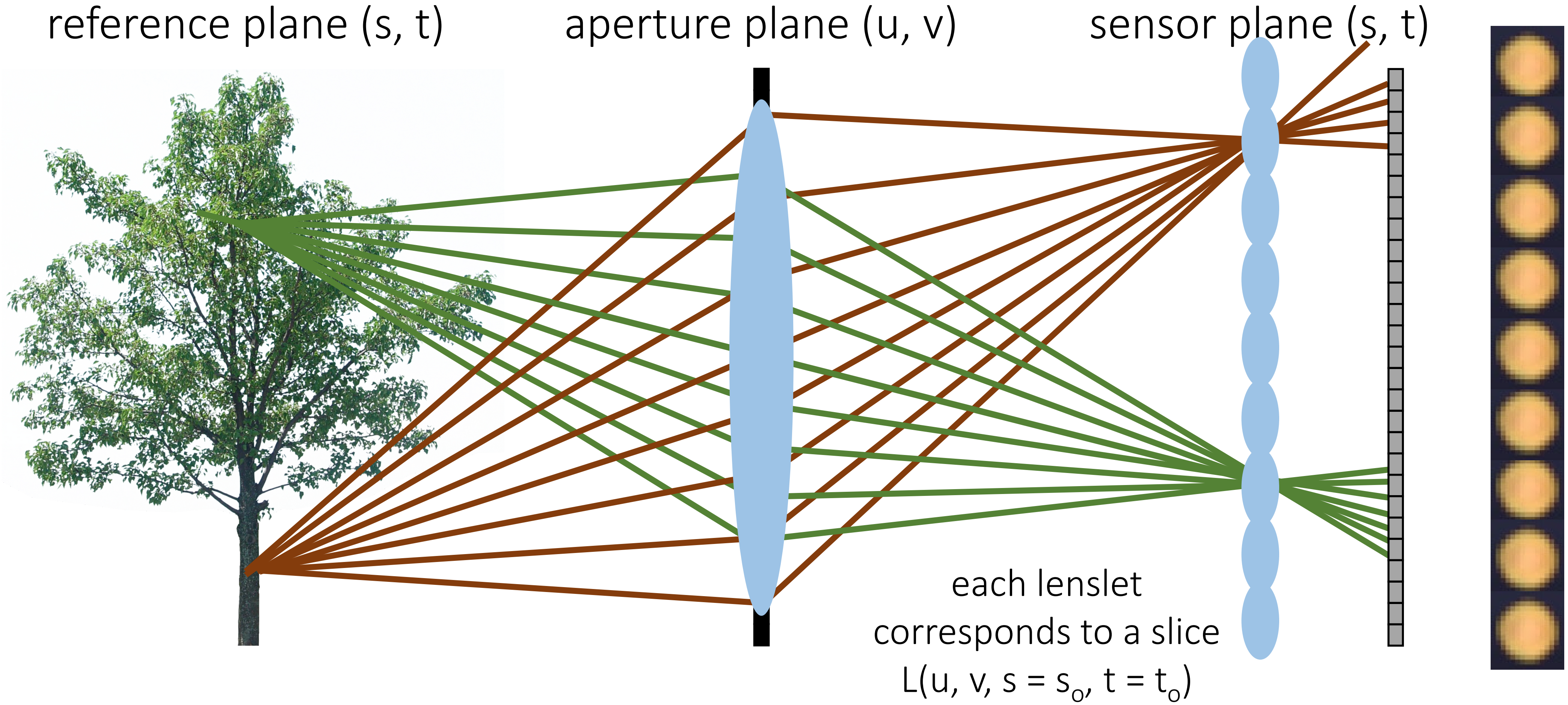
Reminder: a plenoptic “image”





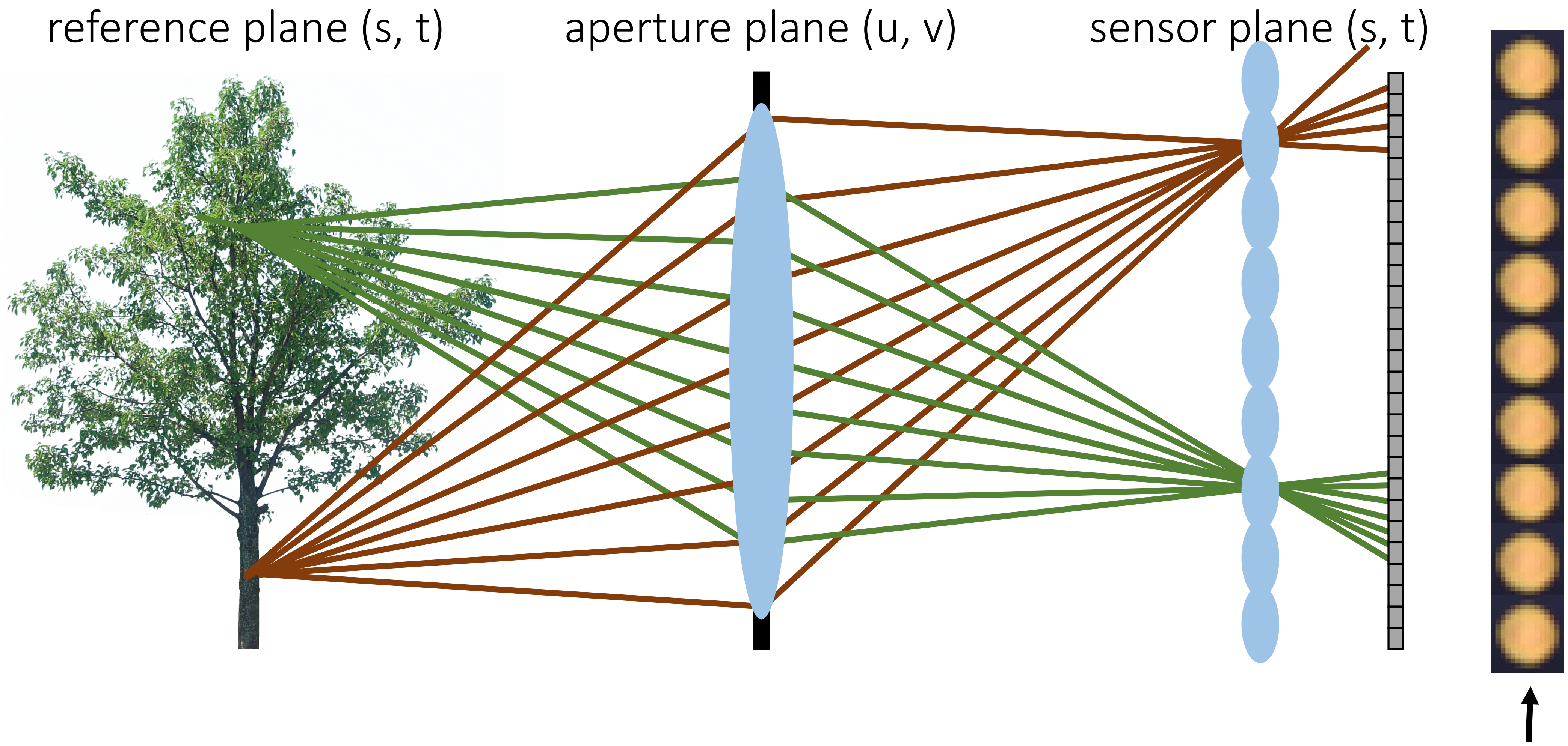
What are these circles?

Reminder: a plenoptic camera



Lightfield L(u, v, s, t)

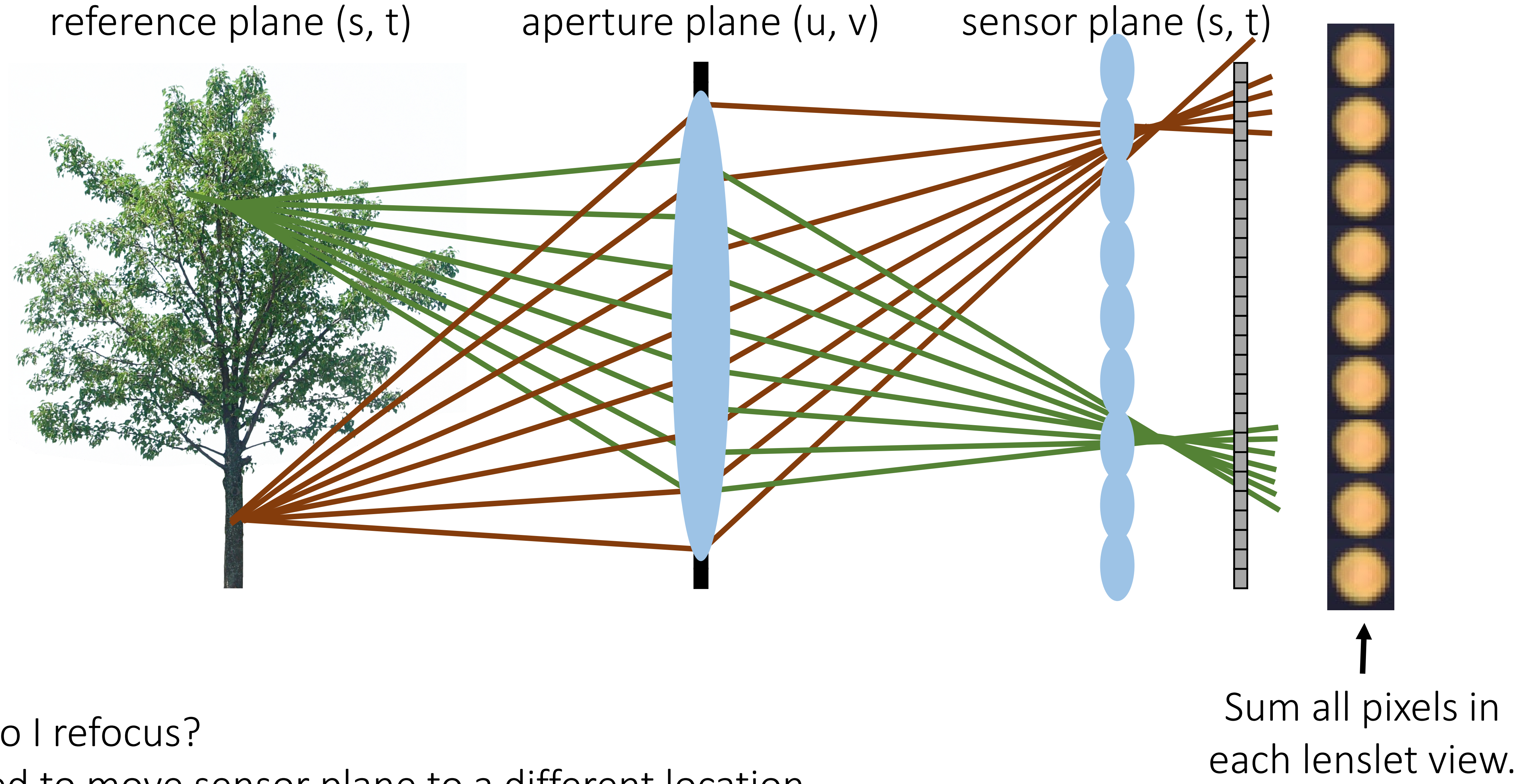
Reminder: form lens image



Sum all pixels in each lenslet view.

How do I refocus?

Reminder: form lens image



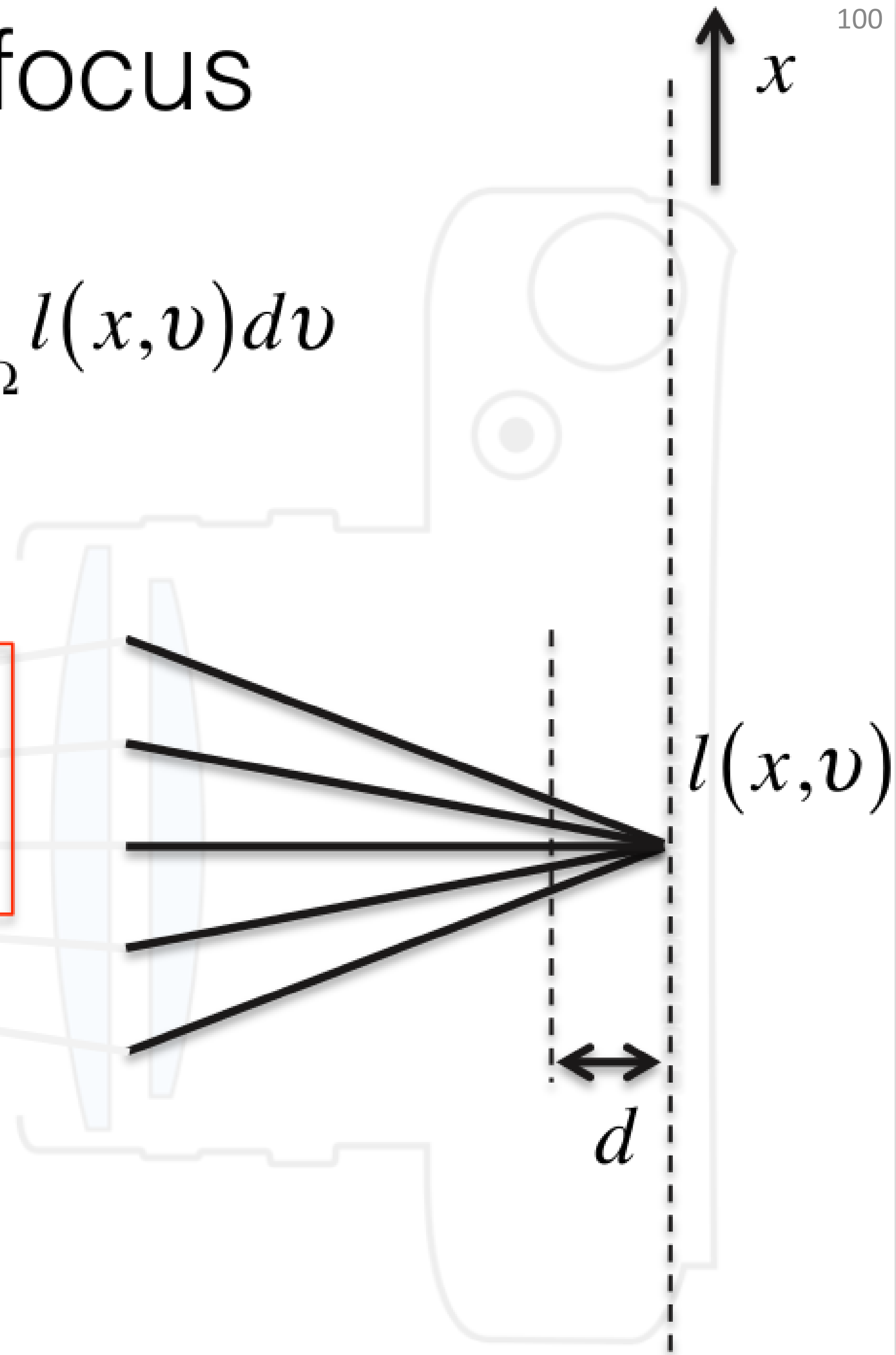
How do I refocus?

- Need to move sensor plane to a different location.

Understanding Refocus

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

$$\boxed{i_d(x) = \int_{\Omega} l(x+dv, v)\,dv}$$

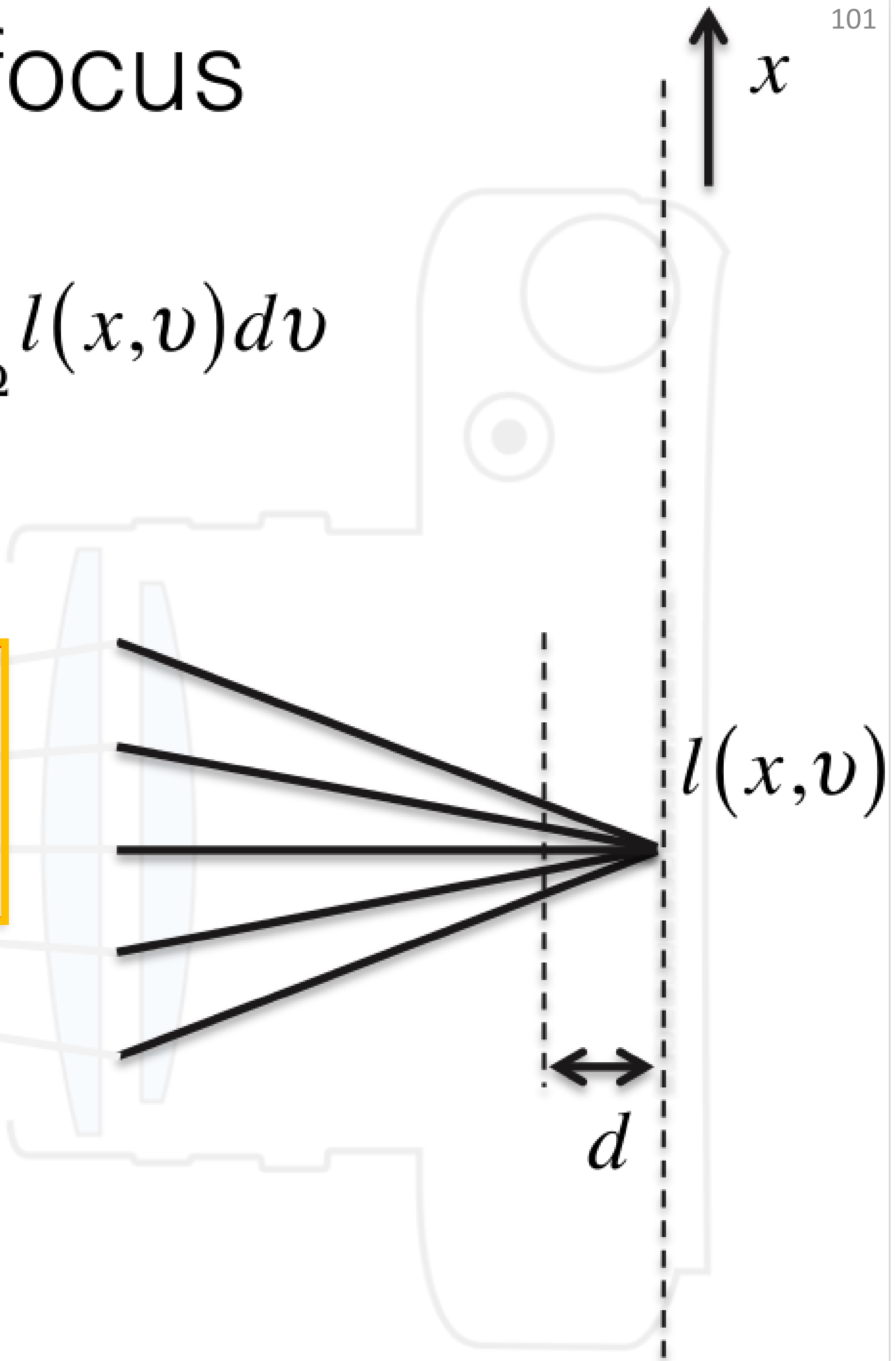


Understanding Refocus

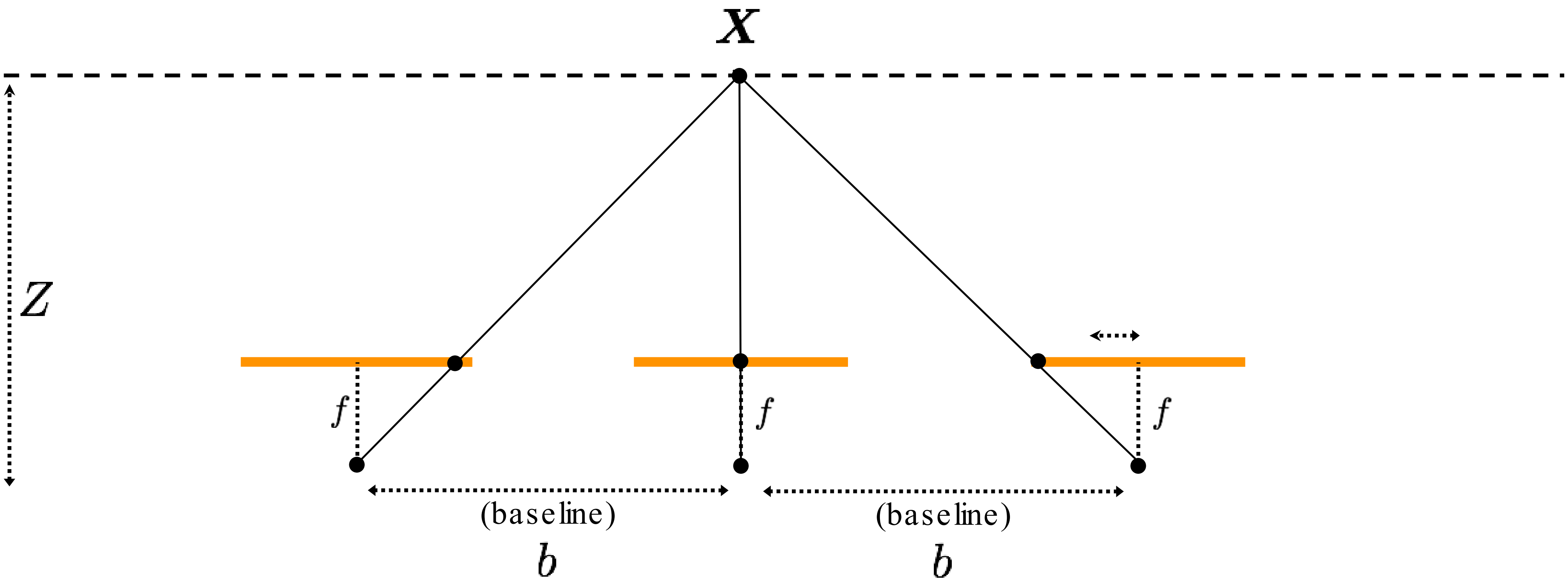
- consider light field inside camera
- synthesize image on sensor $i_{d=0}(x)=\int_{\Omega} l(x,v)\,dv$

$$\boxed{i_d(x)=\int_{\Omega} l(x+dv,v)\,dv}$$

Where did this equation come from?

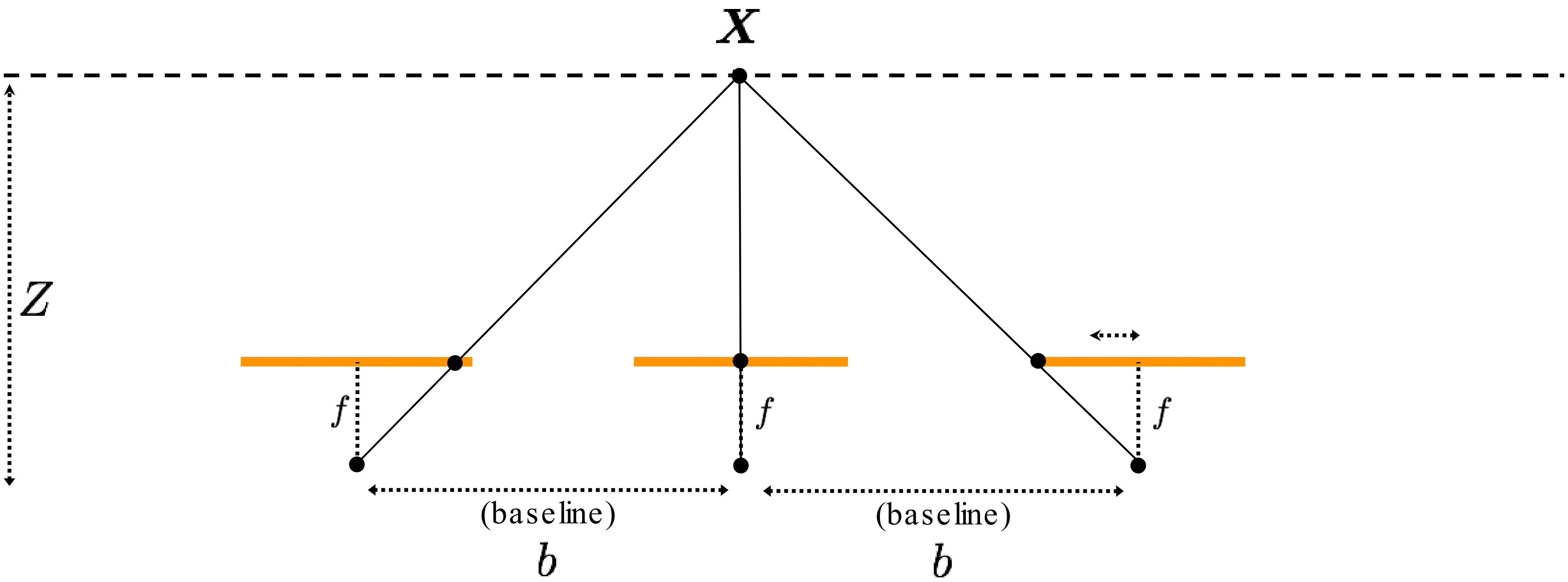


Stereo view of a lightfield camera



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

Stereo view of a lightfield camera



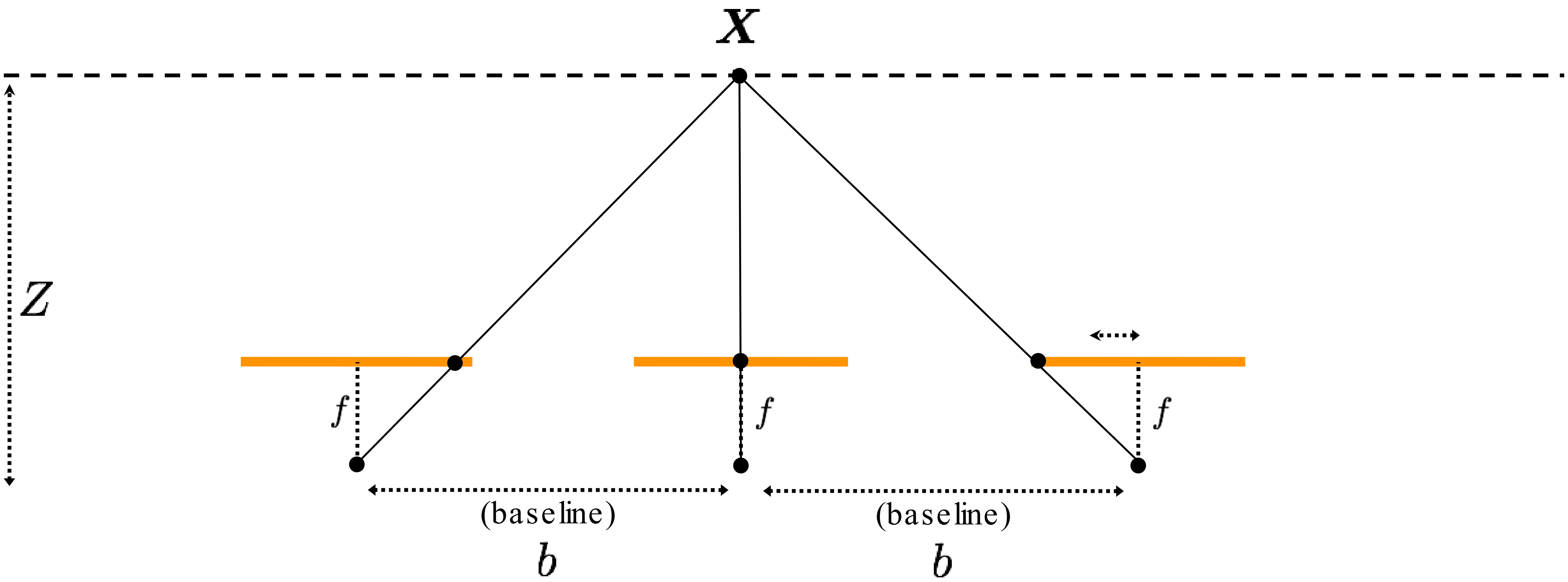
What are the different "cameras" in the lightfield case?

- 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

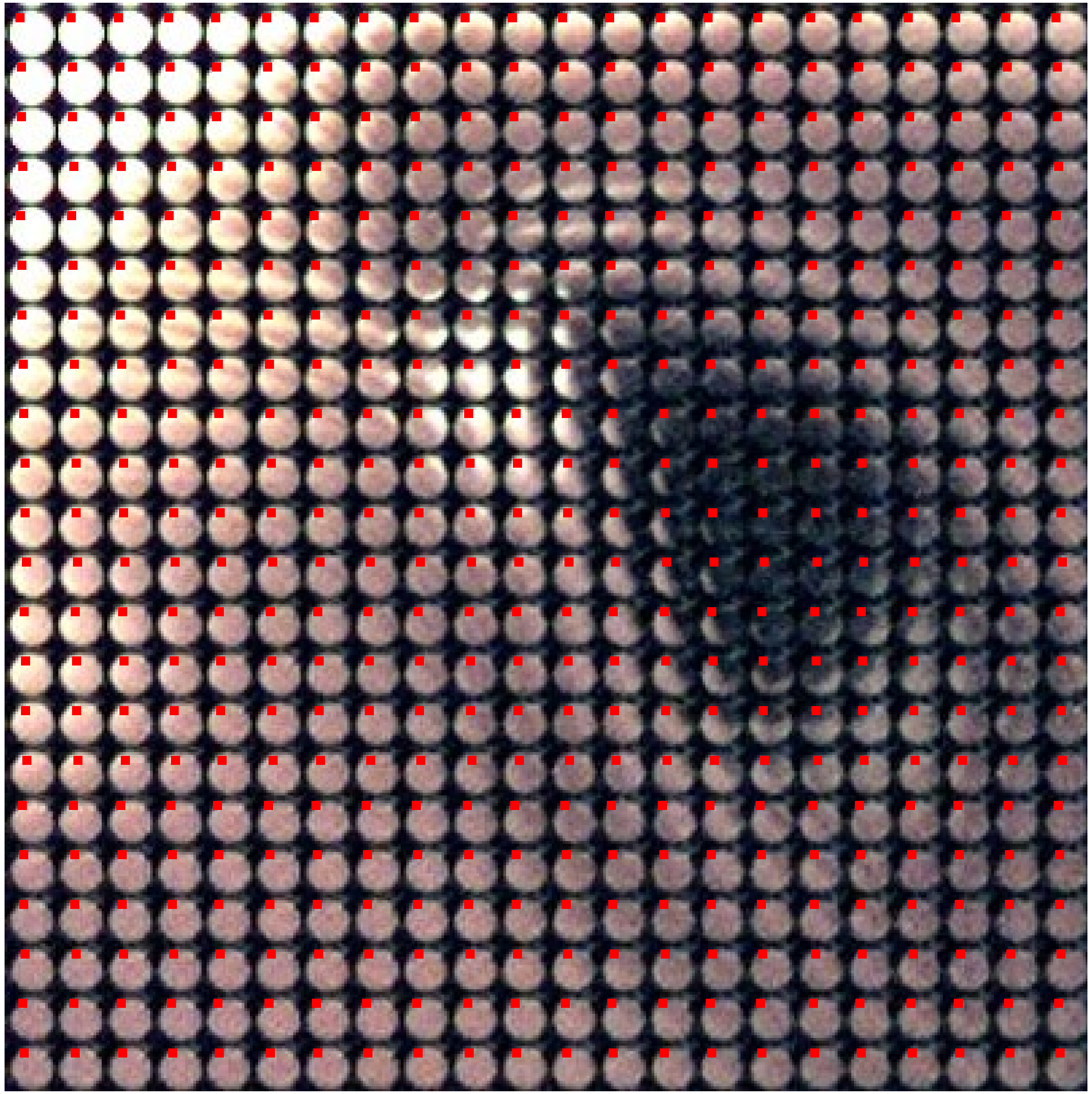
Refocusing example

3D from lightfield

Simulate different viewpoints?

- Pick same pixel within each aperture view

Can we use different viewpoints for stereo?



3D from lightfield

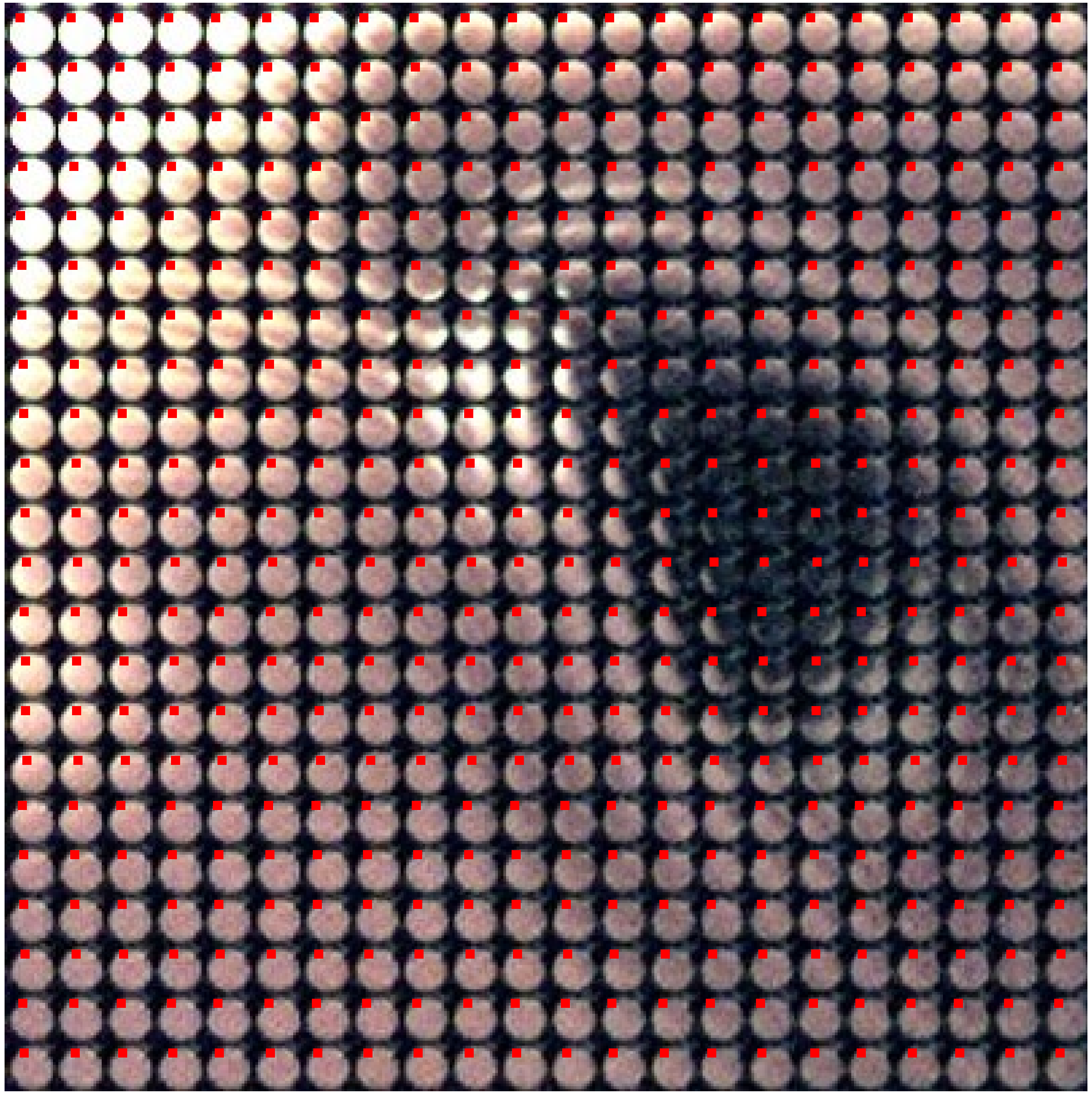
Simulate different viewpoints?

- 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?

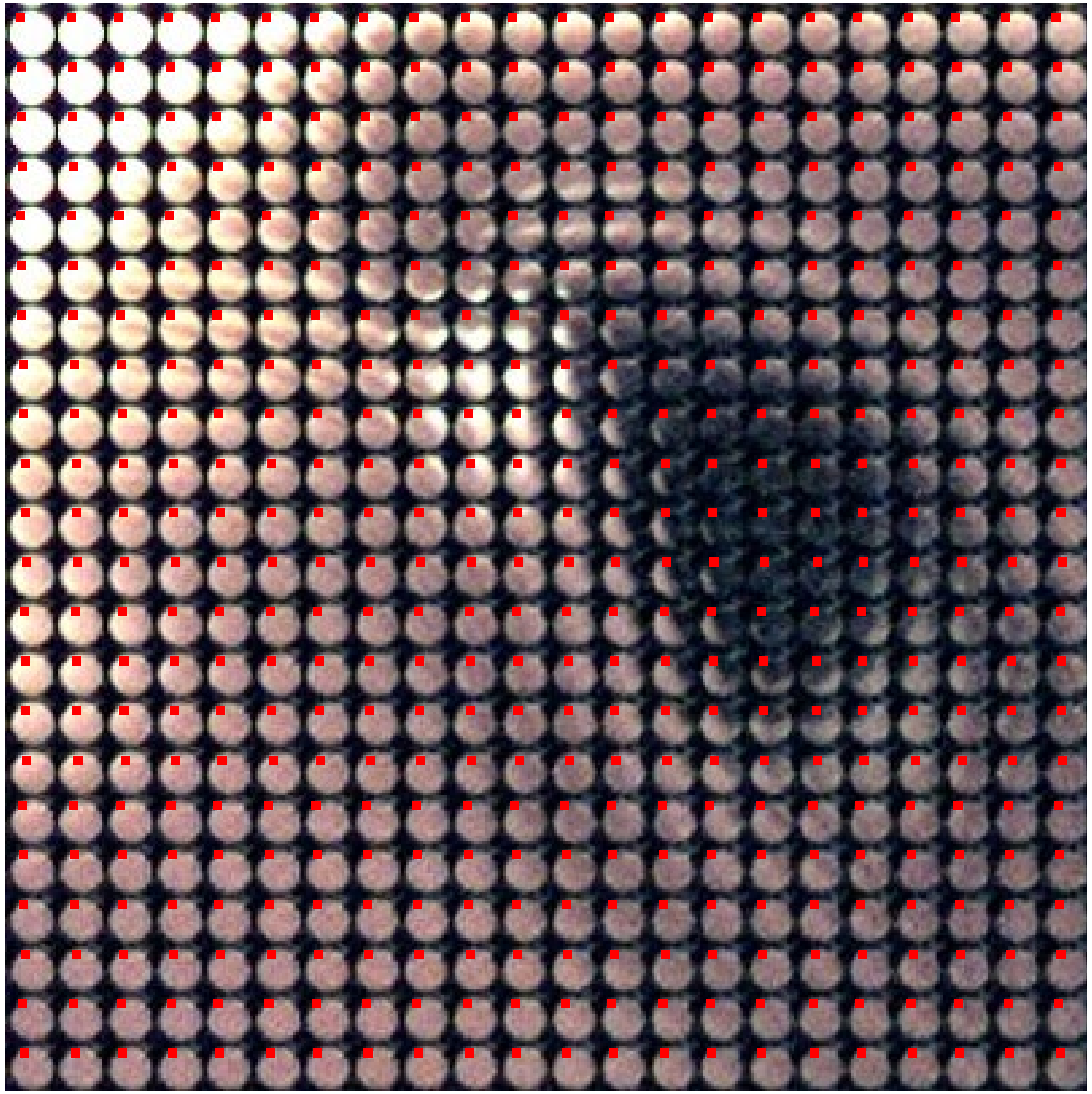
- Pick same pixel within each aperture view

Can we use different viewpoints for stereo?

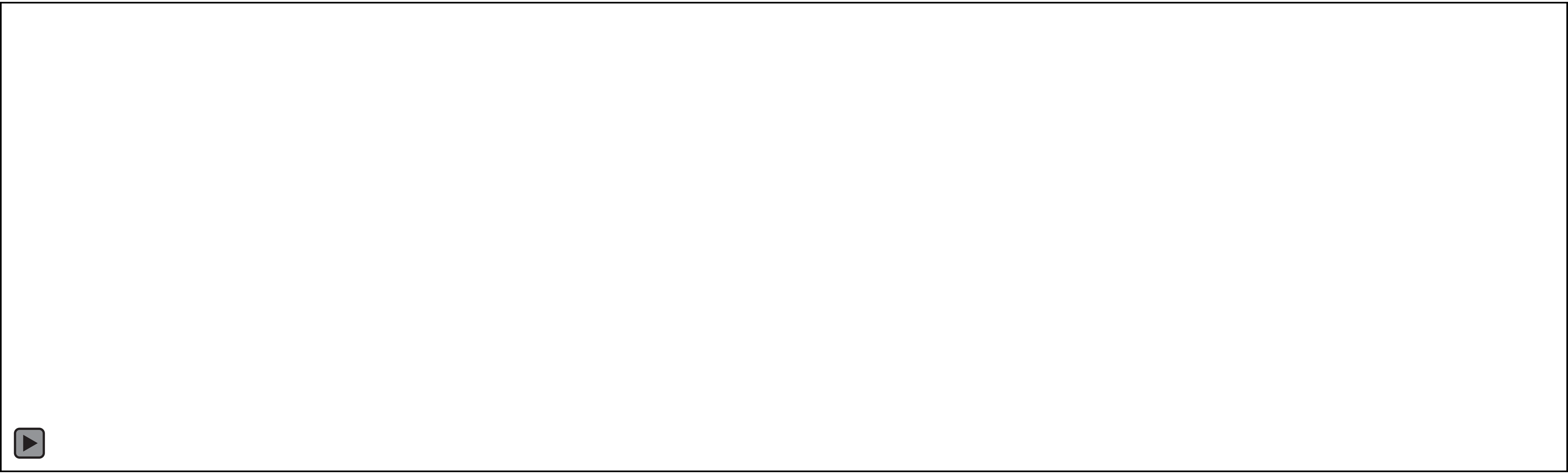
- Very small baseline to use disparity algorithm.
- Standard algorithm only works with two views.

Can we do something better?

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

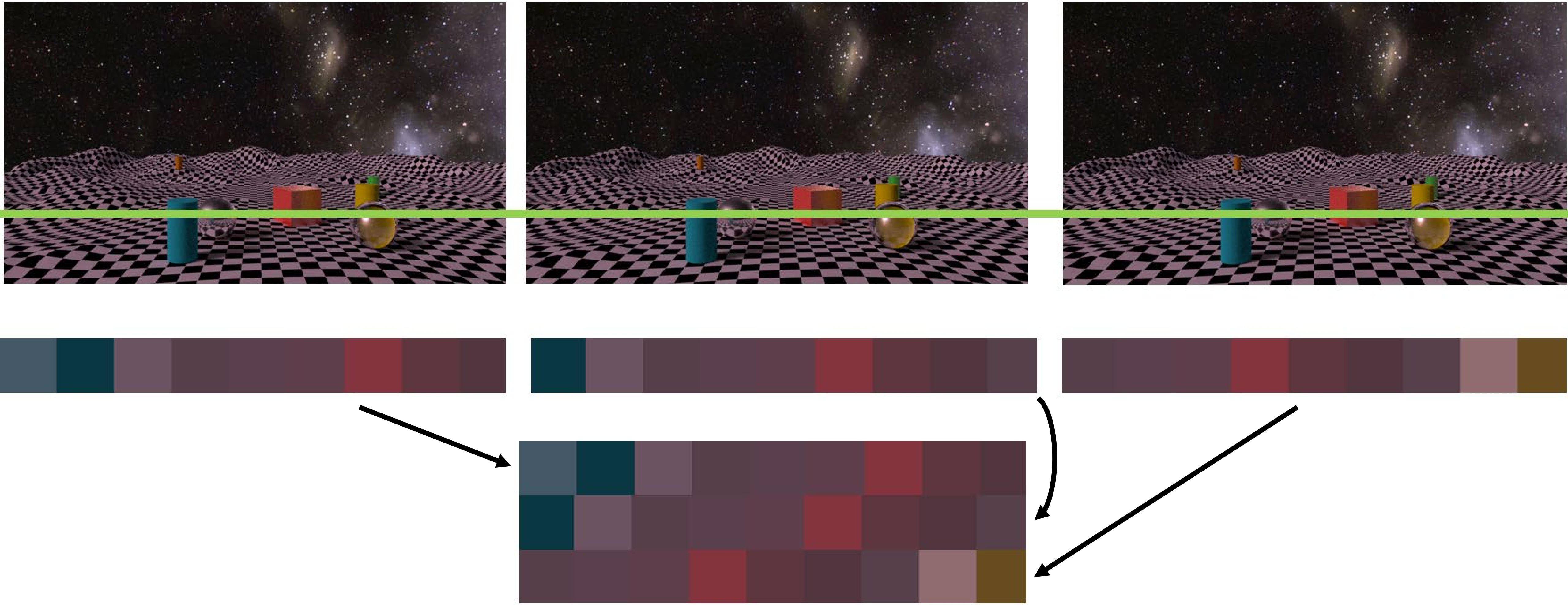


Epipolar plane images (EPIs)



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

Epipolar plane images (EPIs)



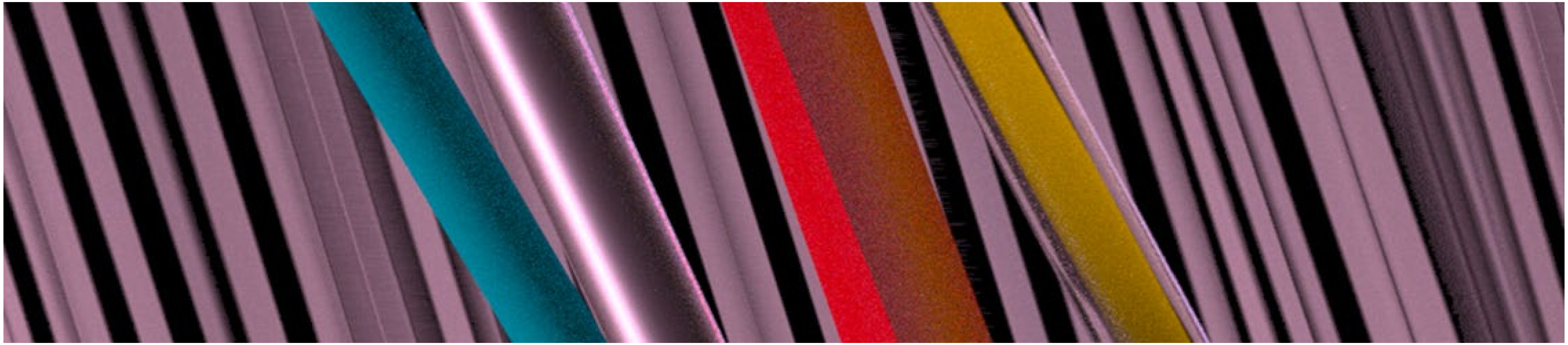
Take the same row out of all images in a scanline, and stack these rows in a new 2D image.

Epipolar plane images (EPIs)

Why do we see straight lines?

move along image columns

move along scanline



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

Epipolar plane images (EPIs)

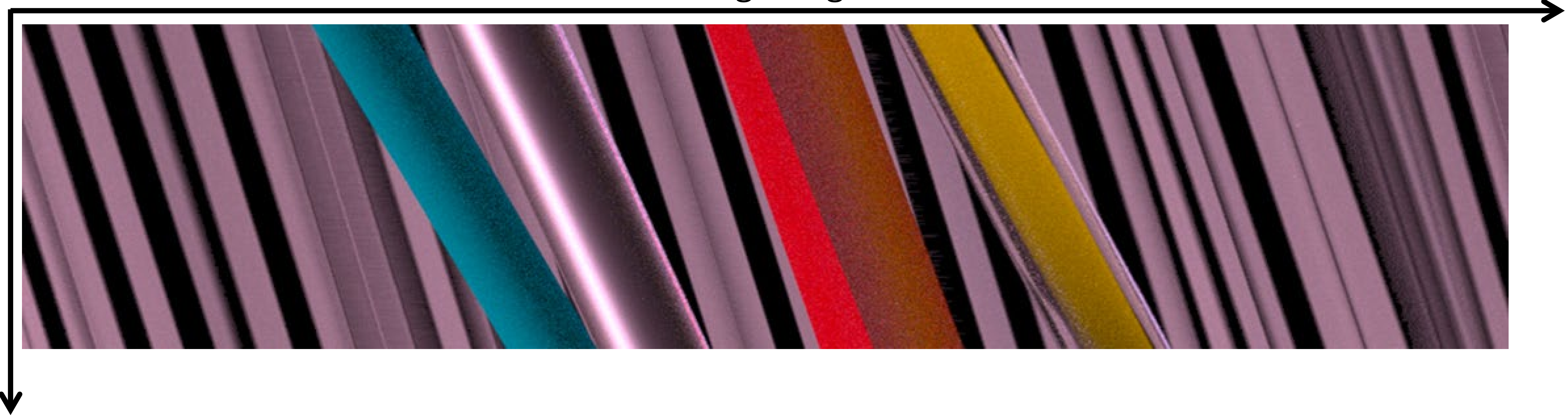
Why do we see straight lines?

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

What does the slope of each line correspond to?

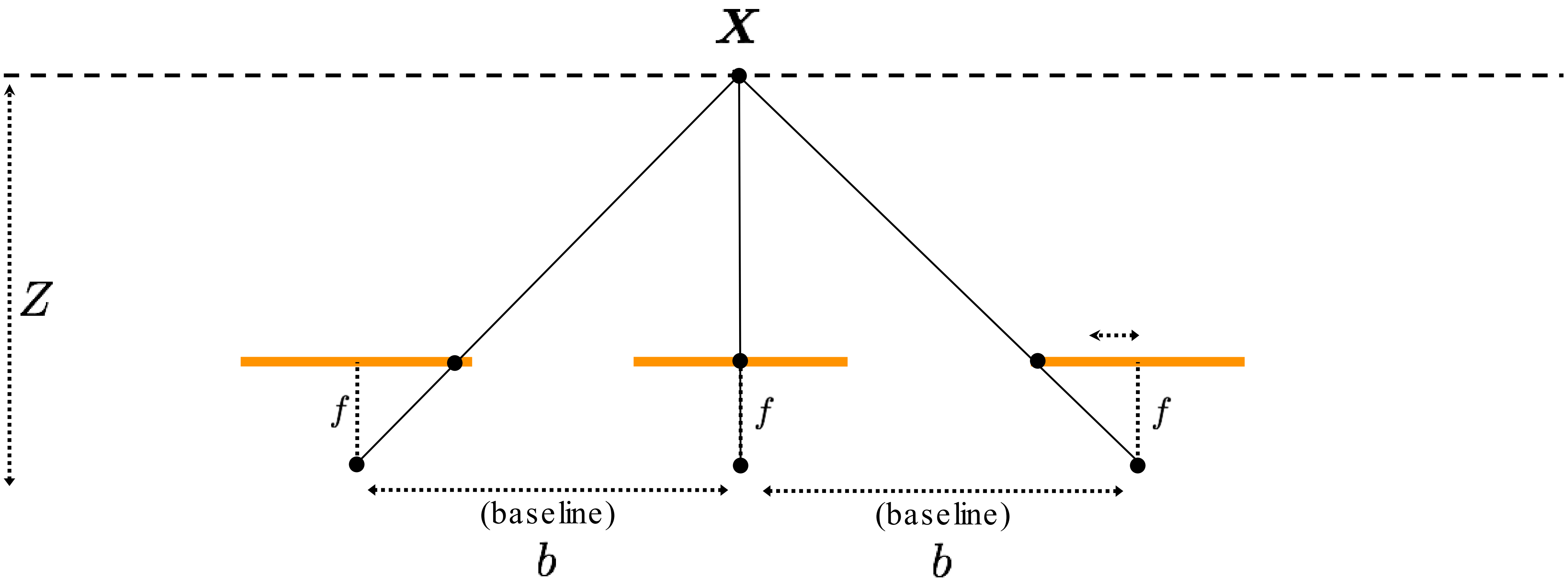
move along image columns

move along scanline



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:

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

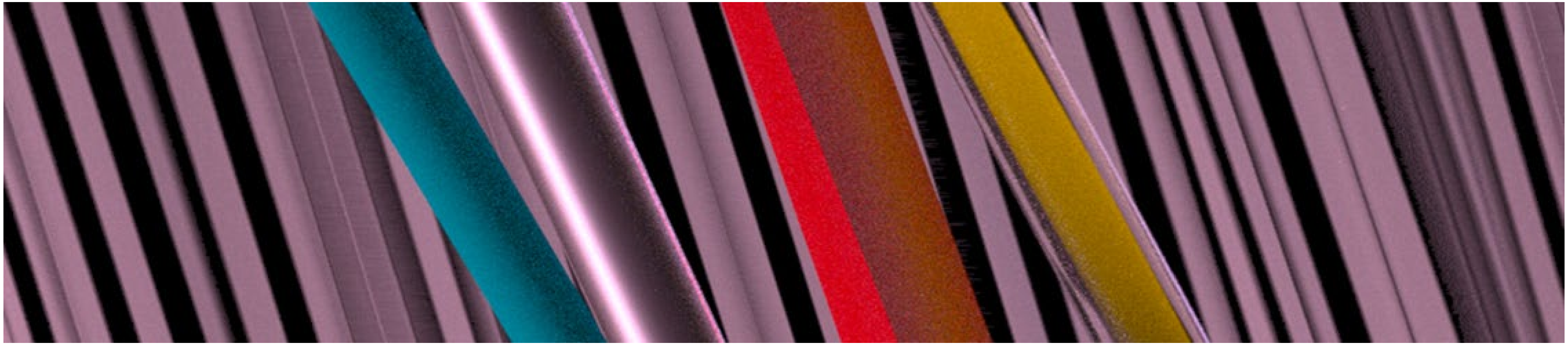
- Changing baseline b corresponds to moving along the scanline.
- Projections x of **X** are on a line of slope inversely proportional to depth.

Epipolar plane images (EPIs)

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

move along image columns

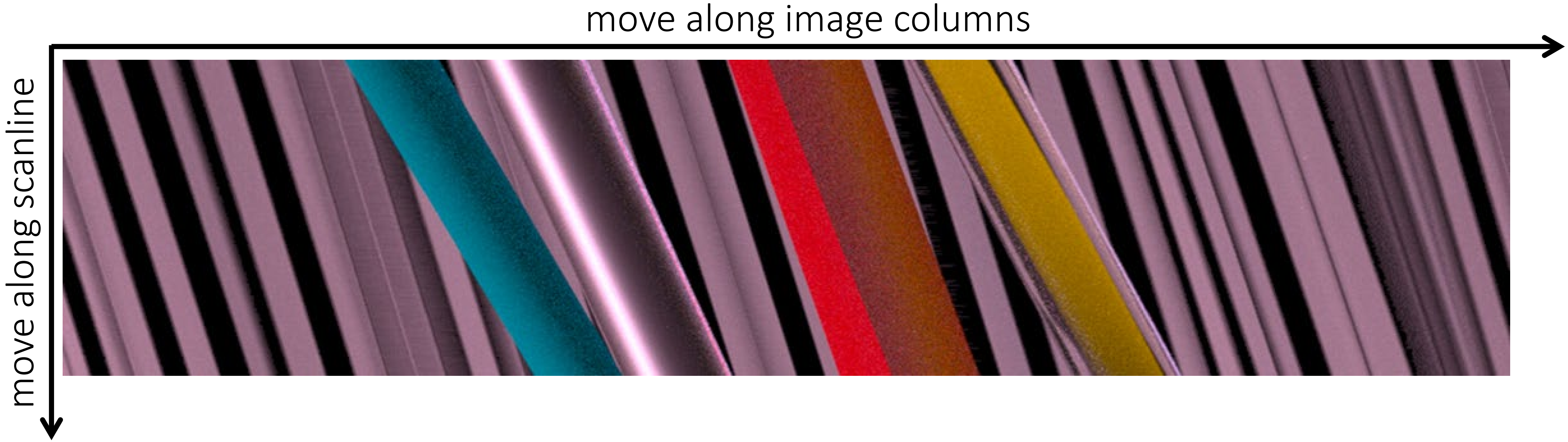
move along scanline



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

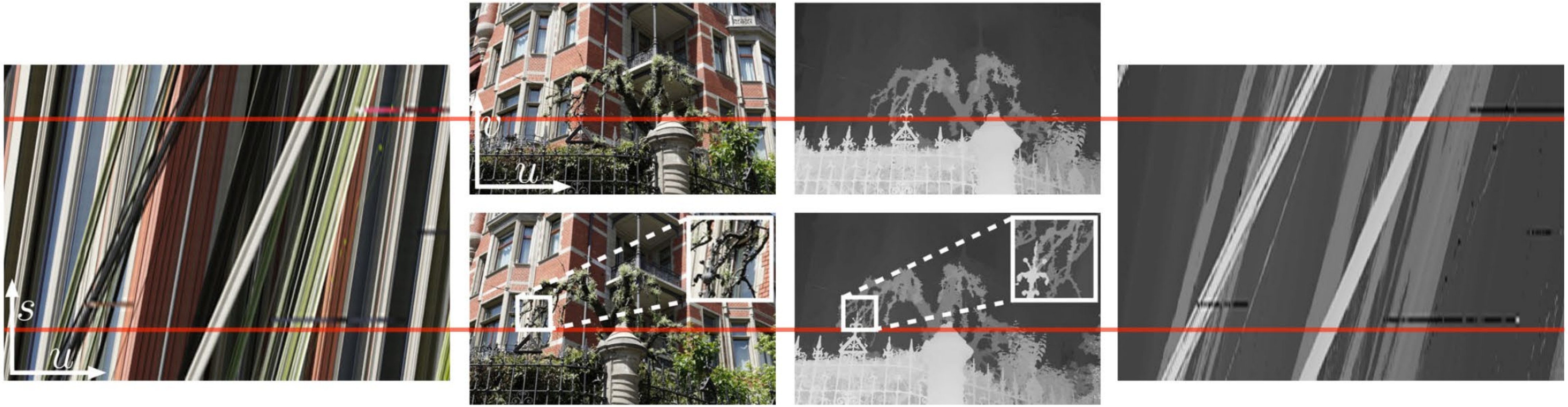
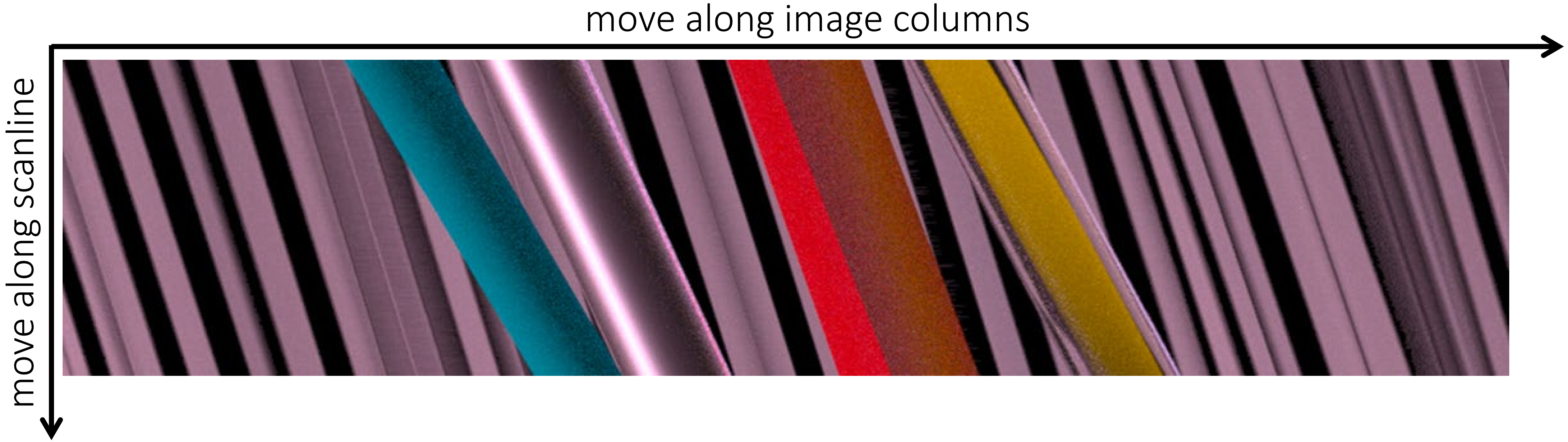


Figure 1: *Our method reconstructs accurate depth from light fields of complex scenes. The images on the left show a 2D slice of a 3D input light field, a so called epipolar-plane image (EPI), and two out of one hundred 21 megapixel images that were used to construct the light field. Our method computes 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.*

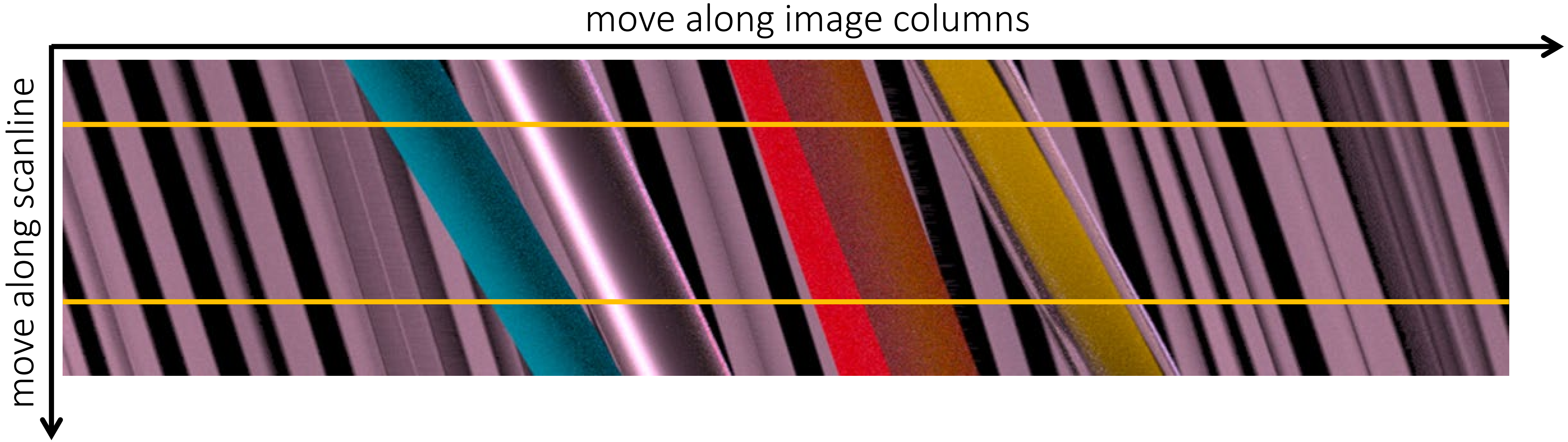
Aside: different types of cameras

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?

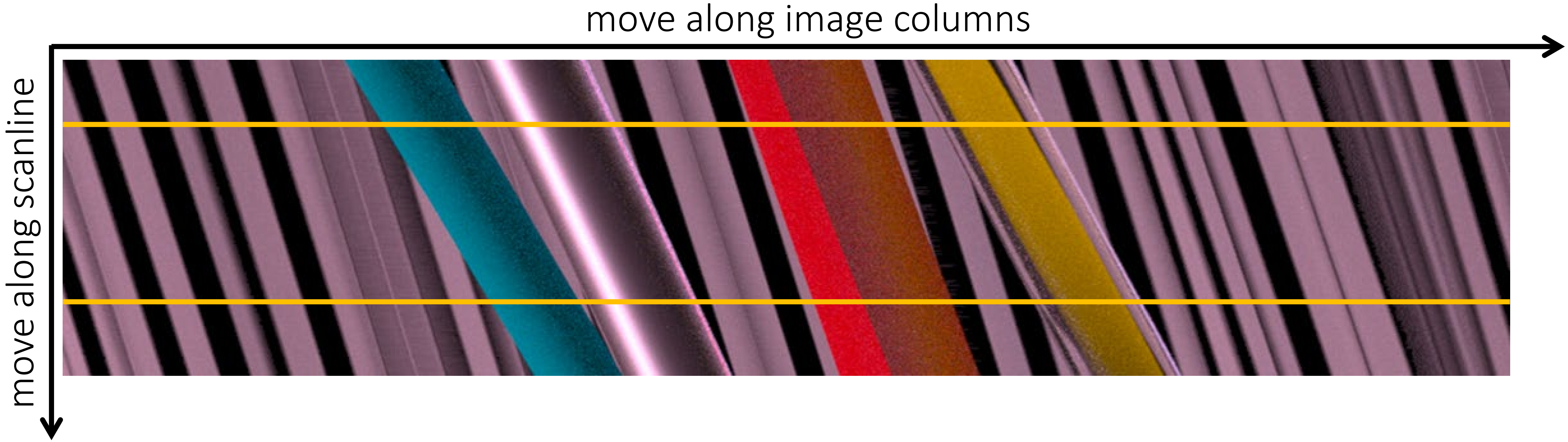


Aside: different types of cameras

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?



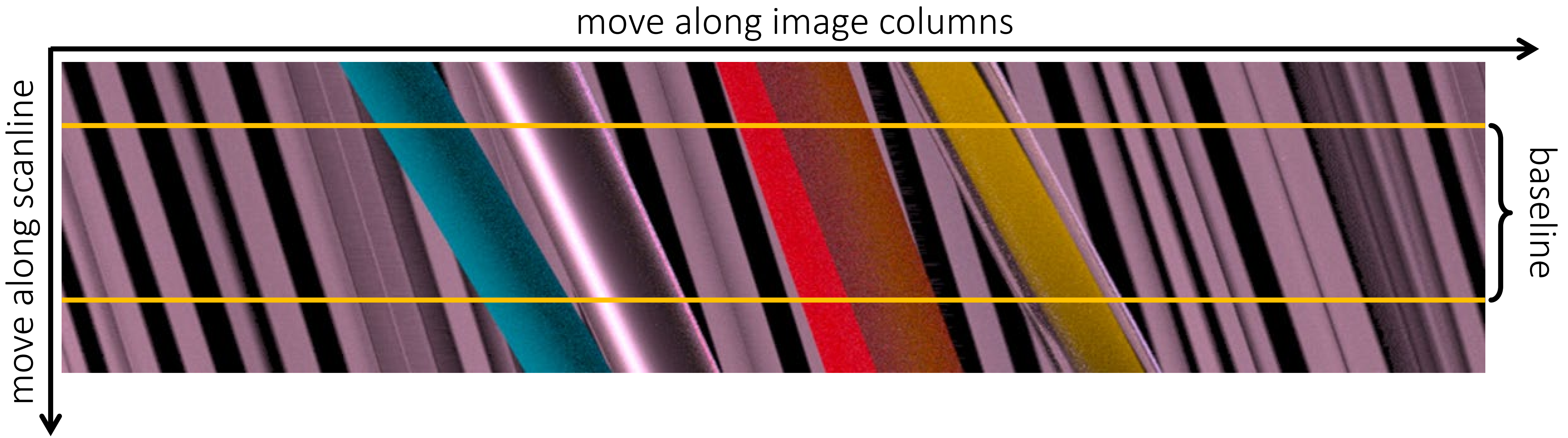
Aside: different types of cameras

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.

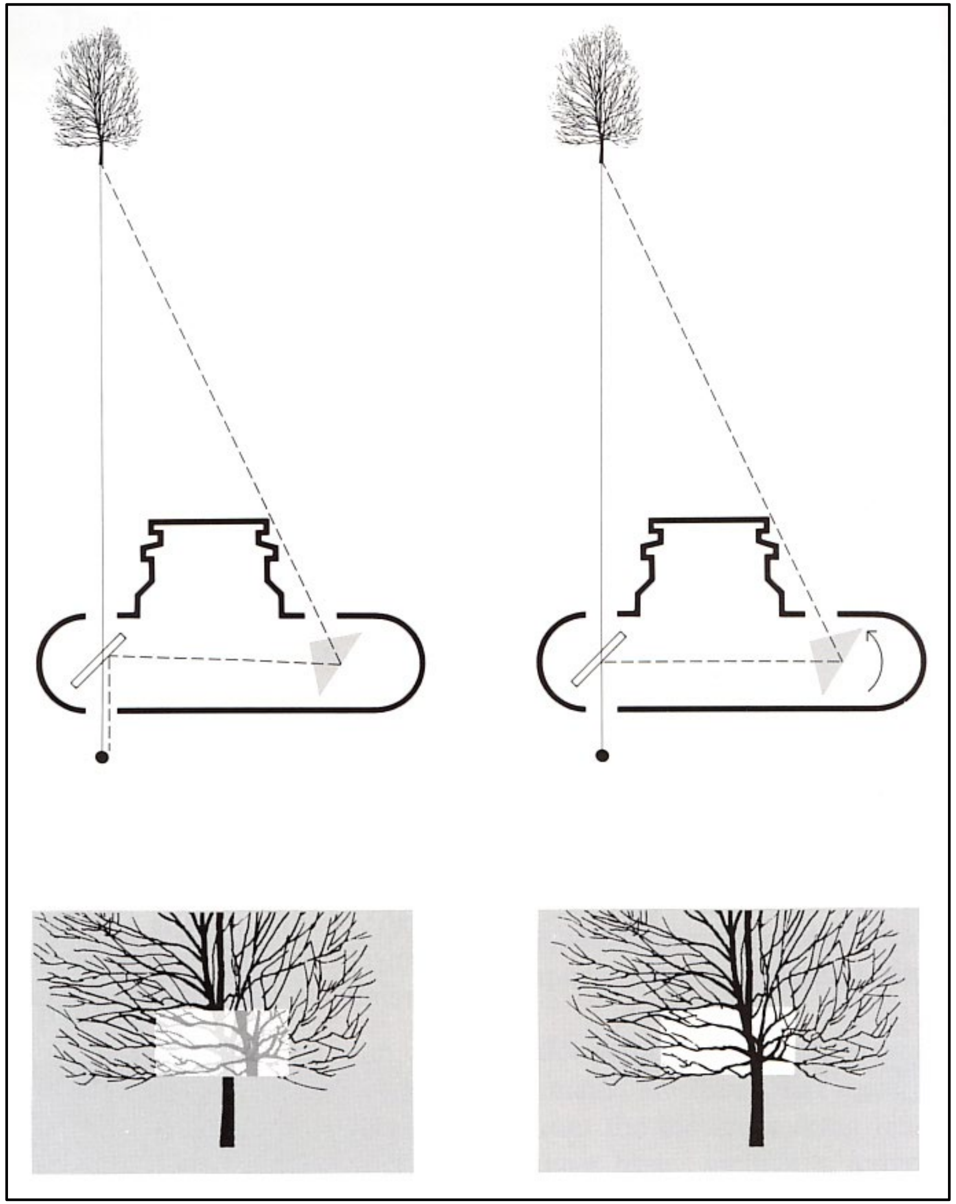


Revisiting auto-focusing

Why does this work in rangefinder cameras?

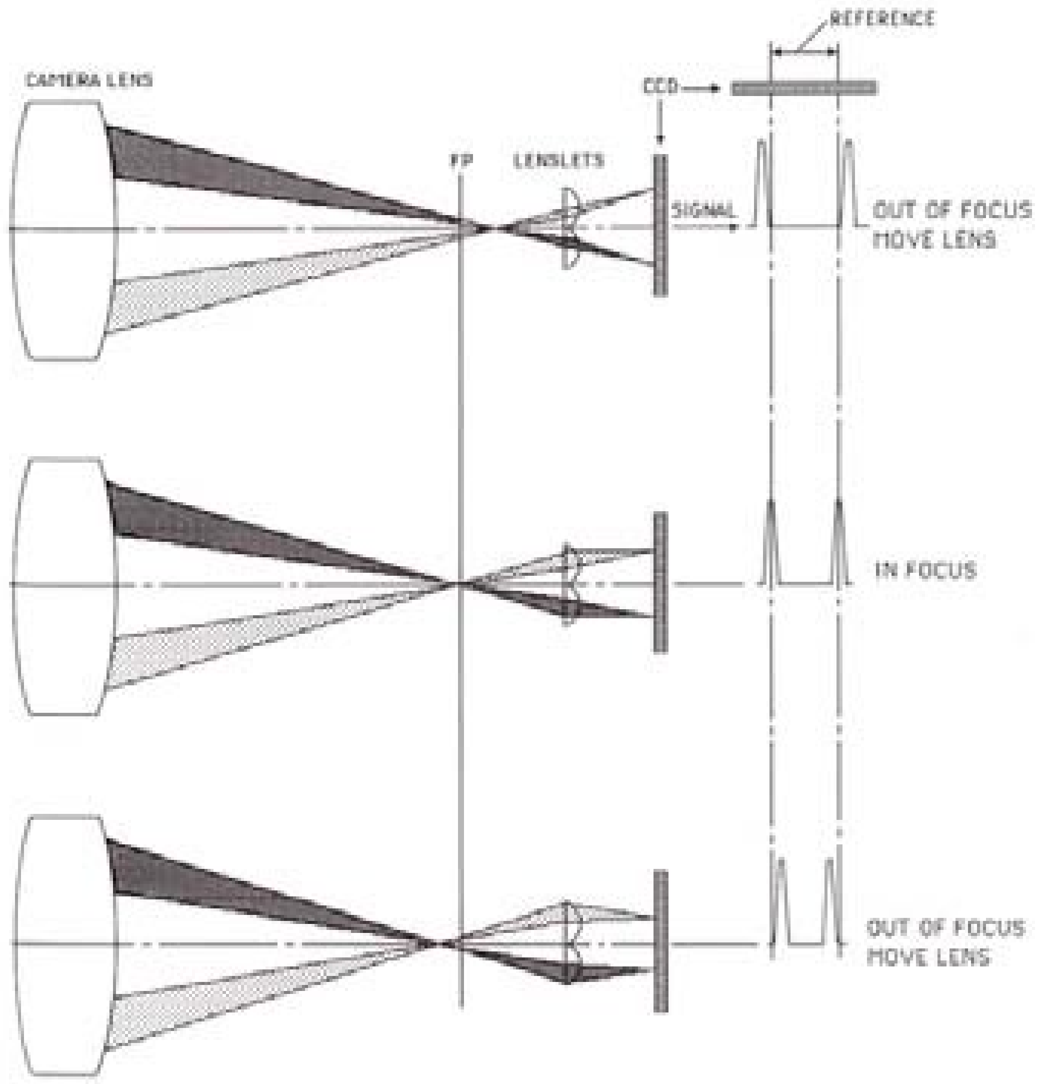
- Focusing based on triangulation: when the image is in focus, you will see the two copies aligned.
- Very accurate but very painstaking.
- Different perspective than that of the main lens.

standard in
Leica cameras



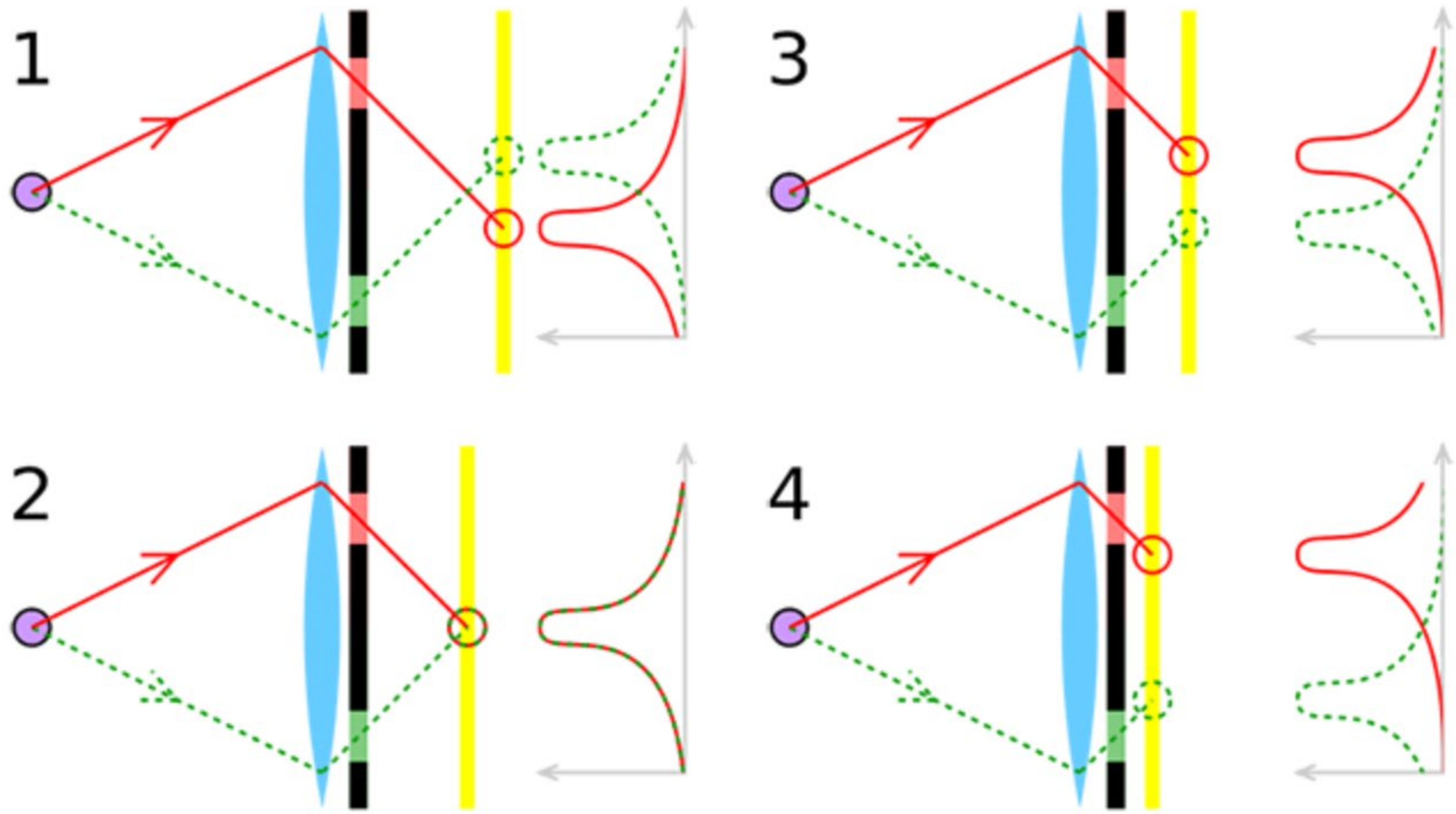
Why does this work for phase detection?

- As the lens moves, ray bundles from an object converge to a different point in the camera and change in angle.
- This change in angle causes them to refocus through two lenslets to different positions on a separate AF sensor.
- A certain spacing between these double images indicates that the object is “in focus”.



Demo: http://graphics.stanford.edu/courses/cs178/applets/autofocuspd.html

Why does this work for phase detection?

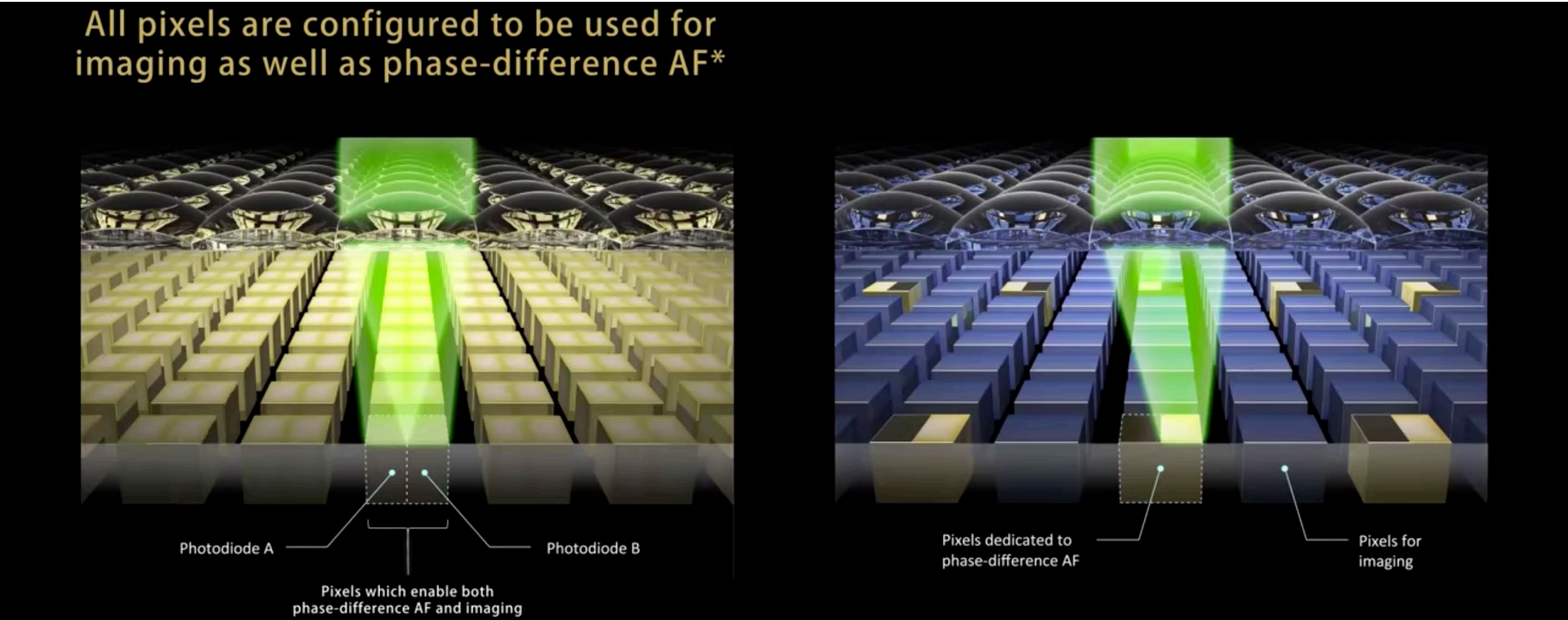


Each yellow box indicates *two* sensors, each measuring light from different parts of the aperture.

- Which one is correct focusing?
- How do you need to move the lens or sensor to get correct focusing?

Dual-pixels: disparity with incredibly small baseline

- Split each pixel into two independent photodiodes–like a two-view lightfield.
- Use different pixels for phase detection.
- Many other interesting opportunities (depth from stereo/lightfield with a tiny baseline).



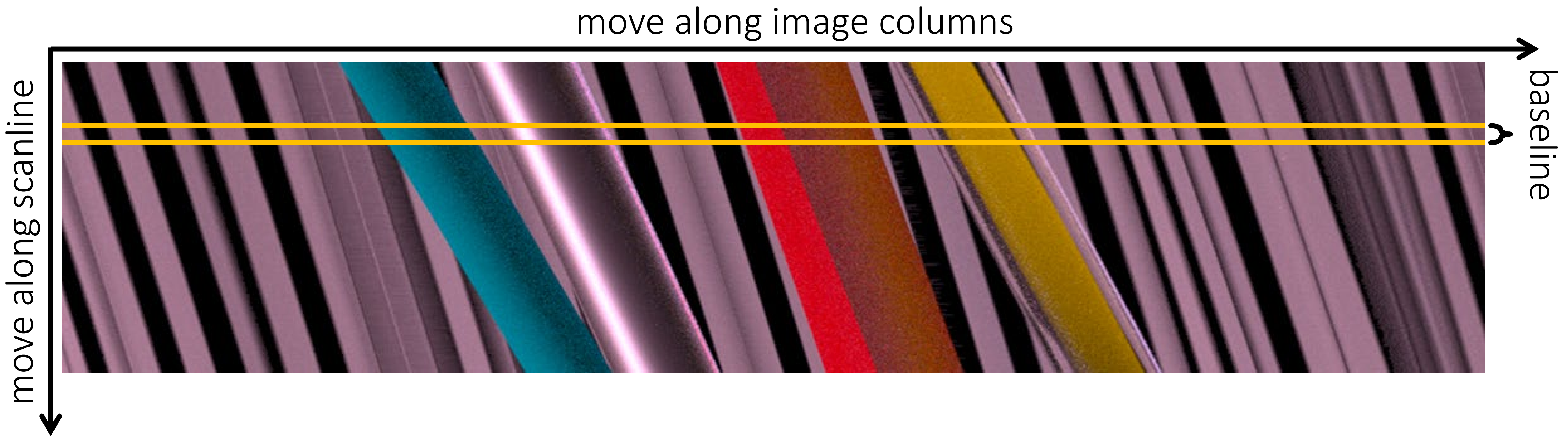
Dual-pixels: disparity with incredibly small baseline

What part of the EPI is captured when we use dual pixels?

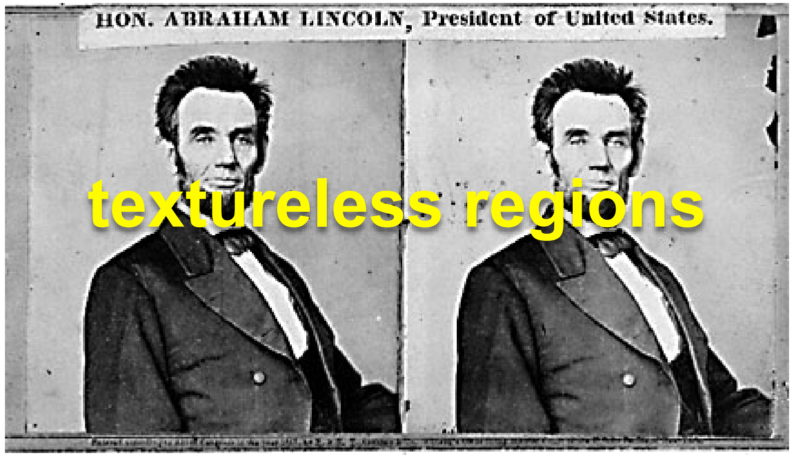
- Two horizontal lines.

How does this compare to using a stereo pair?

- The baseline is *a lot* smaller.



When are correspondences difficult?

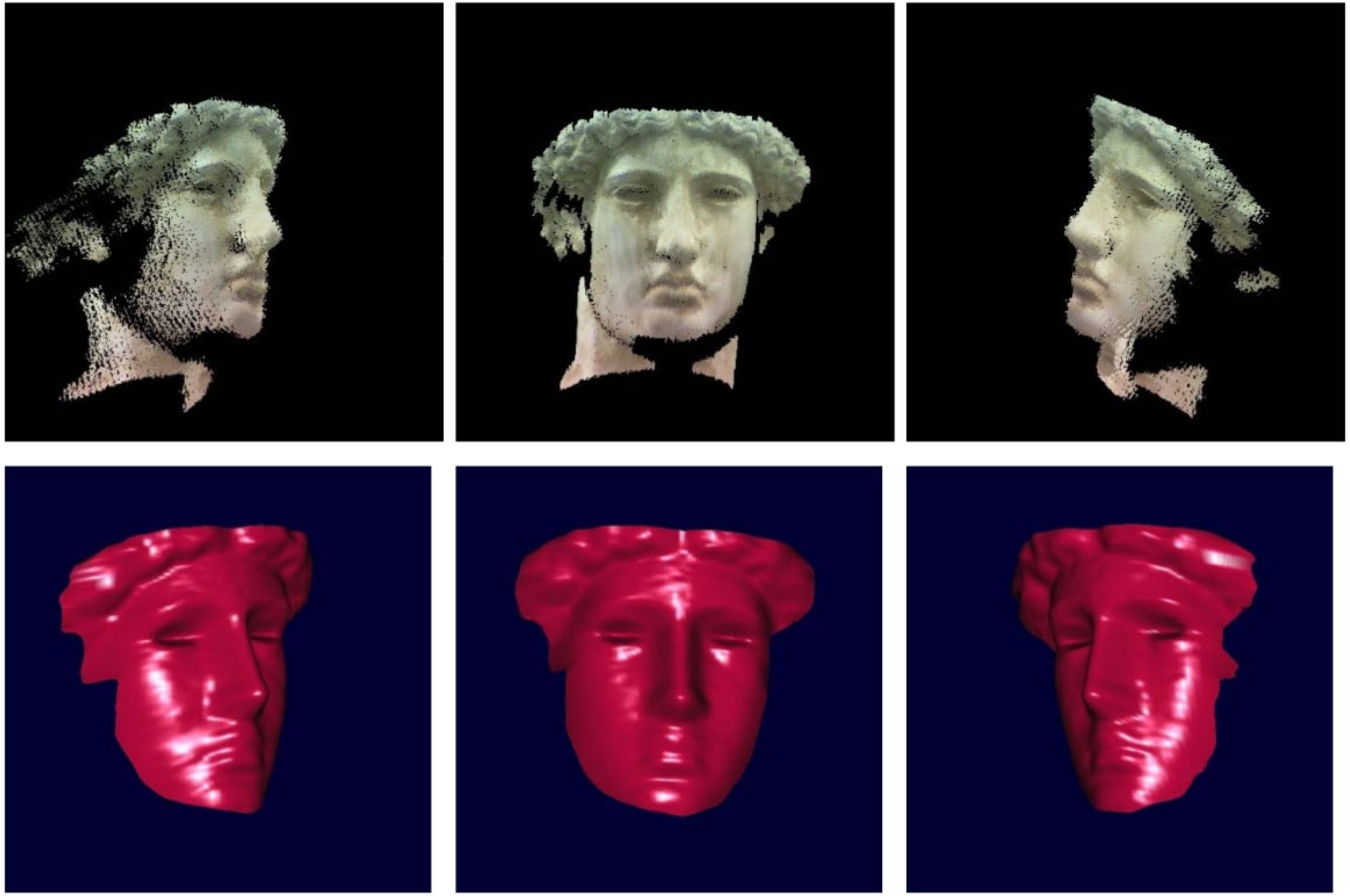


Structured light

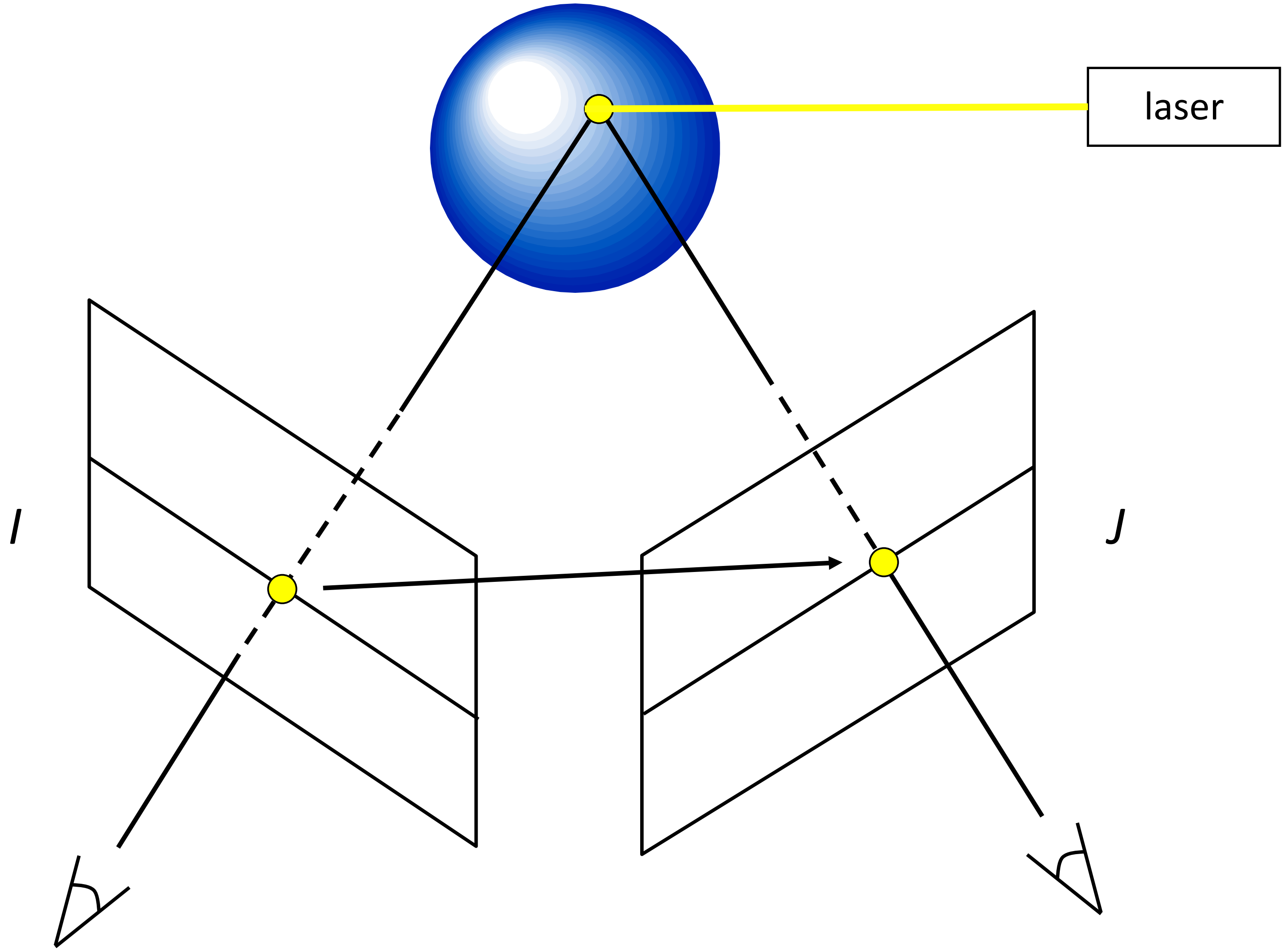
Use controlled ("structured") light to make correspondences easier

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

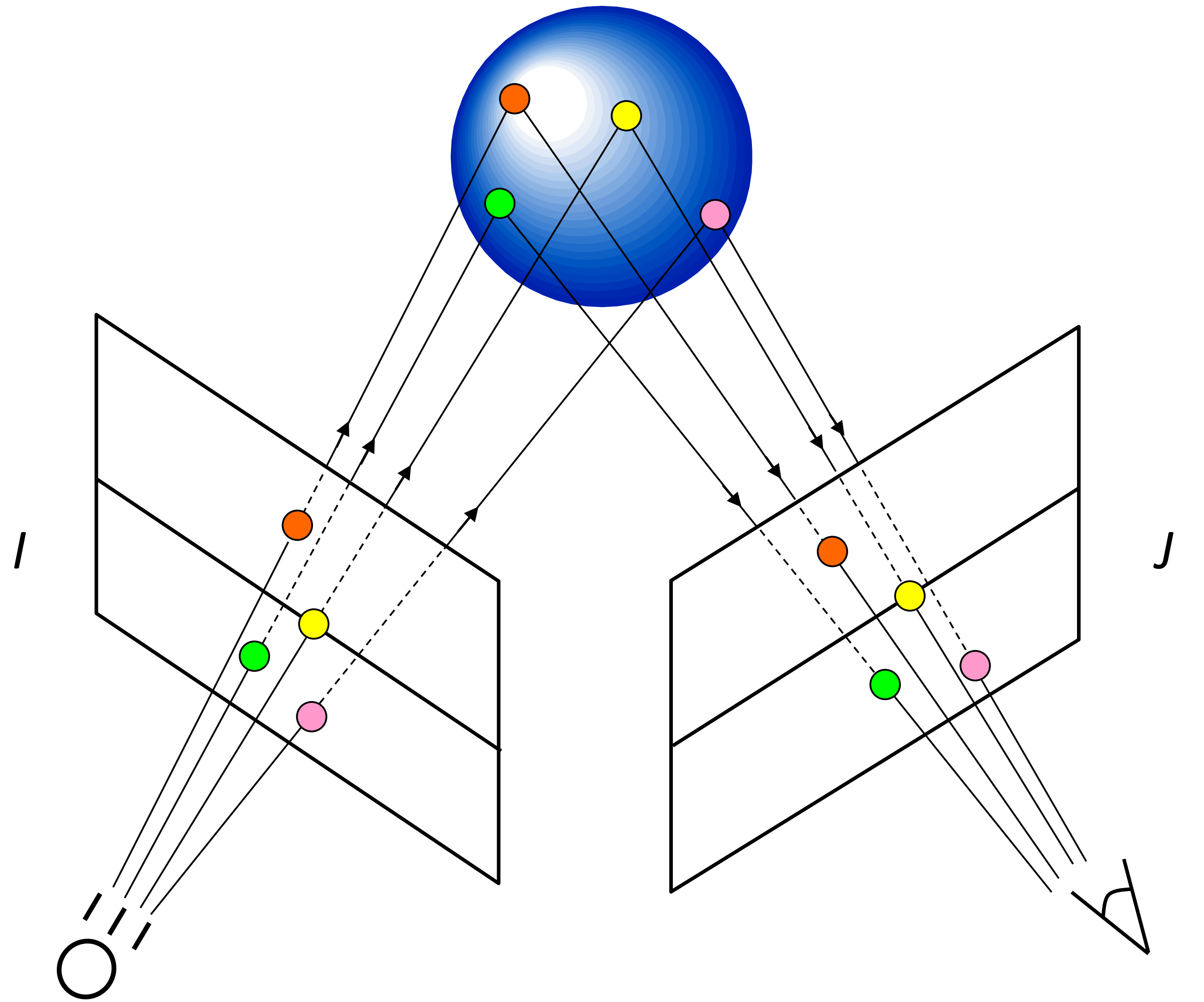


Structured light and two cameras

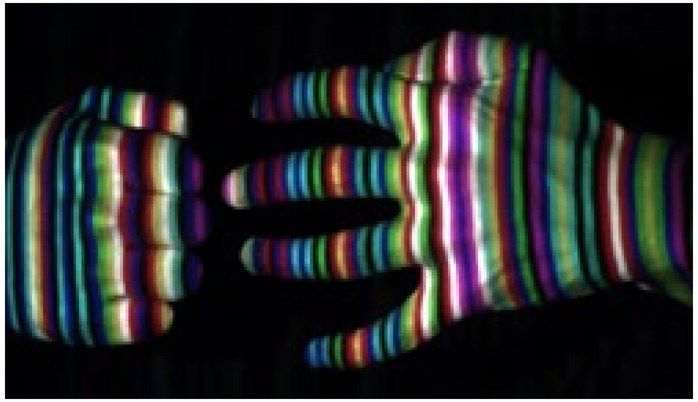


Structured light and one camera

Projector acts like
“reverse” camera

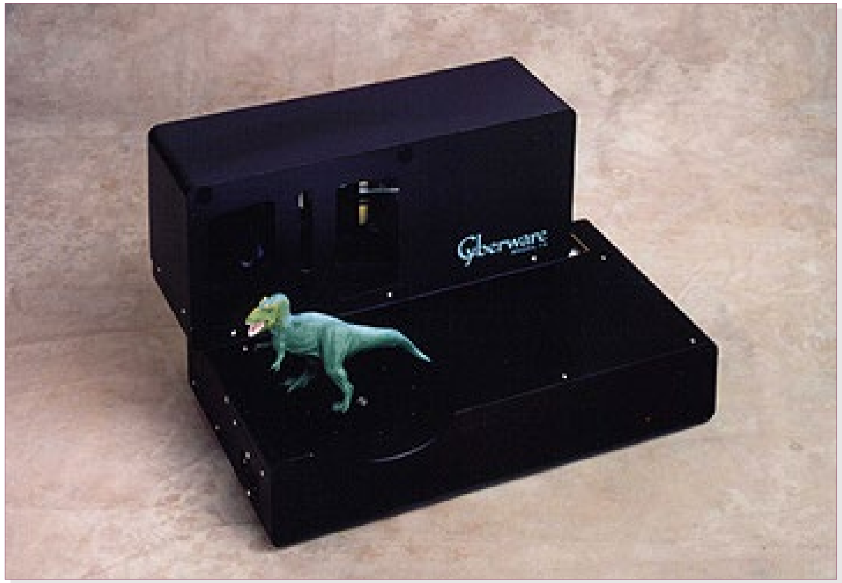


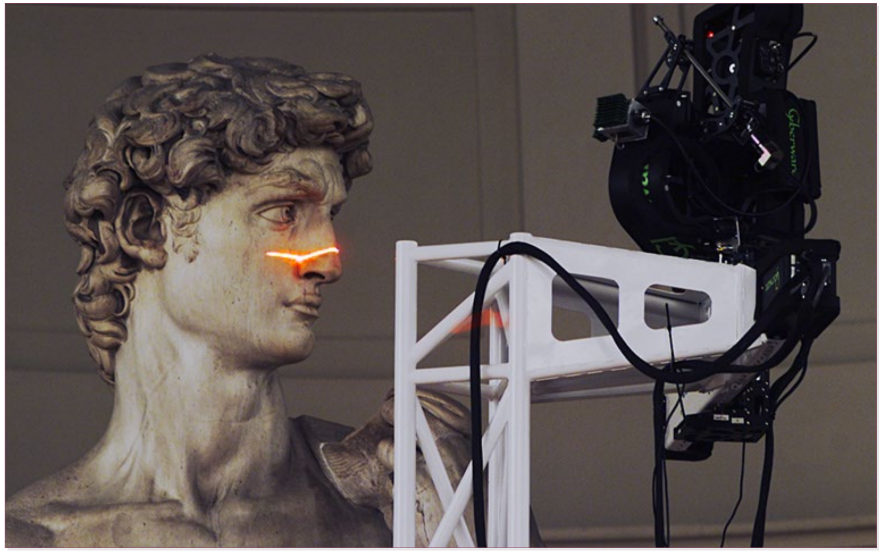
Structured Light



- 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).

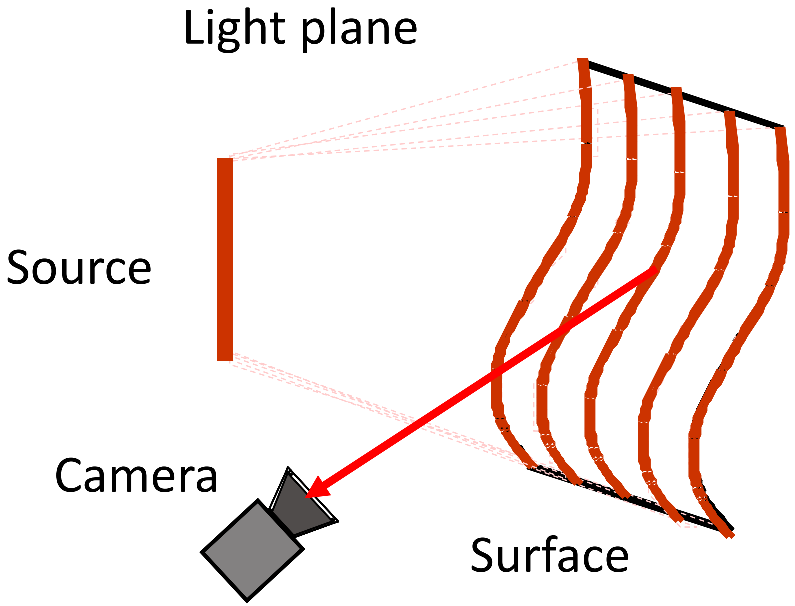
3D Scanning using structured light





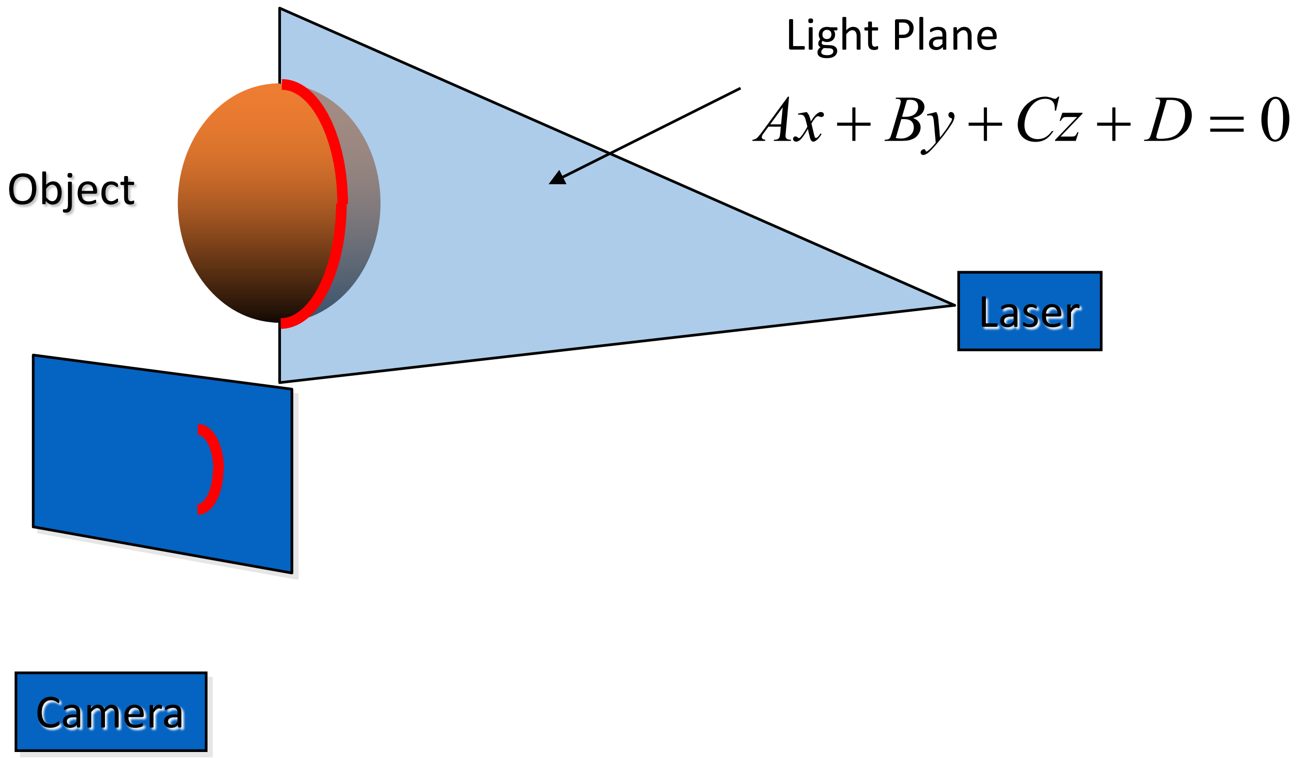
Do we need to illuminate the scene point by point?

Light Stripe Scanning – Single Stripe



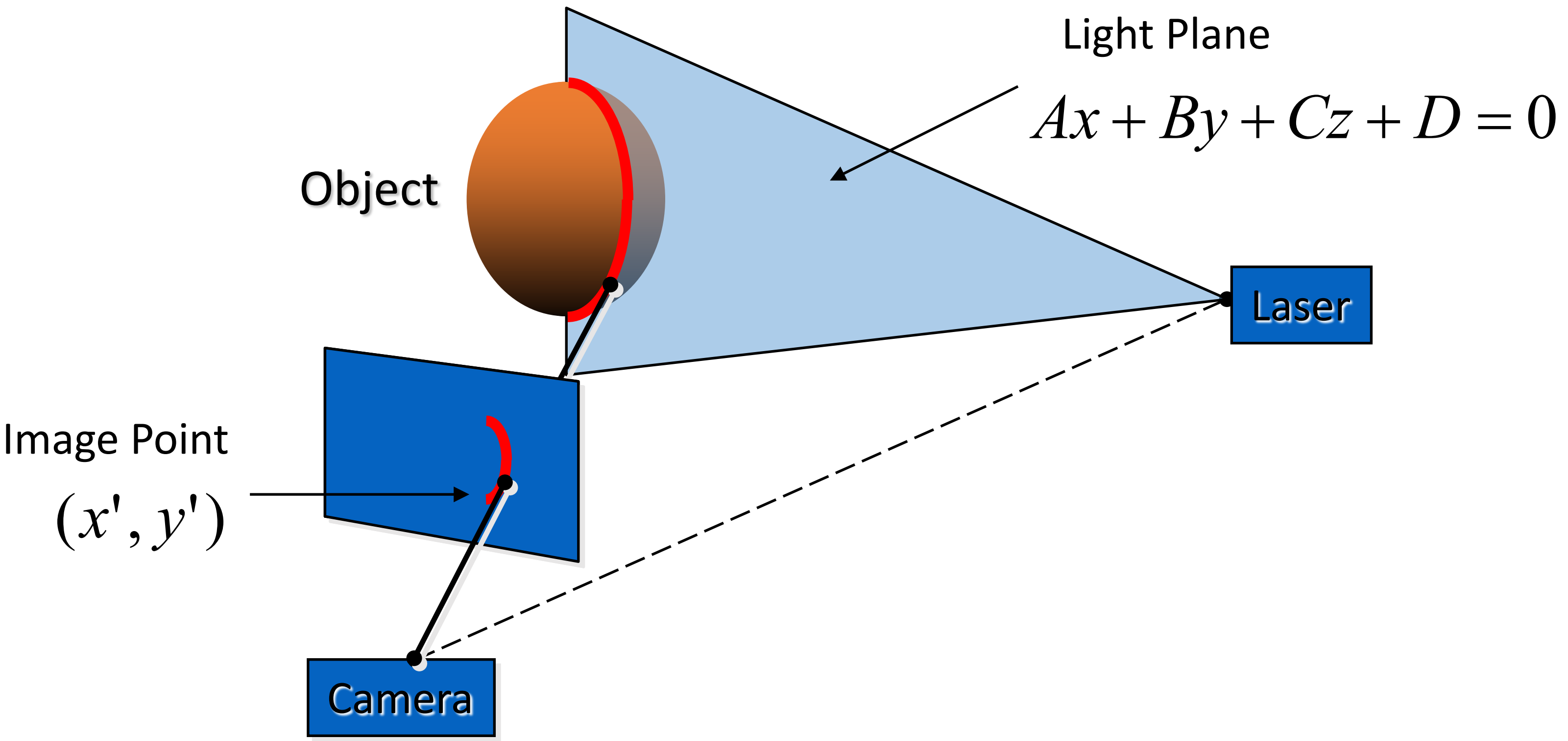
- Faster optical triangulation:
  - Project a single stripe of laser light
  - Scan it across the surface of the object
  - This is a very precise version of structured light scanning
  - Good for high resolution 3D, but still needs many images and takes time

Triangulation



- Project laser stripe onto object

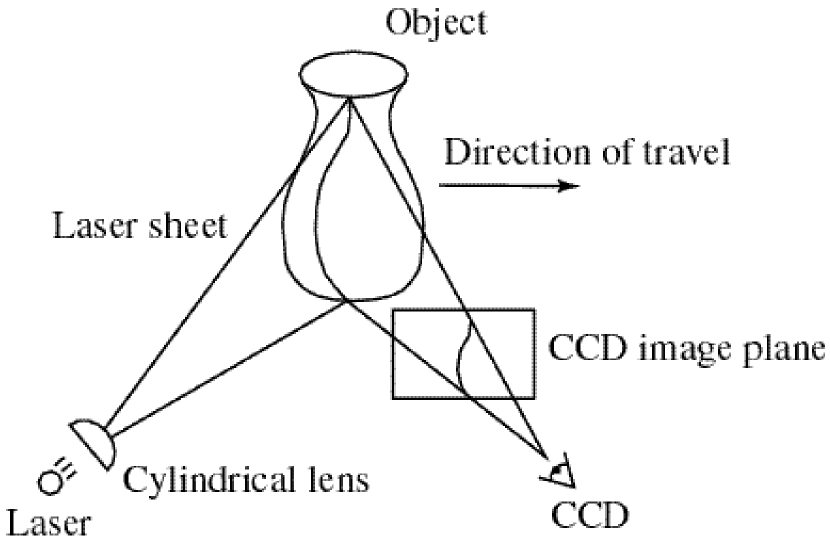
Triangulation

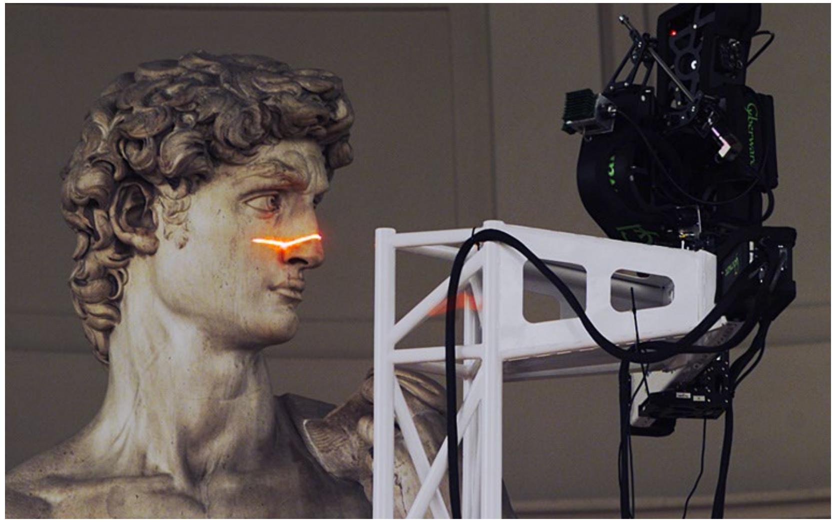


- Depth from ray-plane triangulation:
  - Intersect camera ray with light plane

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

Example: Laser scanner

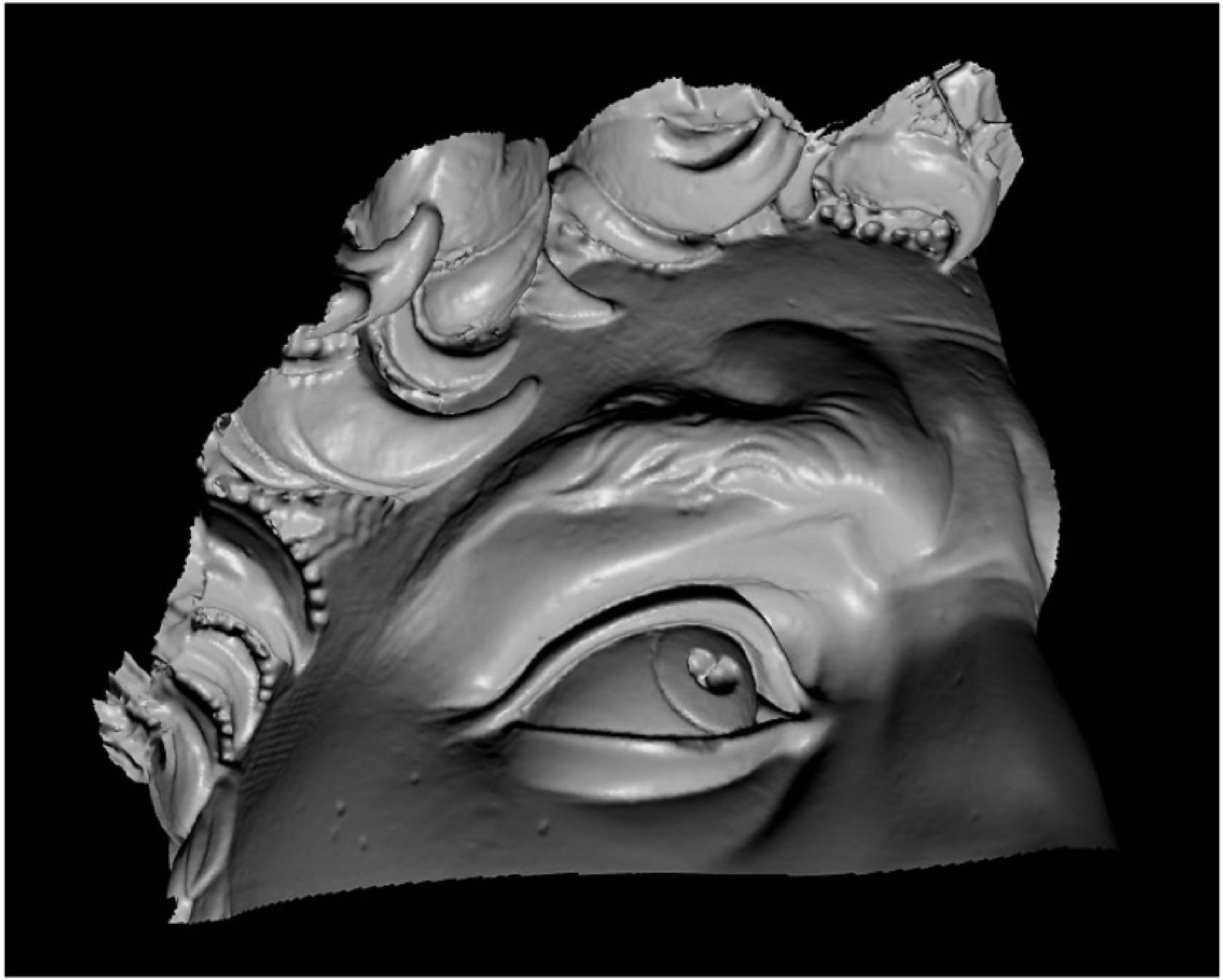




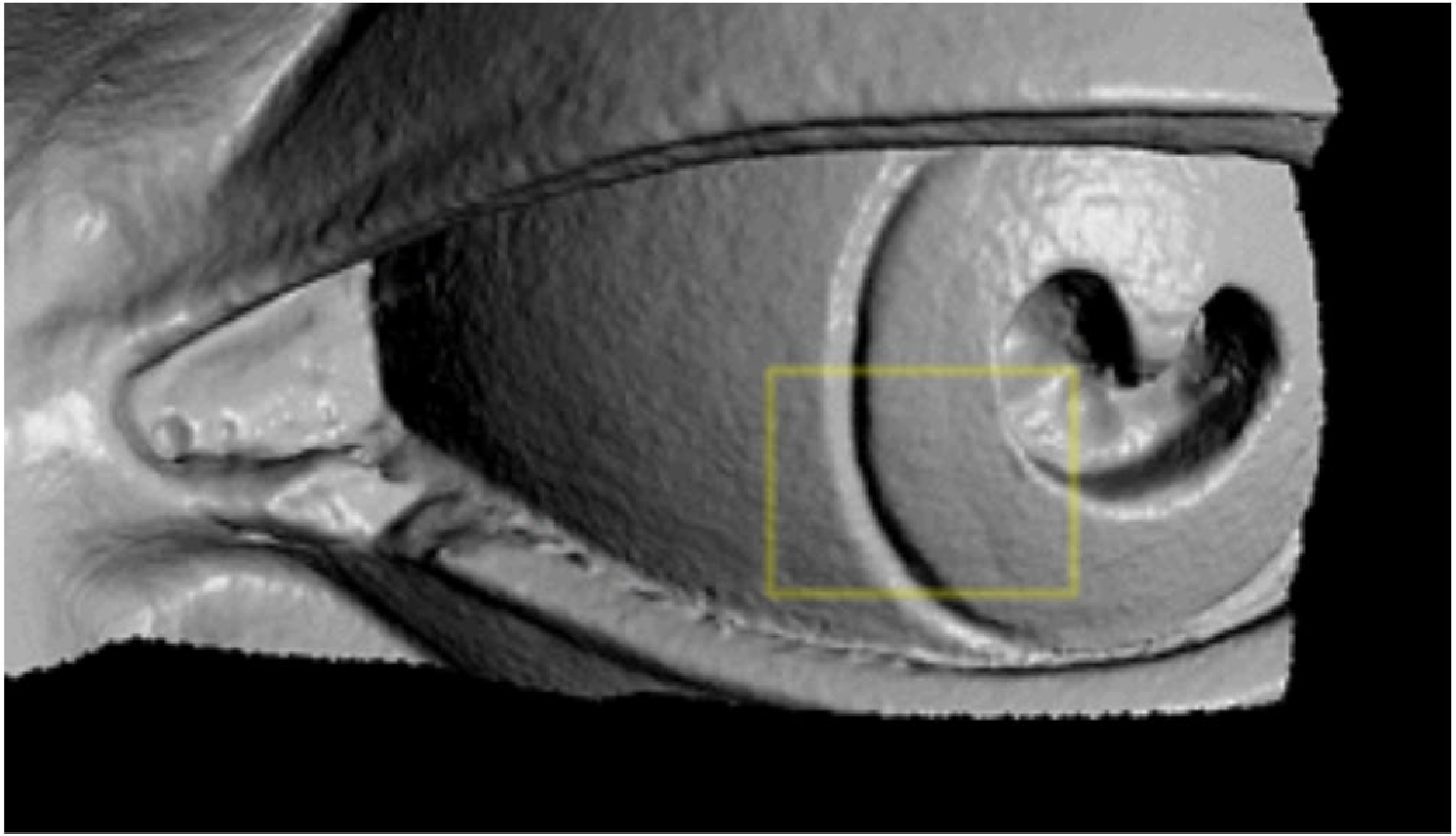
Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/

The Digital Michelangelo Project, Levoy et al.

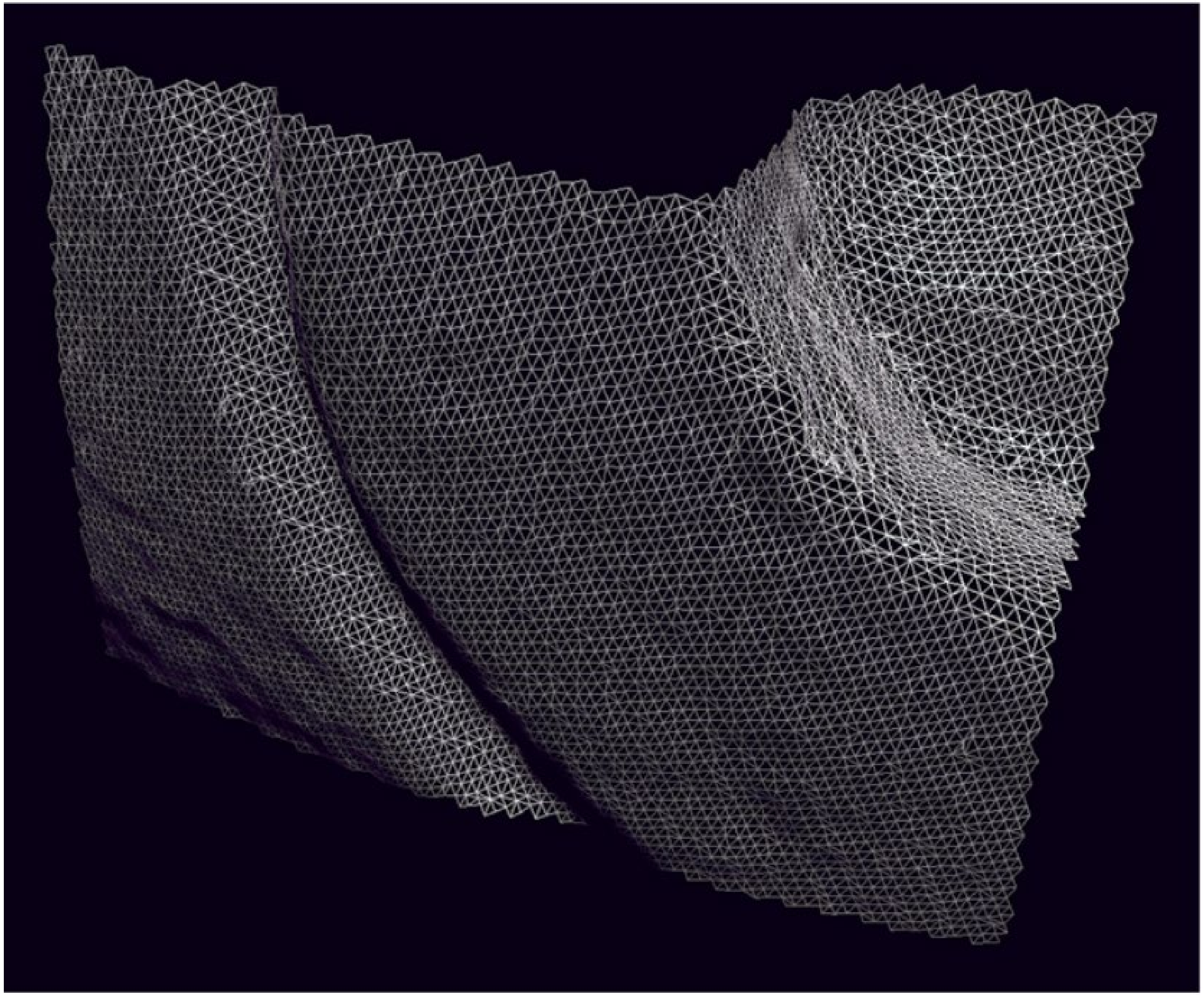
The Digital Michelangelo Project, Levoy et al.



The Digital Michelangelo Project, Levoy et al.



The Digital Michelangelo Project, Levoy et al.



The Digital Michelangelo Project, Levoy et al.

Binary coding

Faster Acquisition?

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

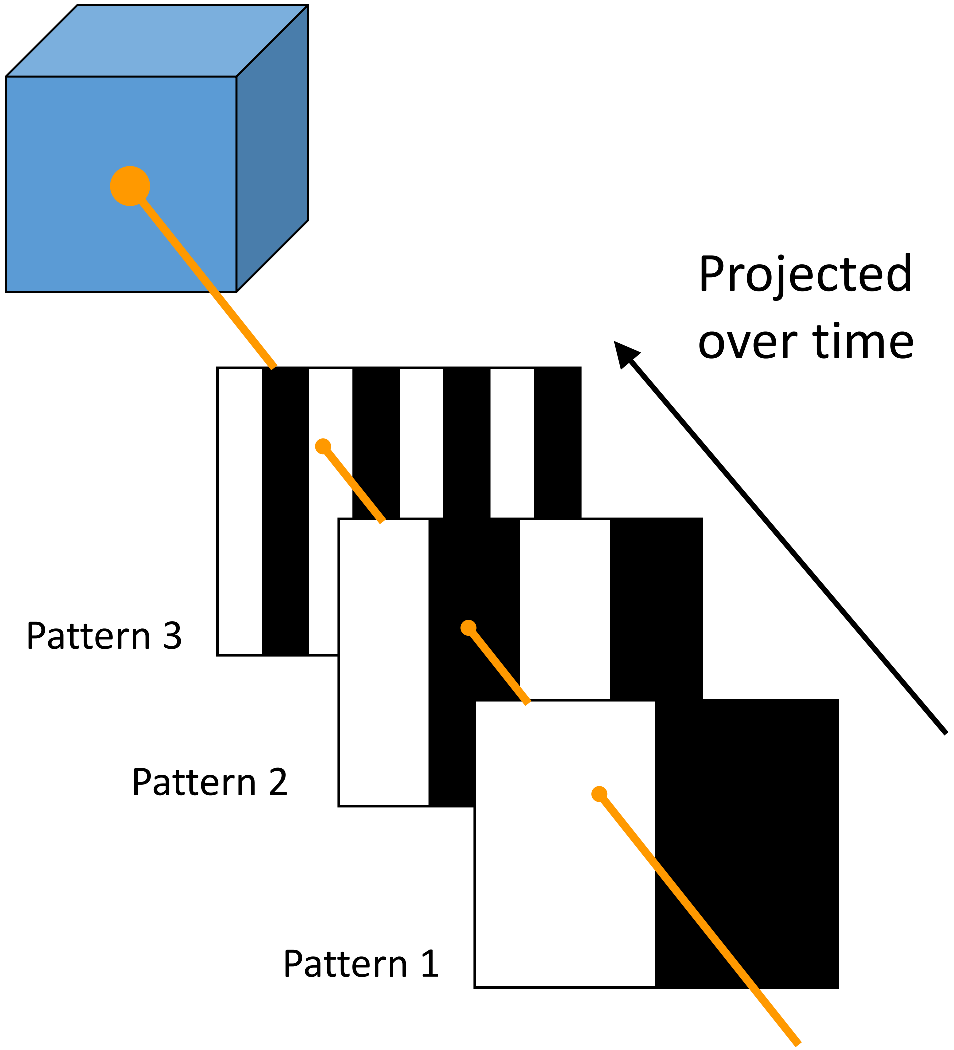
Binary Coding

Faster:

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

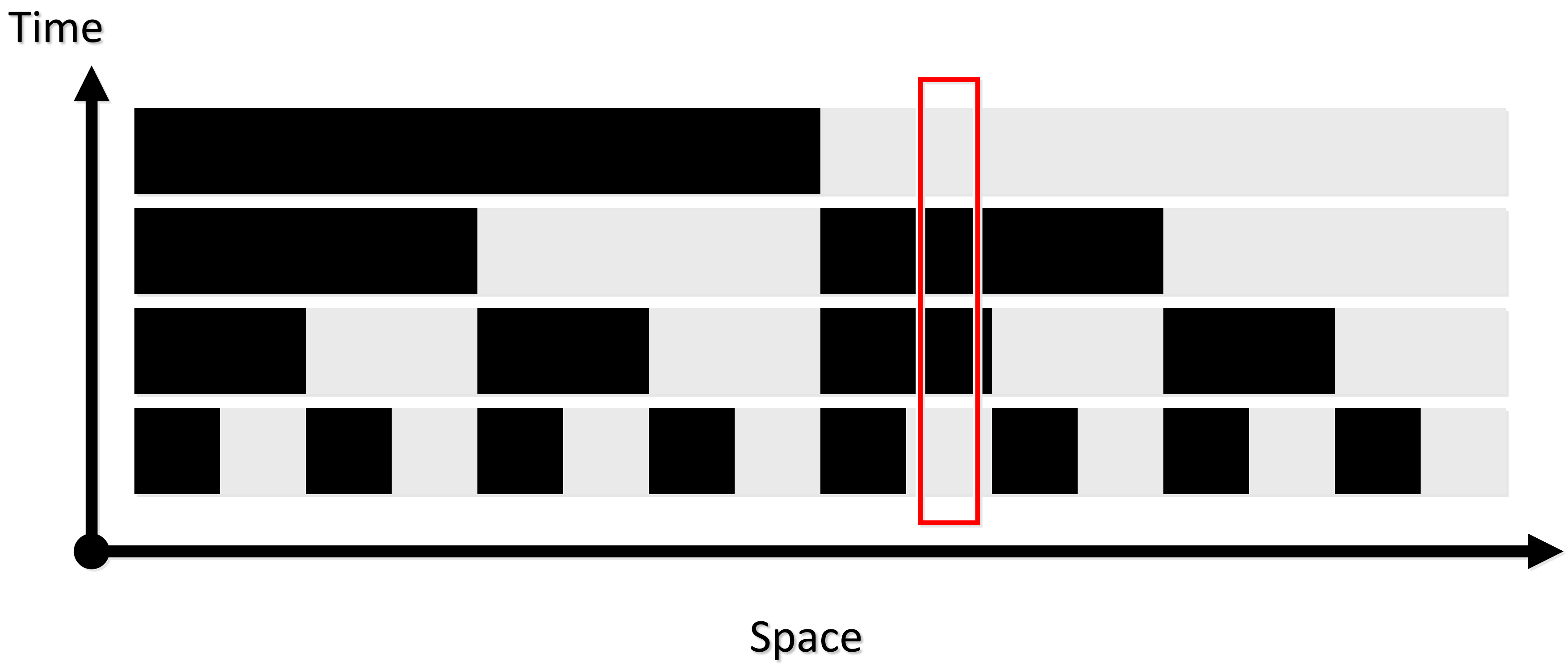
Example:

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



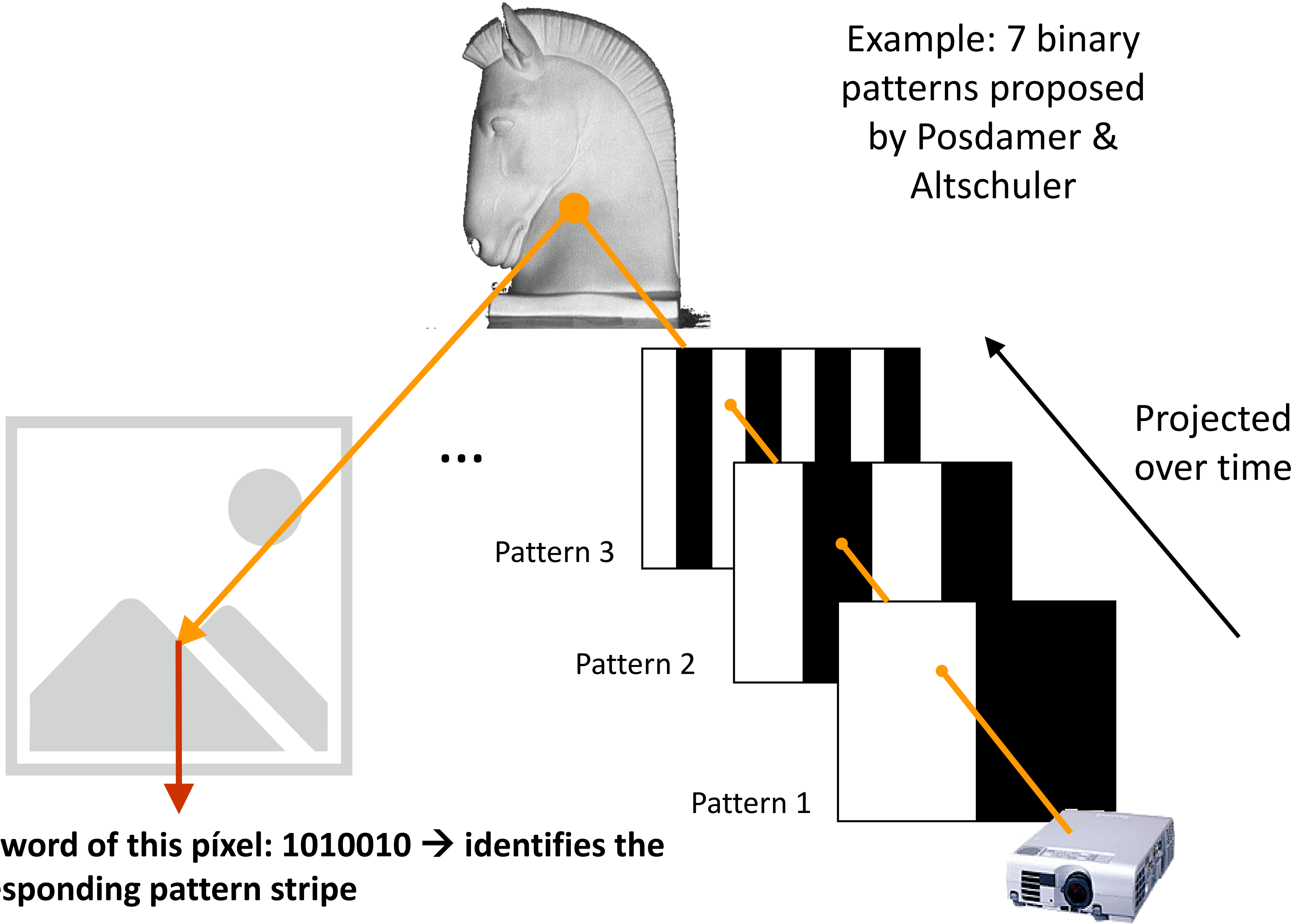
Binary Coding

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



Binary Coding

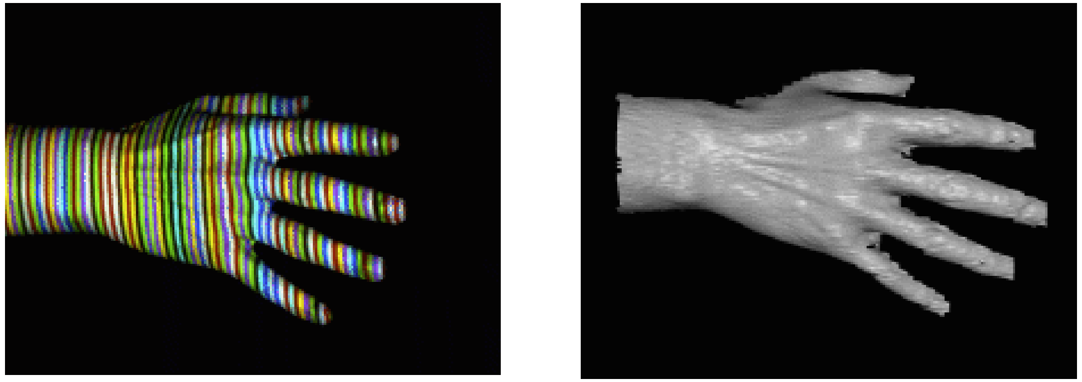
Example: 7 binary patterns proposed by Posdamer & Altschuler



Codeword of this píxel: 1010010 → identifies the corresponding pattern stripe

More complex patterns

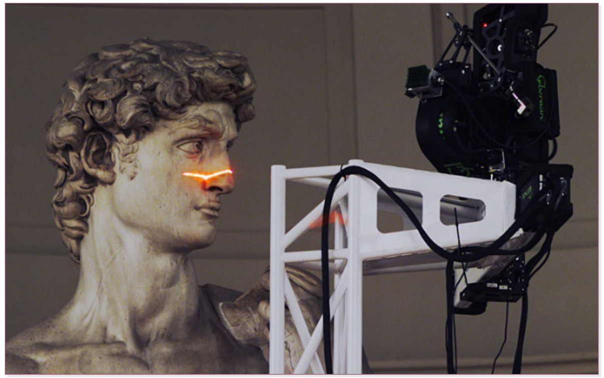
Works despite complex appearances

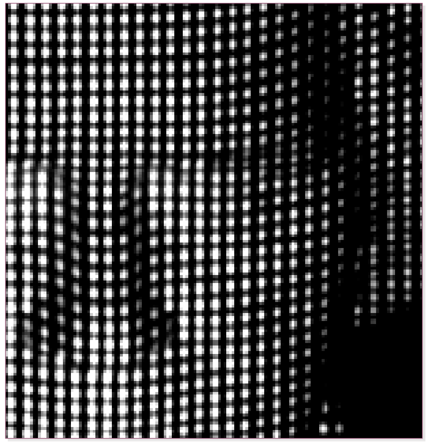


Works in real-time and on dynamic scenes

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

Continuum of Triangulation Methods

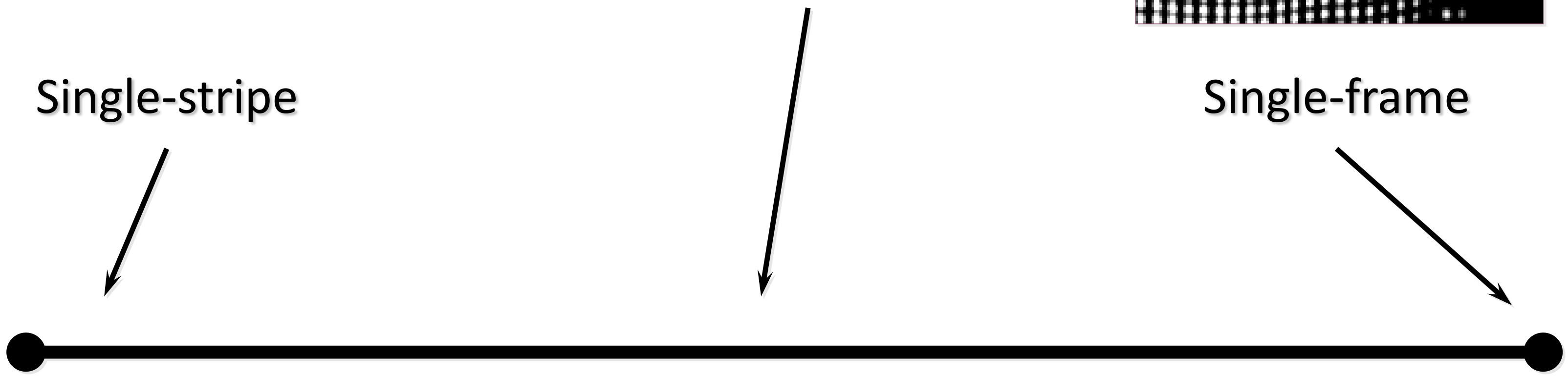




Multi-stripe
Multi-frame

Single-stripe

Single-frame

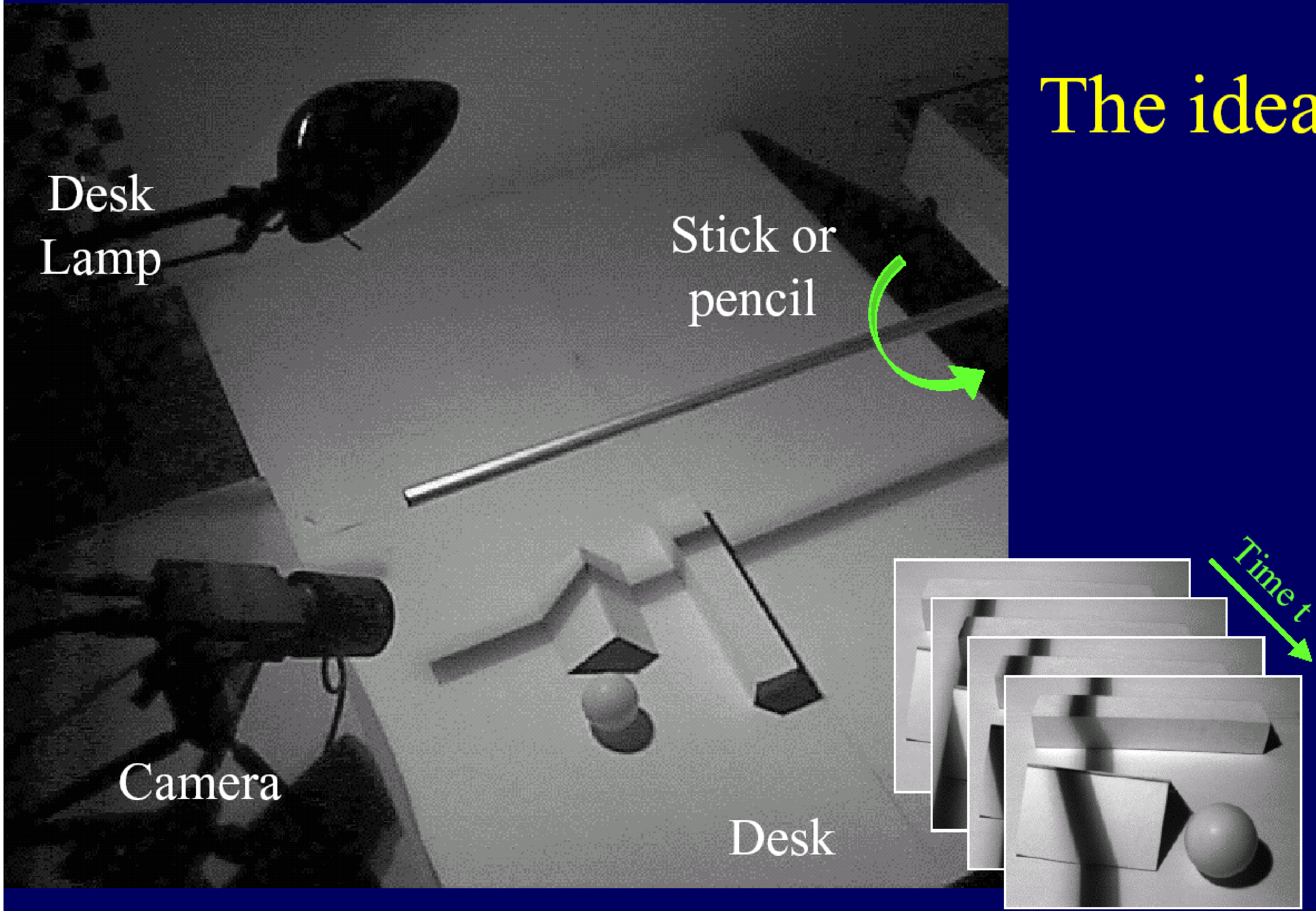


Slow, robust

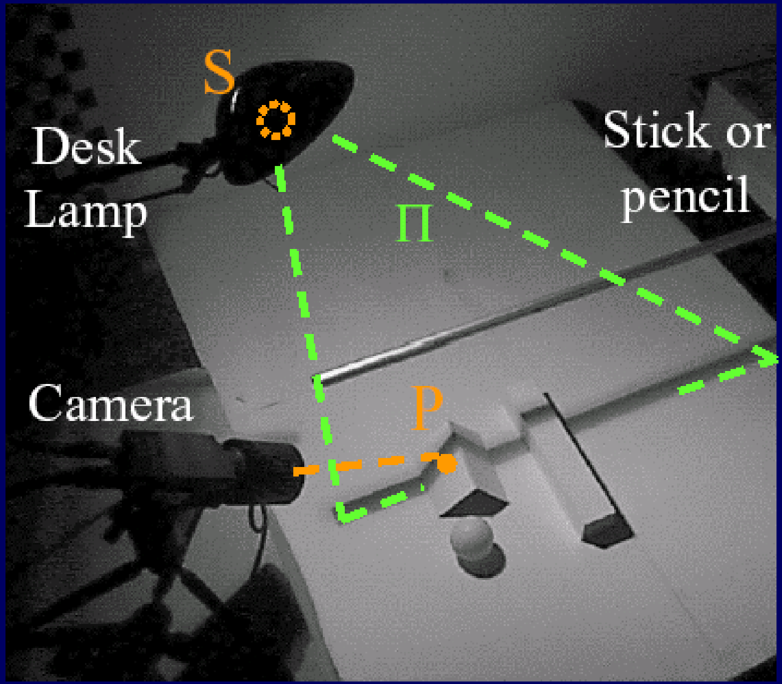
Fast, fragile

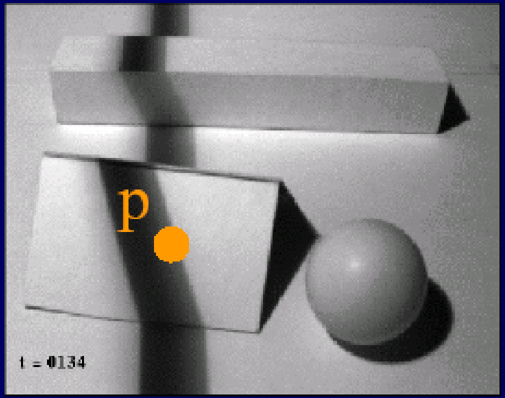
Using shadows

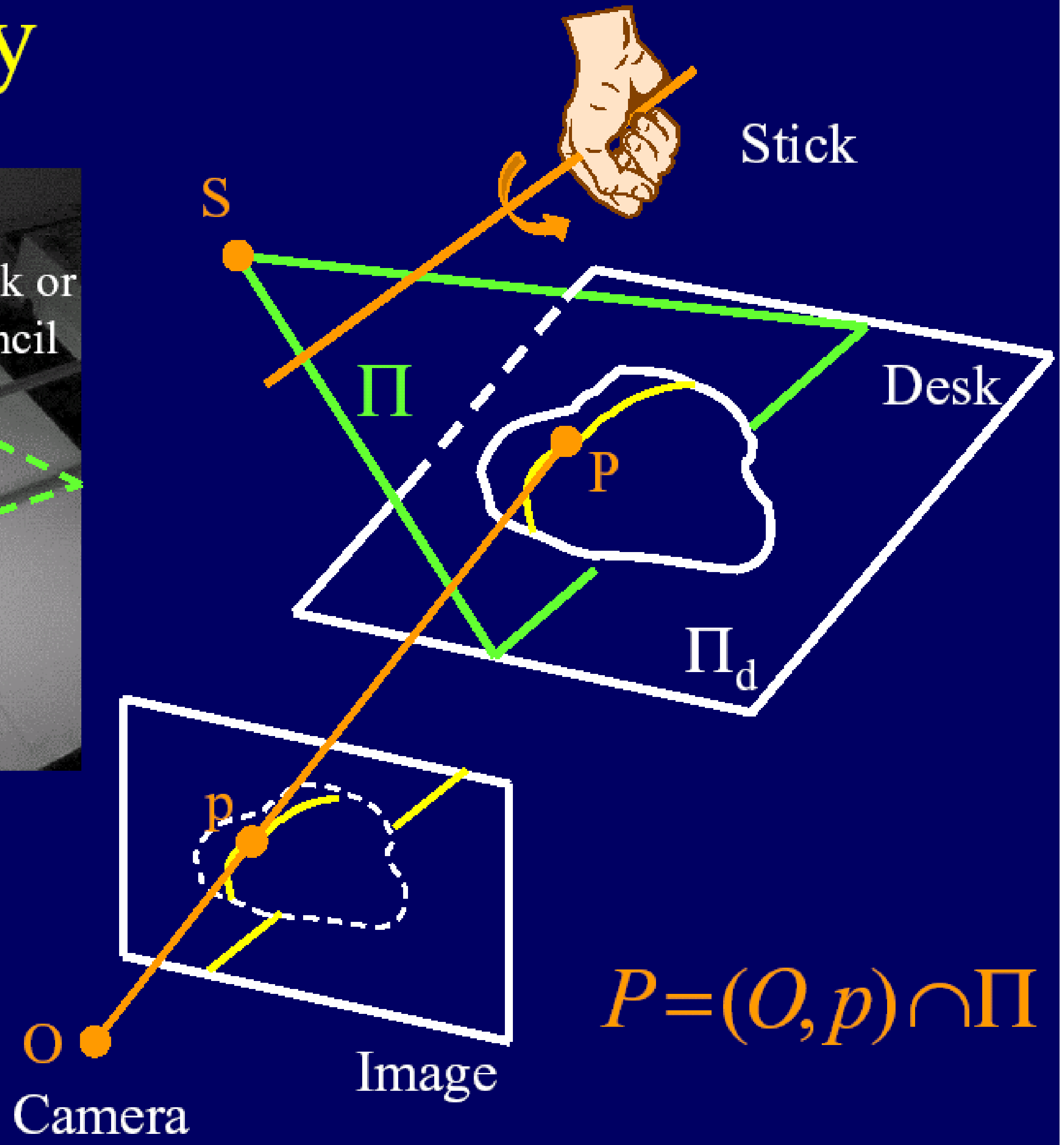
The idea



The geometry

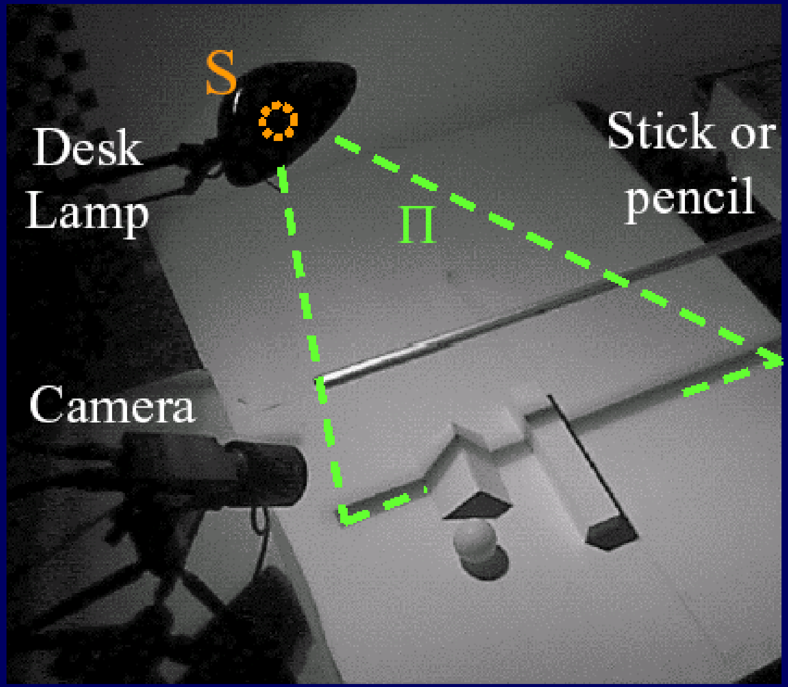


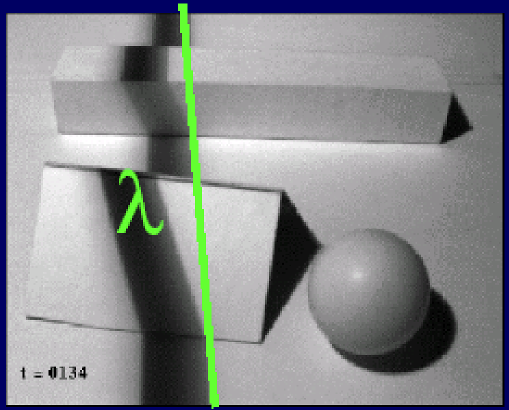


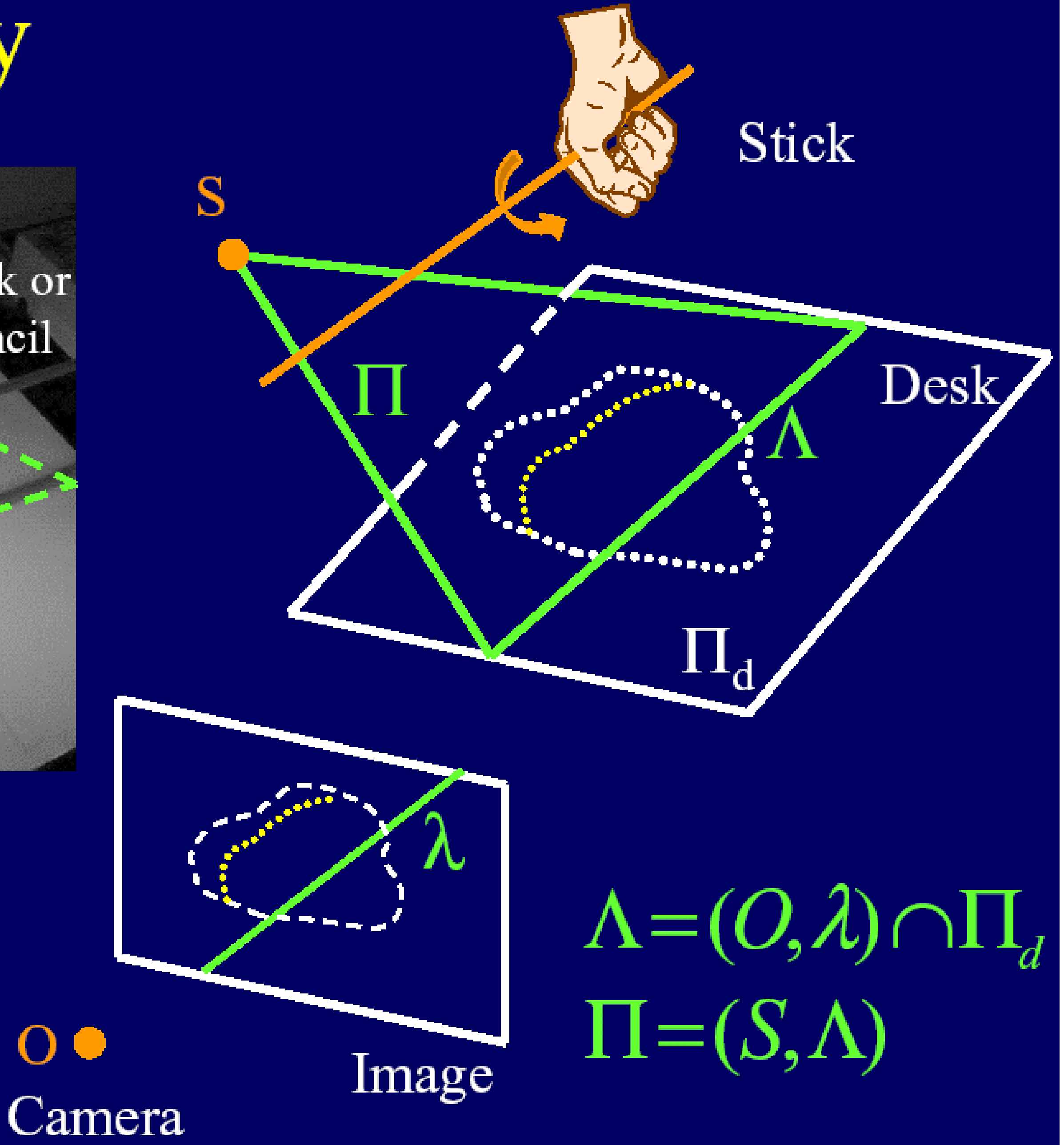


$$P = (O, p) \cap \Pi$$

The geometry



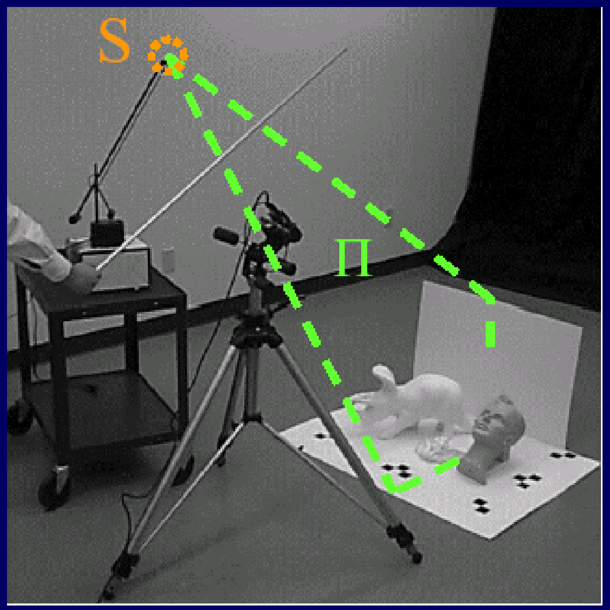


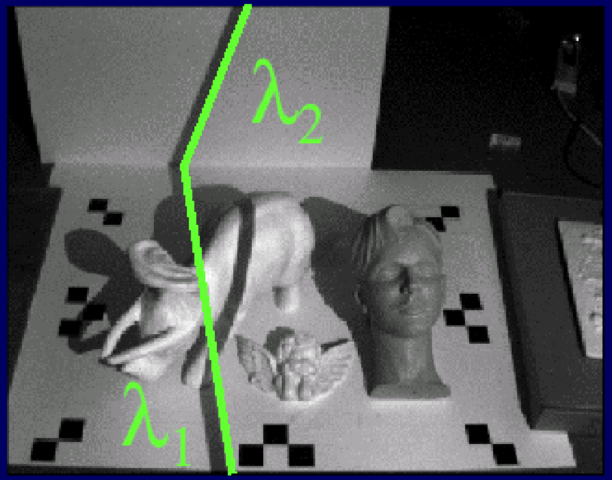


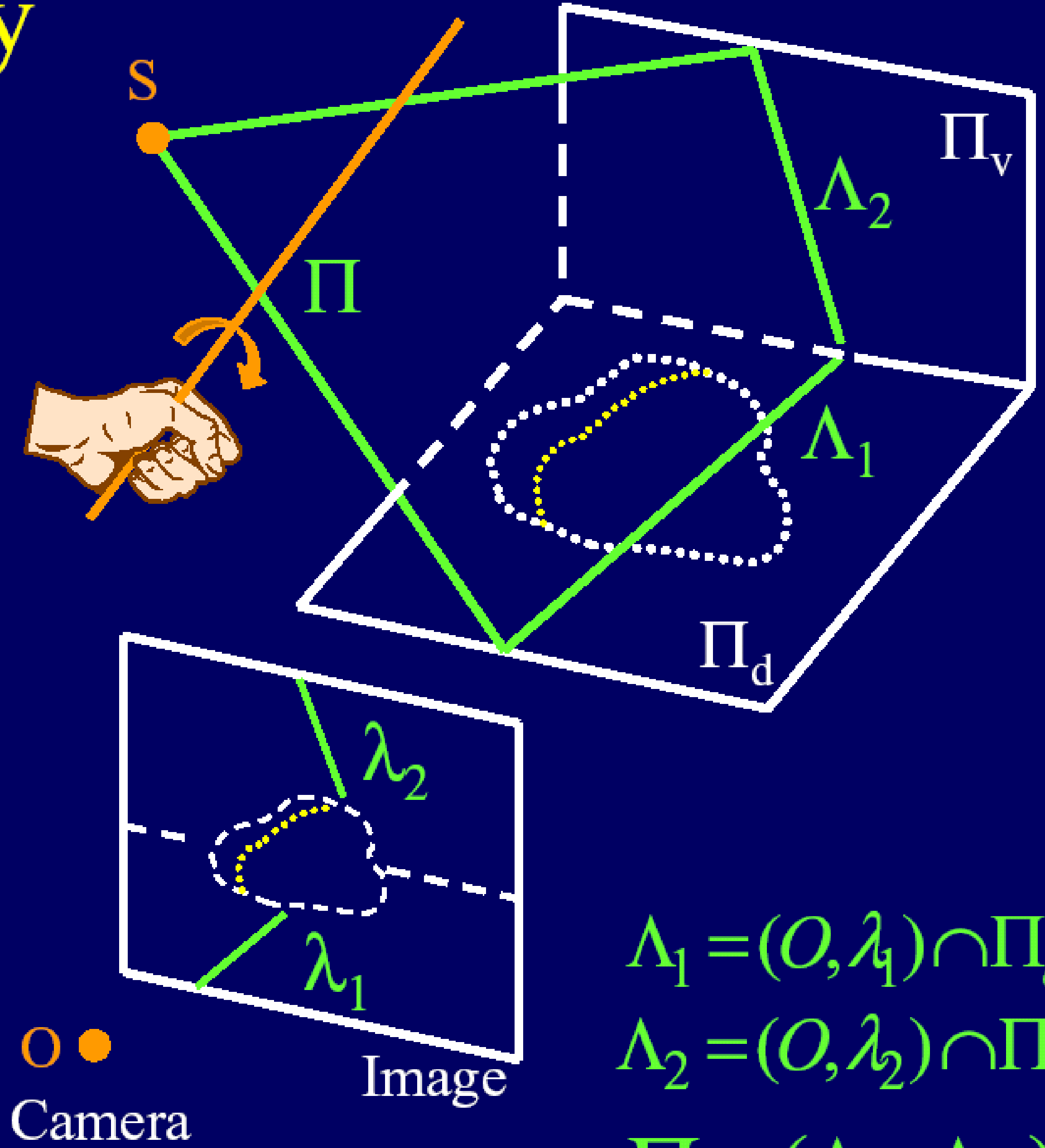
$$\Lambda = (O, \lambda) \cap \Pi_d$$

$$\Pi = (S, \Lambda)$$

The geometry





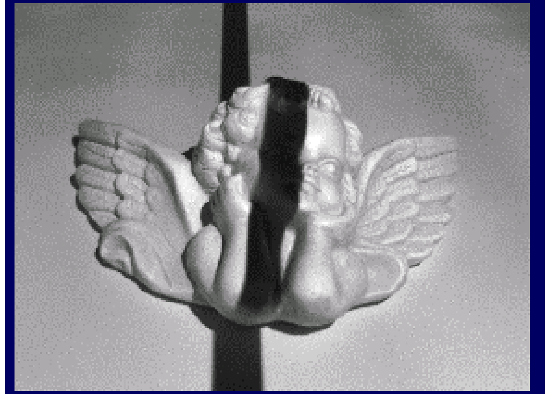


$$\Lambda_1 = (O, \lambda_1) \cap \Pi_d$$

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

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

Angel experiment

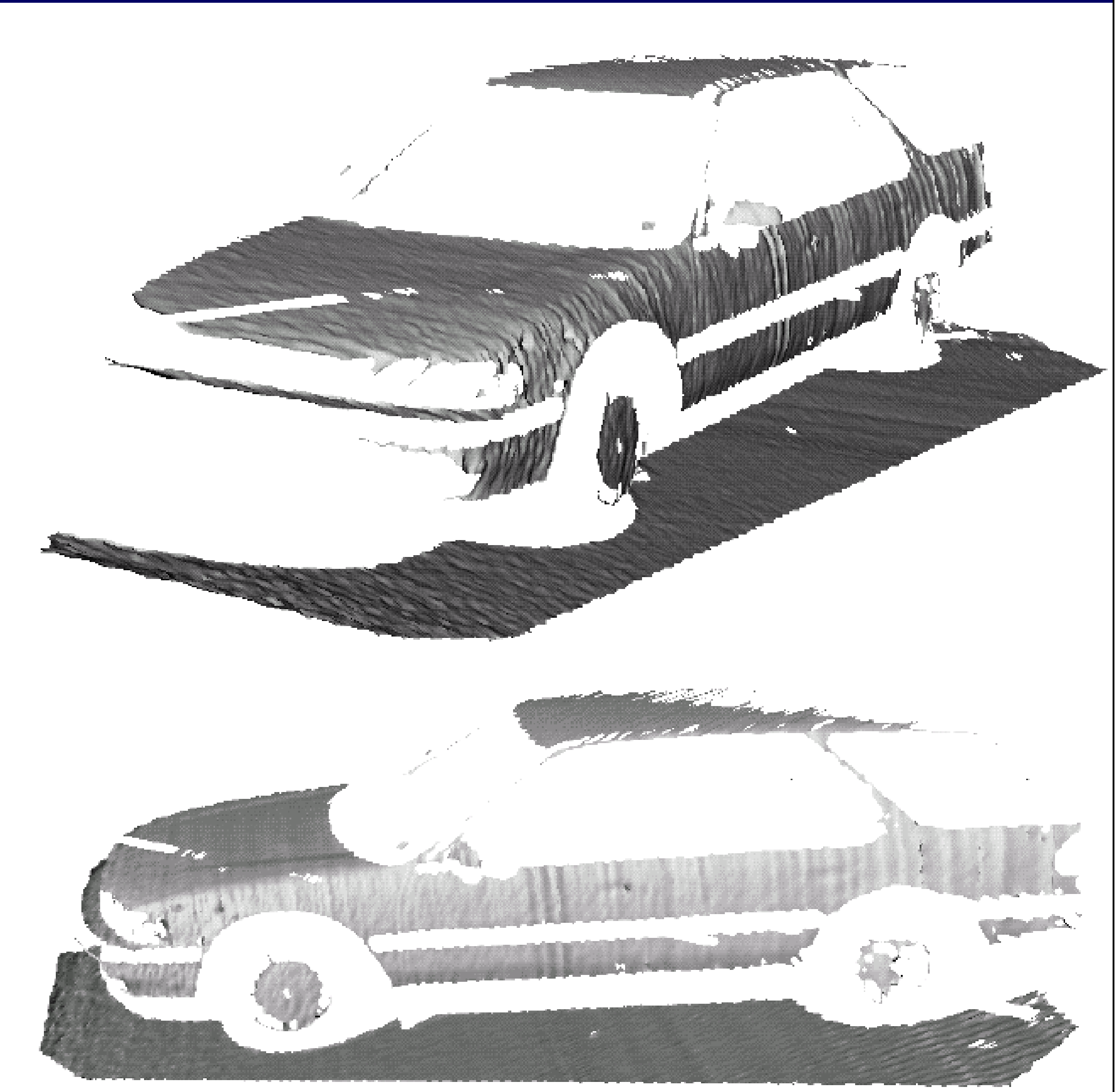


Accuracy: 0.1mm over 10cm → ~ 0.1% error

Scanning with the sun

Accuracy: 1cm over 2m

→ ~ 0.5% error



References

Basic reading:

- Szeliski textbook, Sections 7.1, 8.1, Chapter 11, Sections 11.1, 12.2.
- Hartley and Zisserman, Section 11.12.
- 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 problems.
- 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.
- Garg et al., “Learning Single Camera Depth Estimation using Dual-Pixels,” ICCV 2019.
- Xin et al., “Defocus Map Estimation and Deblurring from a Single Dual-Pixel Image,” ICCV 2021.
  - These two papers discuss the relationship between focusing and depth estimation, and how to apply it in the context of dual-pixel cameras.