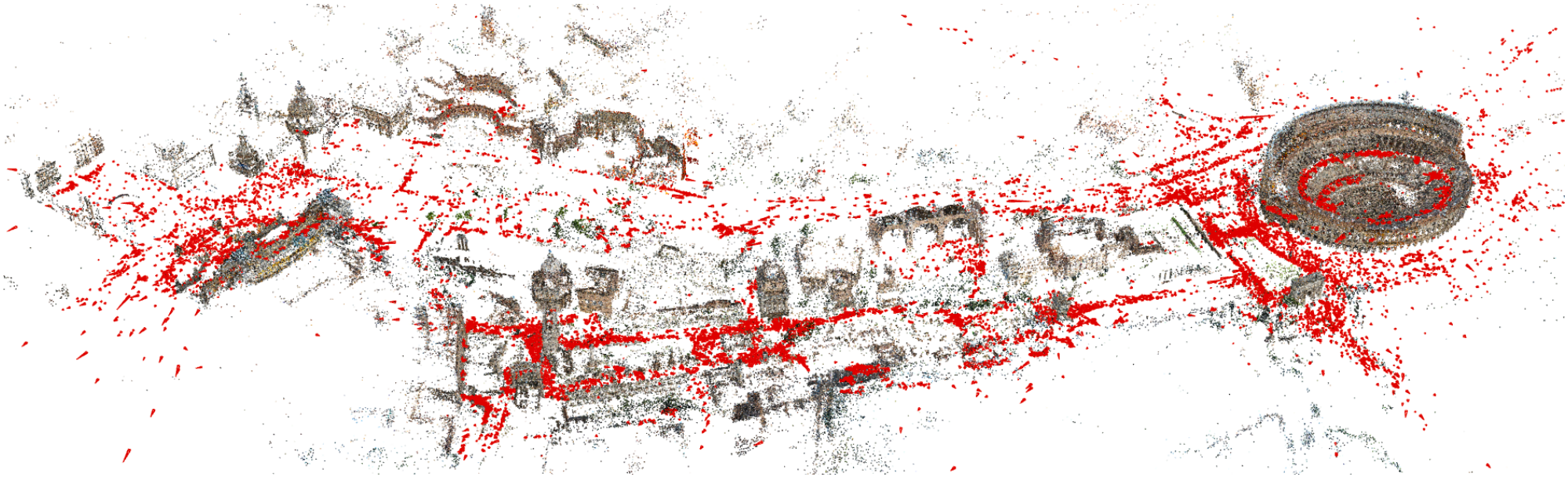
3D Vision: Calibration, Stereo



A lot of slides borrowed from Noah Snavely + Shree Nayar's YT series: First principals of Computer Vision

CS194: Intro to Computer Vision and Comp. Photo

Angjoo Kanazawa, UC Berkeley, Fall 2021

Breaking out of 2D

…now we are ready to break out of 2D

And enter the real world!

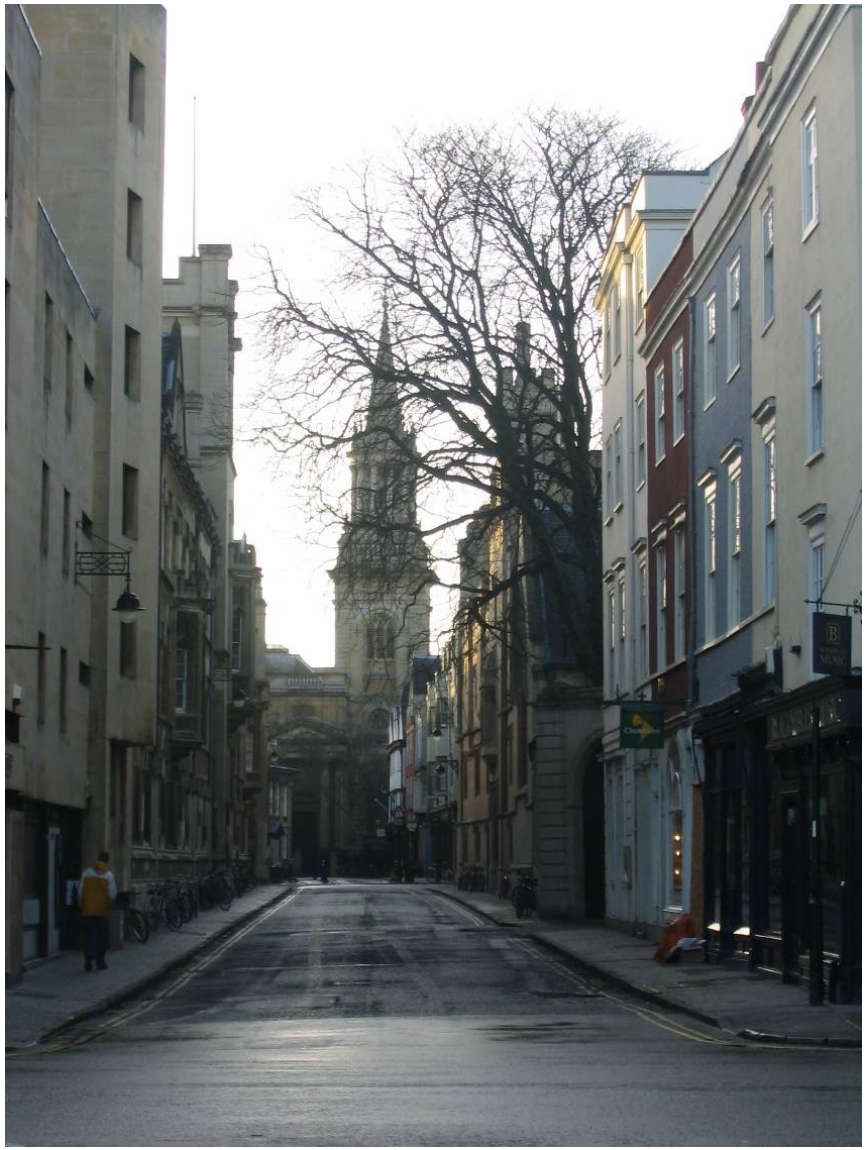
on to 3D…

Enough of images!

We want more of the plenoptic function

We want real 3D scene walk-throughs:

- Camera rotation
- Camera translation



3D is super cool!

https://rd.nytimes.com/projects/reconstructing-journalistic-scenes-in-3d

3D is super cool!

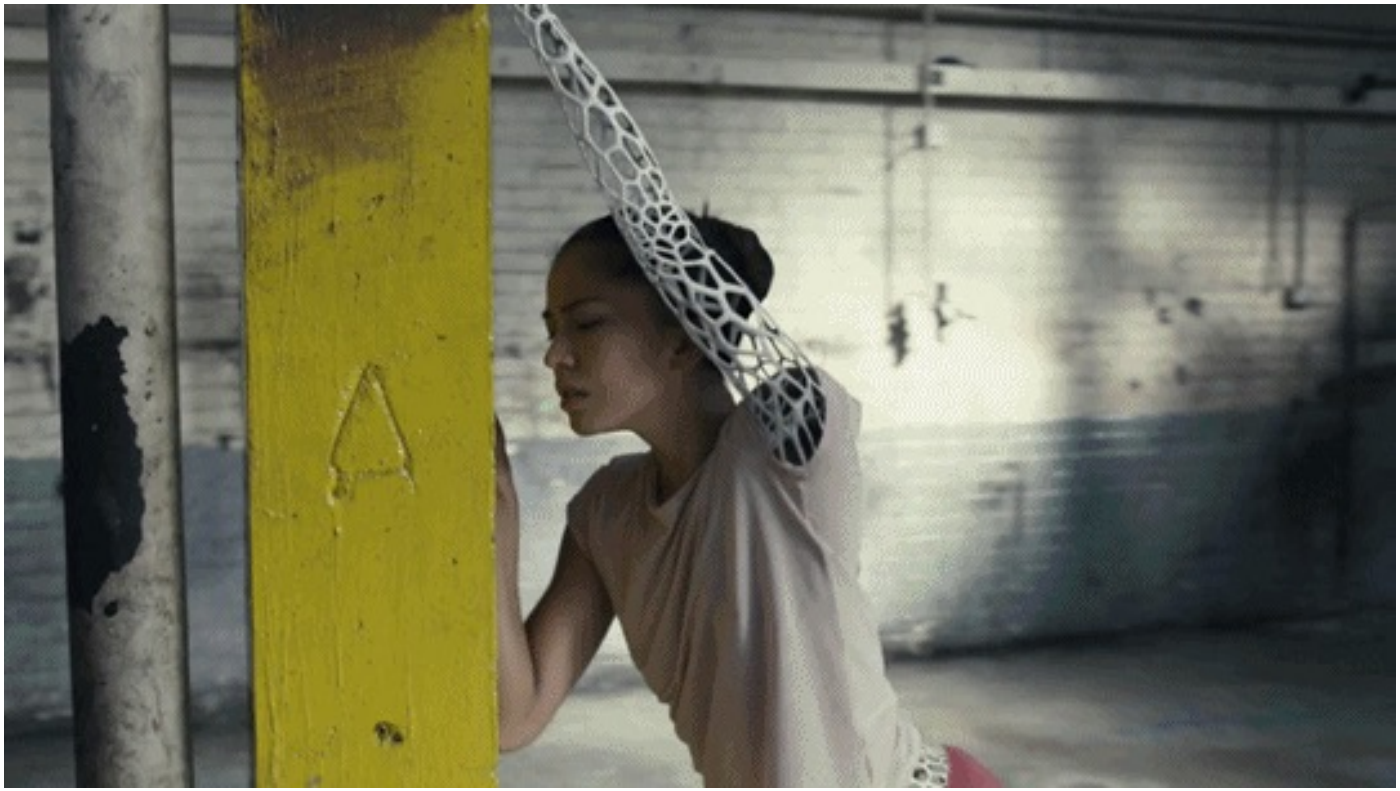
@capturingreality

@organiccomputer

NeRF in the wild (will get to in few more lectures)

NeRF in the Wild, Martin-Brualla, Radwan et al. CVPR 2021

Not just about 3D reconstruction



[The Chemical Brothers - Wide Open ft. Beck, MV]

3D for video editing

My Research

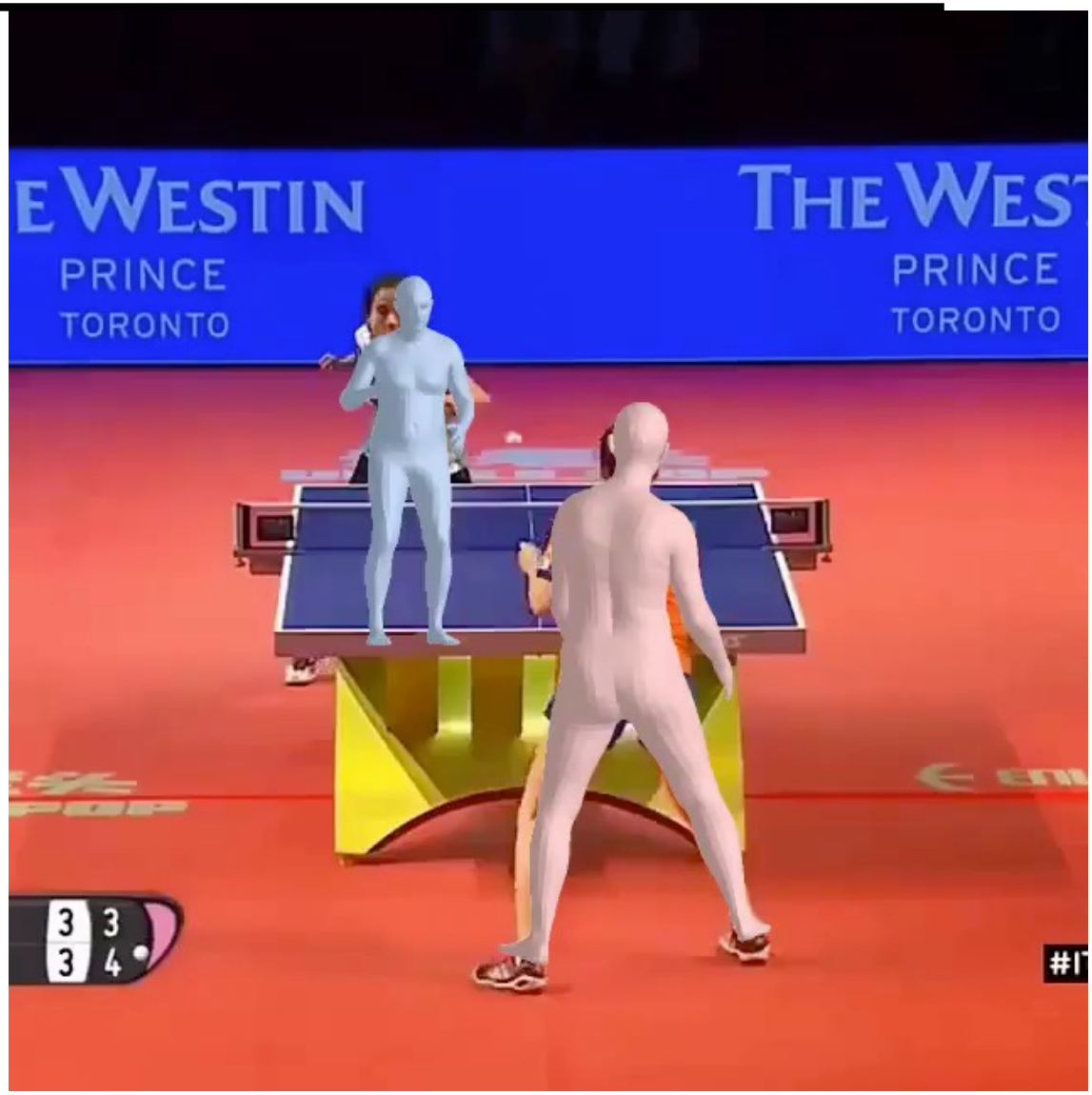
Single-View 3D Human Mesh Recovery

[Bogo∗, **Kanazawa**∗, Lassner, Gehler, Romero, Black ECCV ’16]

In everyday photos

Kanazawa, Black, Jacobs, Malik. CVPR 2018

Or from Video



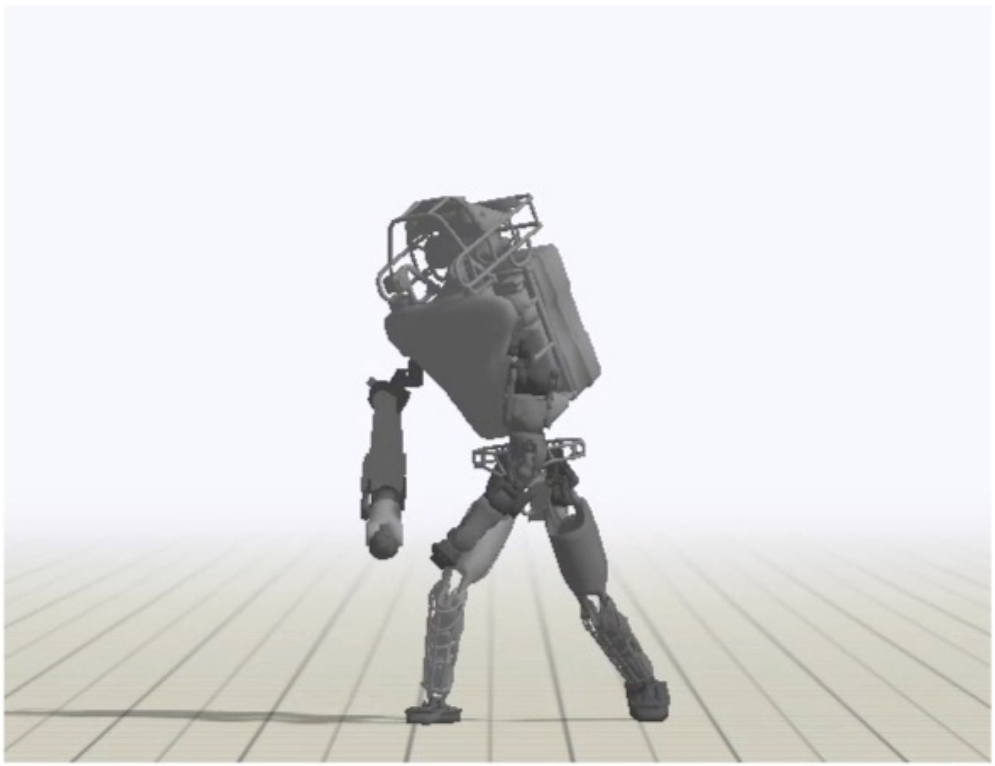
Kanazawa, Zhang, and Felsen et al. CVPR 2019

In more detail

Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization,
Saito, Huang, Natsume, Morishima, **Kanazawa**, Li, ICCV 2019

Teaching robots how to dance from watching YouTube

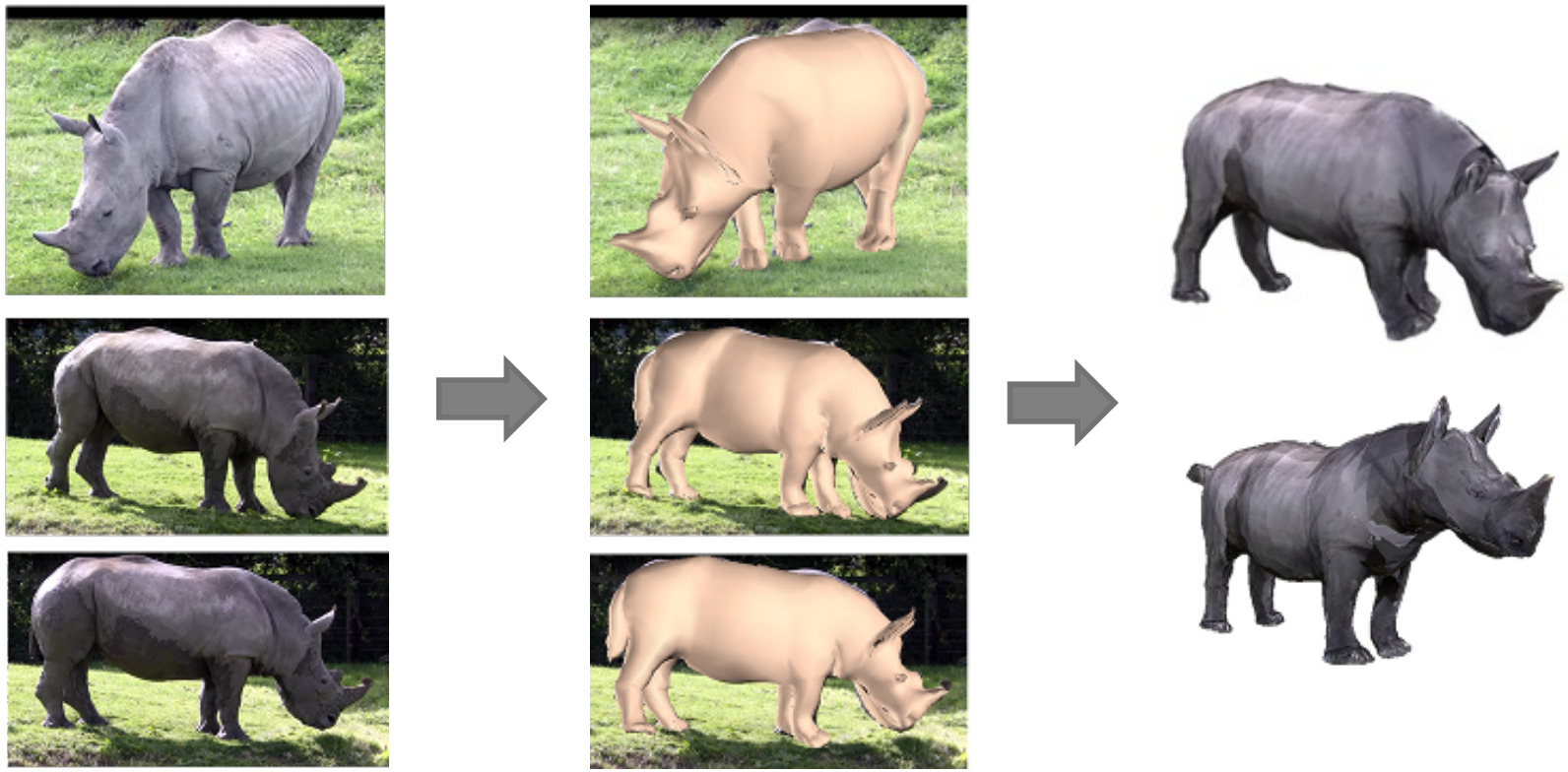
Video



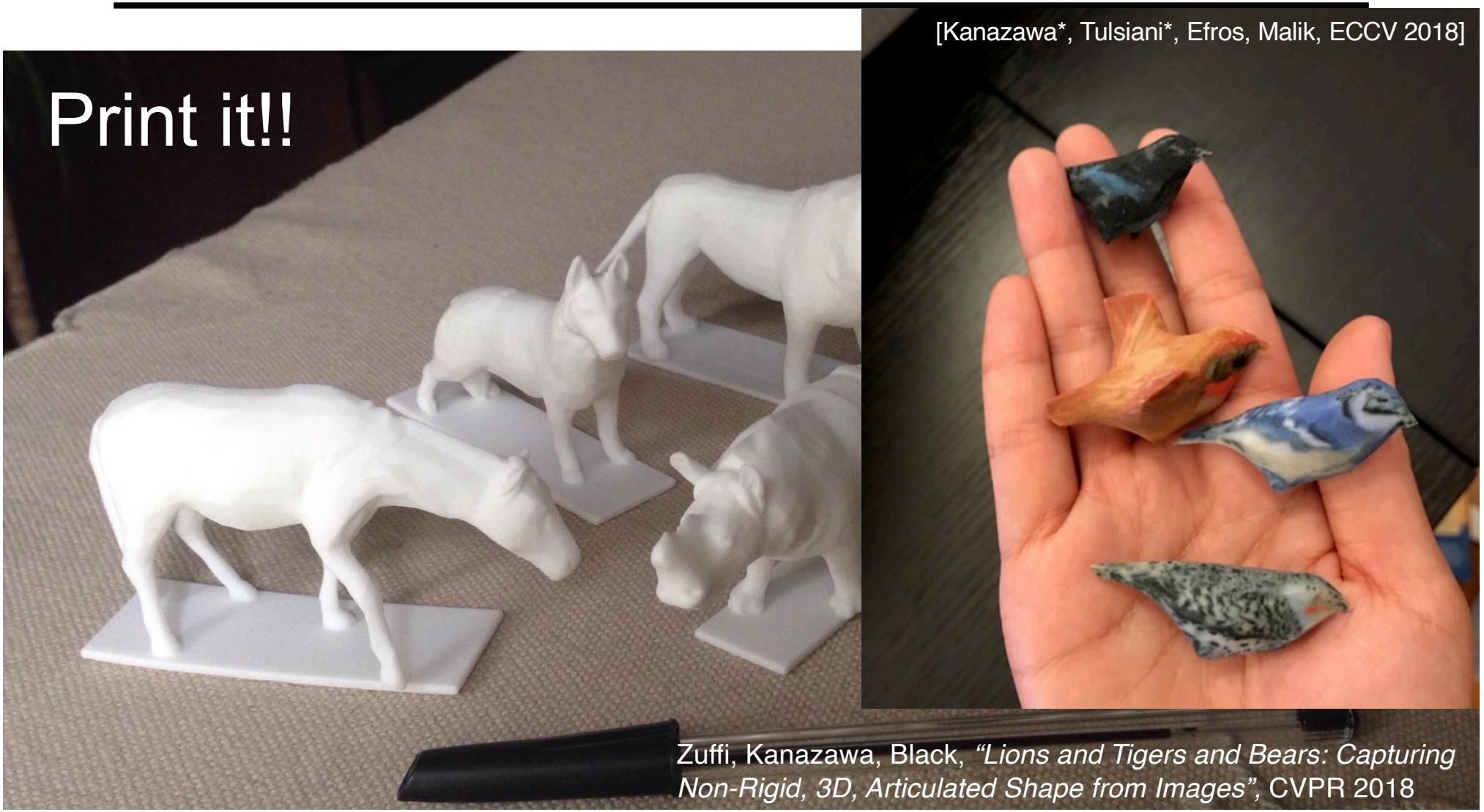
Policy

Peng, Kanazawa, Malik, Abbeel, Levine
"SFV: Reinforcement Learning of Physical Skills from Videos", SIGGRAPH Asia 2018

Reconstructing Animals with Human Input



Zuffi, Kanazawa, Black, *"Lions and Tigers and Bears: Capturing Non-Rigid, 3D, Articulated Shape from Images"*, CVPR 2018


Print it!!
[Kanazawa*, Tulsiani*, Efros, Malik, ECCV 2018]
Zuffi, Kanazawa, Black, *"Lions and Tigers and Bears: Capturing Non-Rigid, 3D, Articulated Shape from Images"*, CVPR 2018

Flying into an image

Infinite Nature: Perpetual View Generation of Natural Scenes from a Single Image, ICCV 2021

so on to 3D…

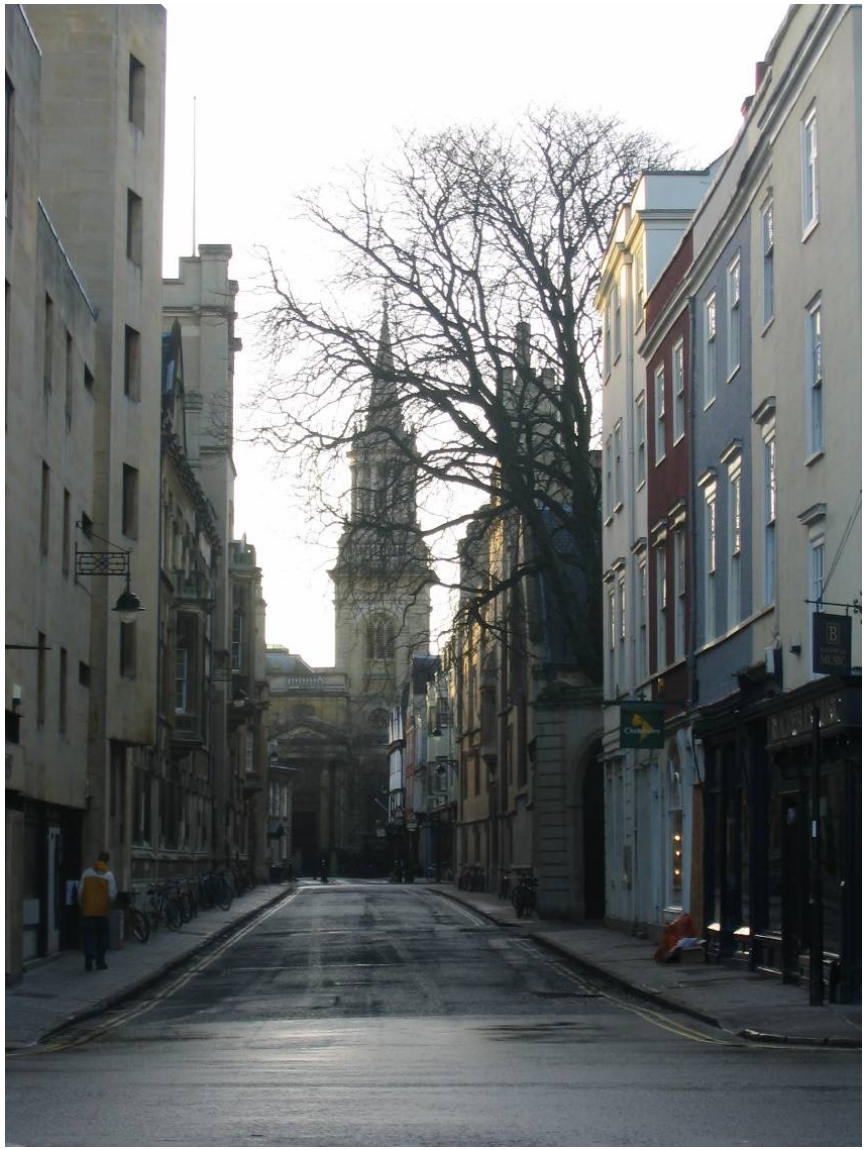
Enough of images!

We want more of the plenoptic function

We want real 3D scene walk-throughs:

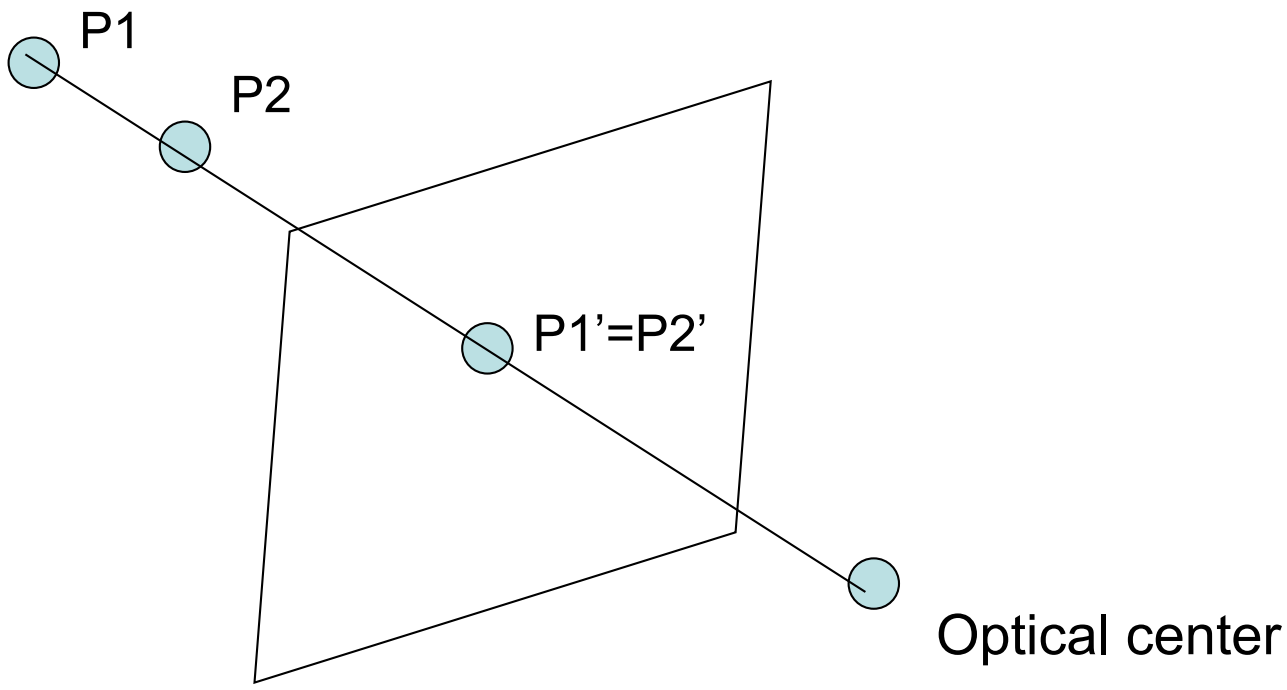
- Camera rotation
- Camera translation

Can we do it from a single photograph?



Why multiple views?

- Structure and depth are inherently ambiguous from single views.



Why multiple views?

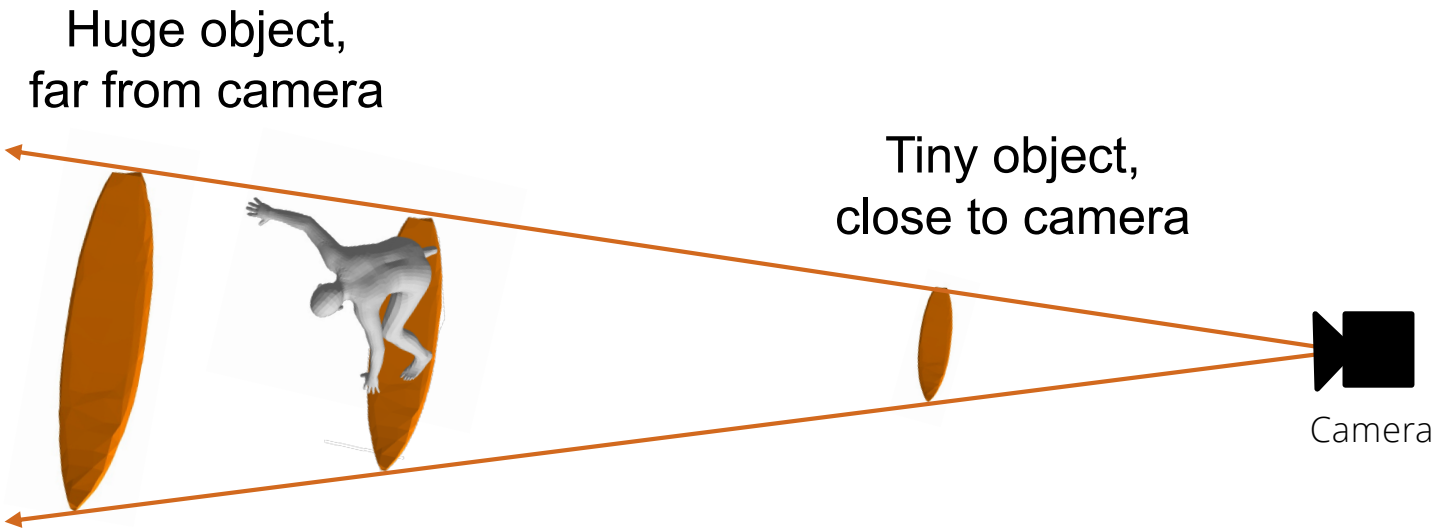
- Structure and depth are inherently ambiguous from single views.

Images from Lana Lazebnik

Fundamental Scale Ambiguity of 2D → 3D

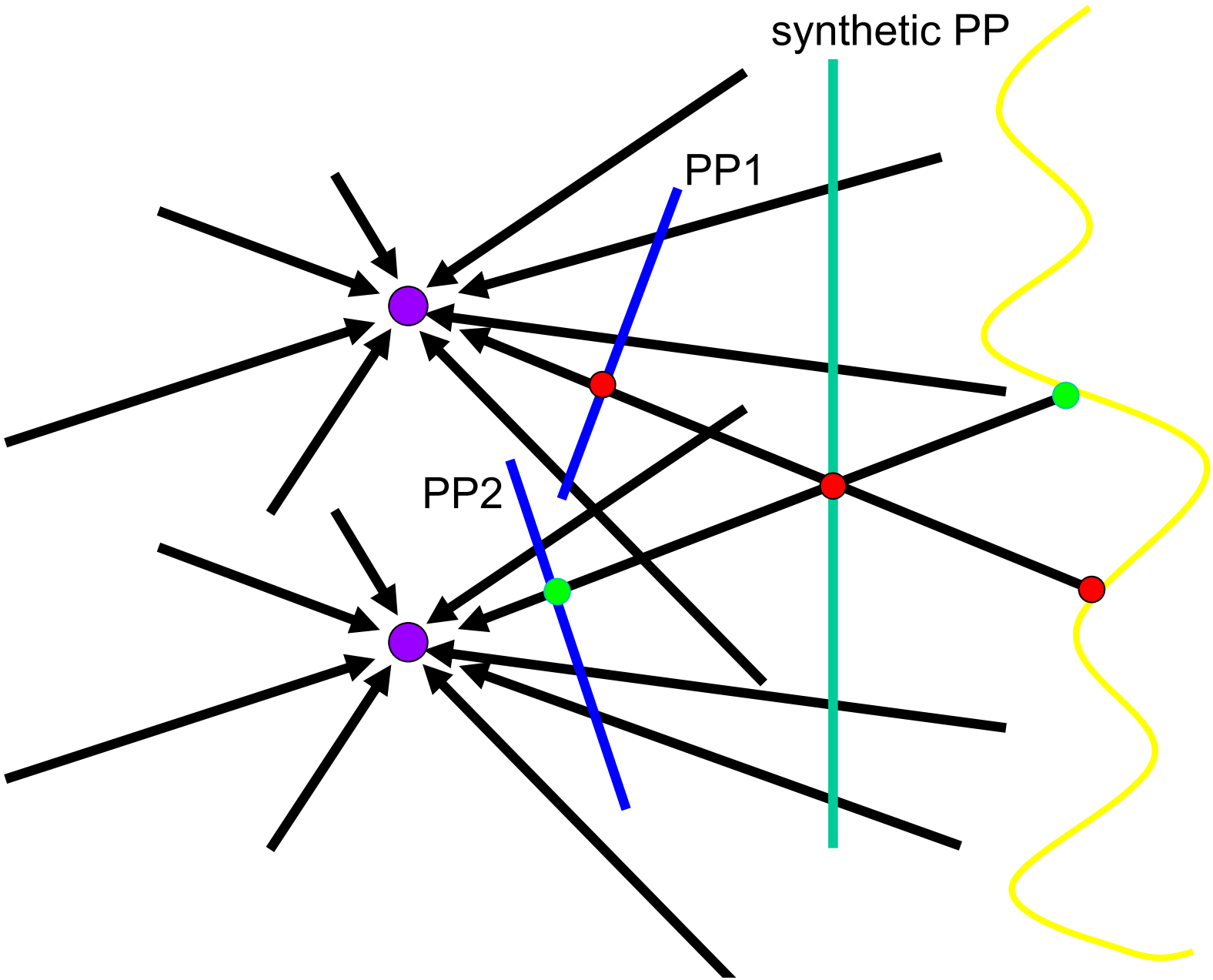
Original Image

Same 2D Projection



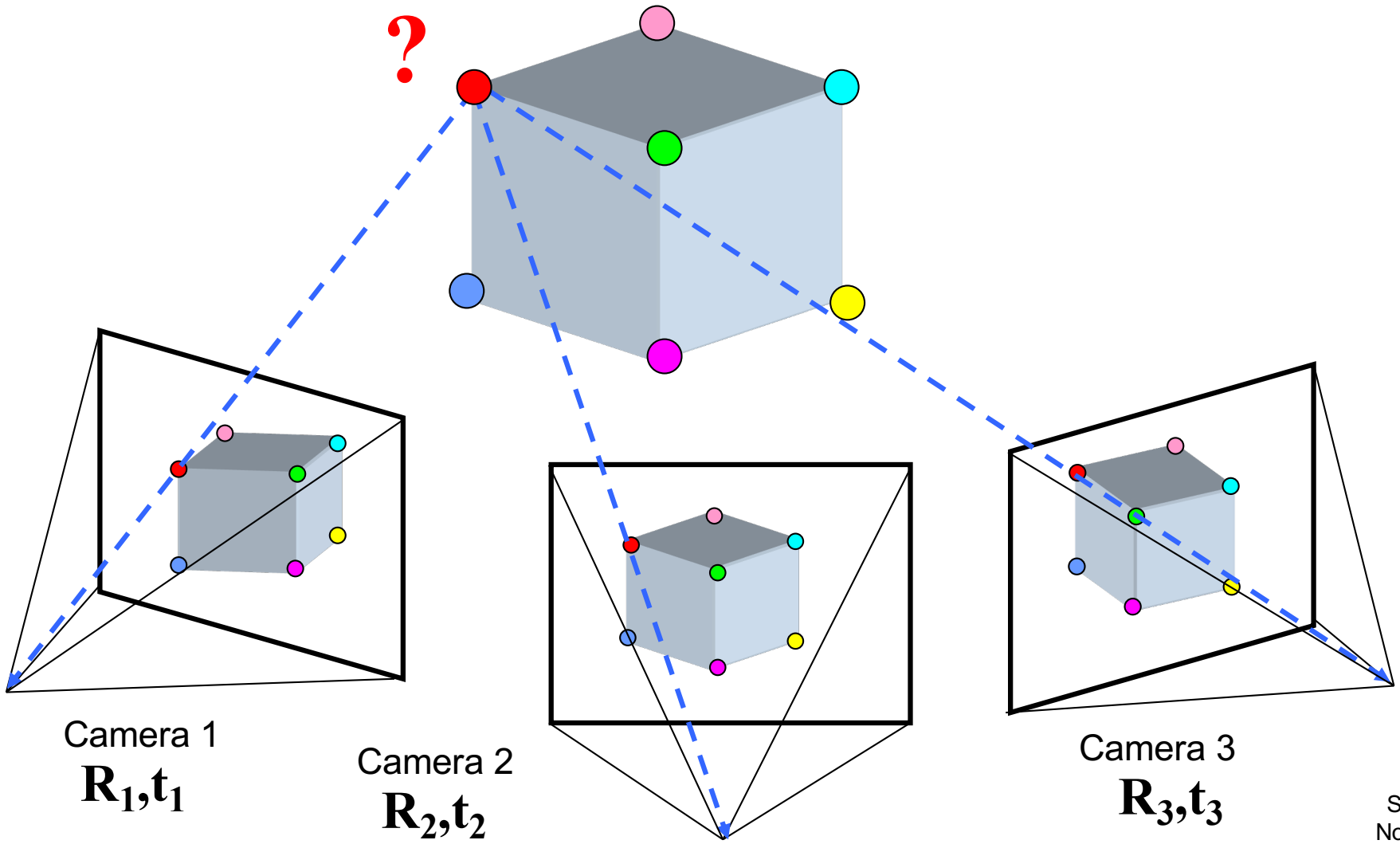
Infinite Possible 3D Interpretations

Need to change the camera center



Multi-view geometry problems

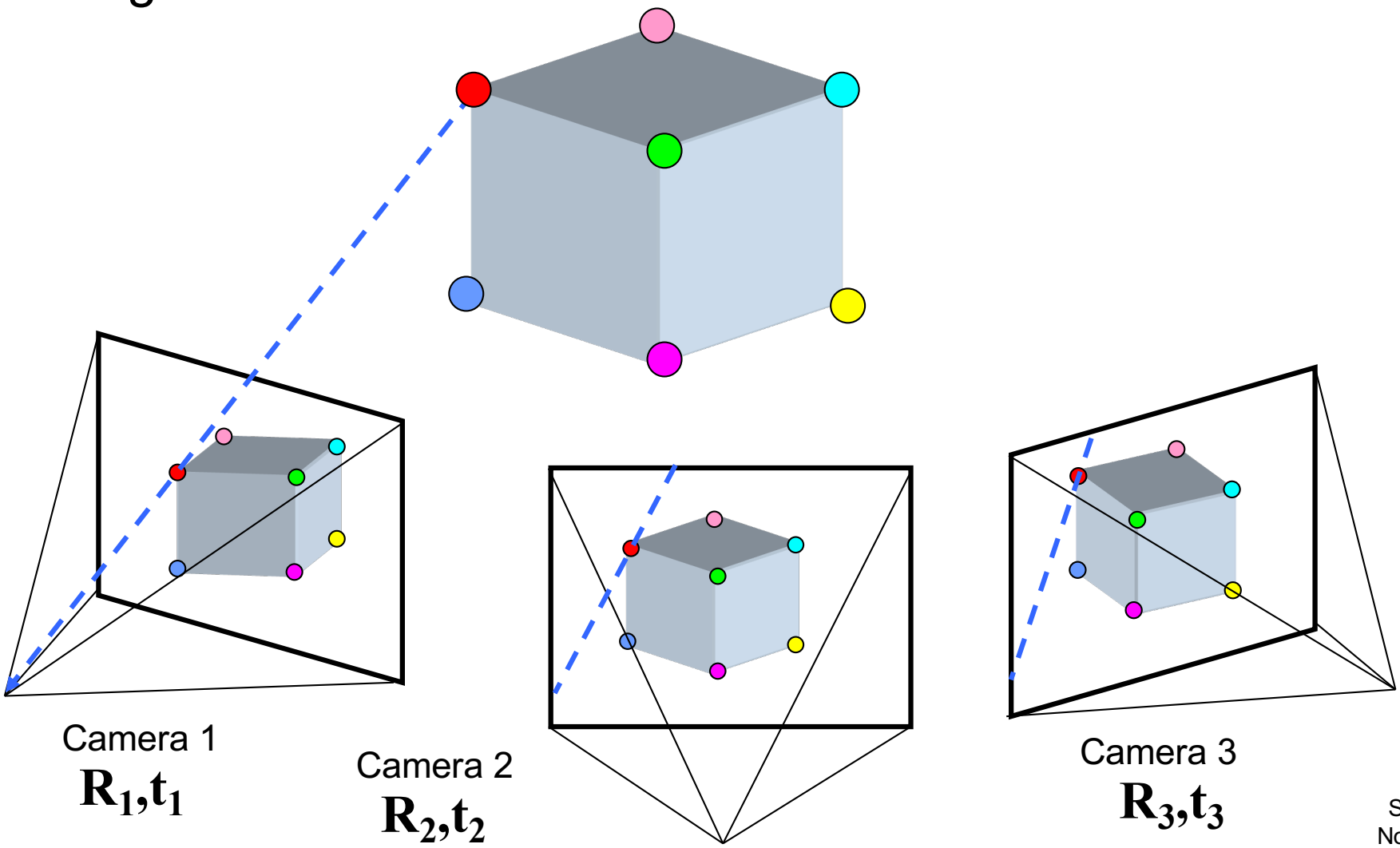
- **Structure:** Given projections of the same 3D point in two or more images, compute the 3D coordinates of that point



Slide credit:
Noah Snavely

Multi-view geometry problems

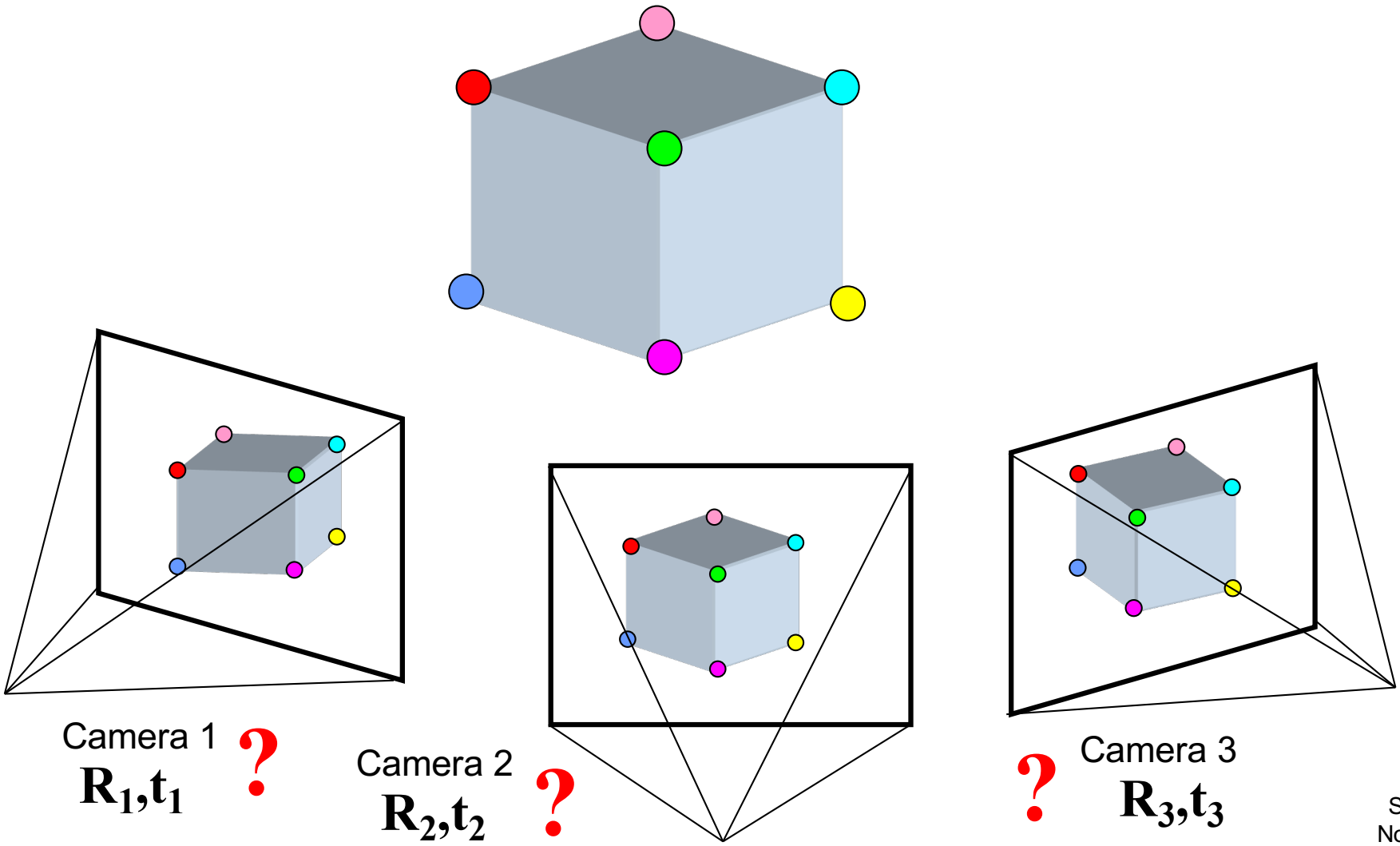
- **Correspondence:** Given a point in one of the images, where could its corresponding points be in the other images?



Slide credit:
Noah Snavely

Multi-view geometry problems

- **Motion:** Given a set of corresponding points in two or more images, compute the camera parameters



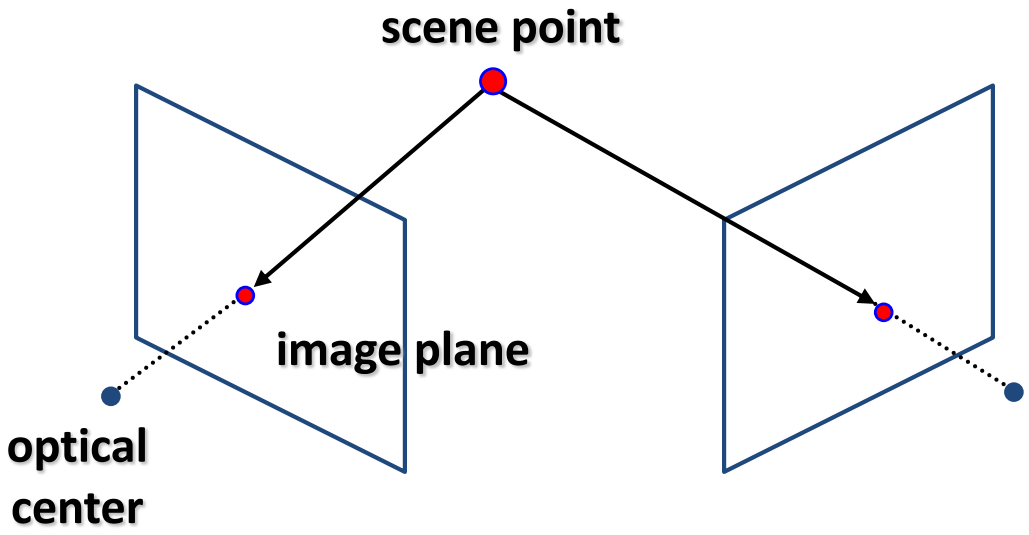
Slide credit:
Noah Snavely

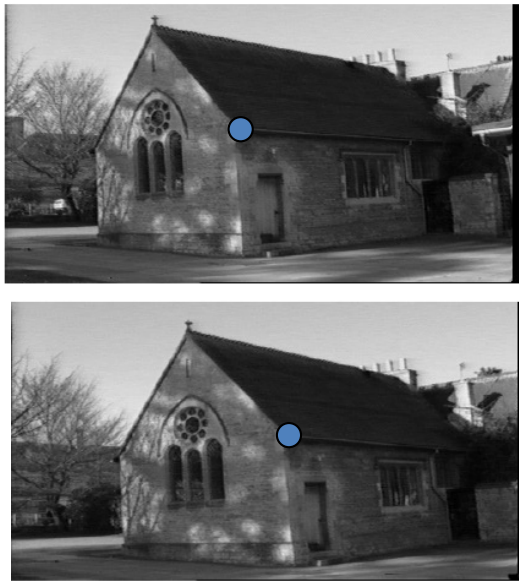
Today

- **Two** camera system = Stereo
- Calibrating the cameras
- Estimating depth from correspondences

Estimating depth with stereo

- **Stereo**: shape from “motion” between **two** views
- We’ll need to consider:
  - 1. Info on camera pose (“calibration”)
  - 2. Image point correspondences





Stereo vision

Two cameras, simultaneous views

Single moving camera and static scene

Cameras in world coordinate frame

We only have images and pixels

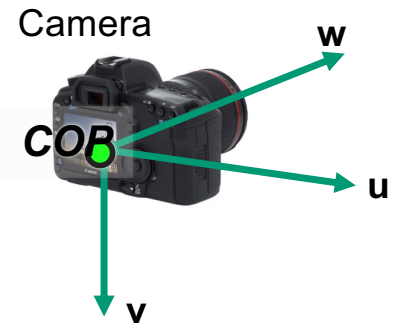
To go from pixels to 3D location in the world, we need to know two things about the camera:

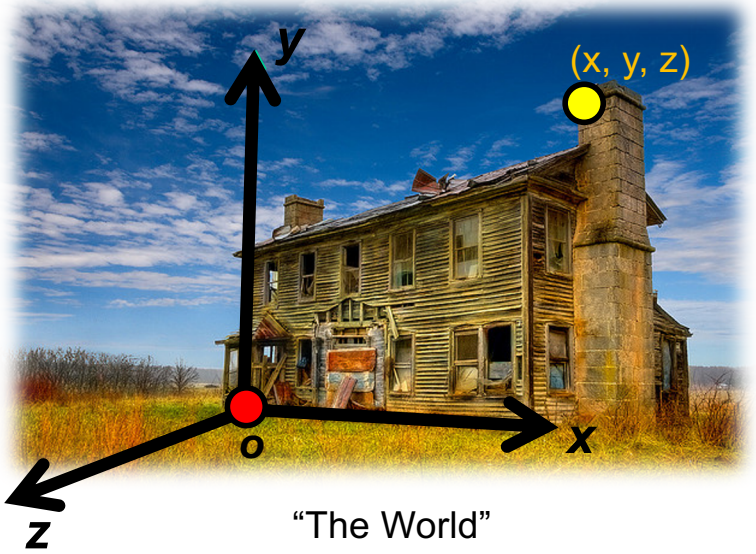
1. Position of the camera with respect to the world (extrinsics)
2. How the camera maps a point in the world to image (intrinsics)

Problem setup

There is a world coordinate frame and camera looking at the world

How can we model the geometry of a camera?





"The World"

Three important coordinate systems:

1. *World* coordinates
2. *Camera* coordinates
3. *Image* coordinates

Slide credit: Noah Snavely

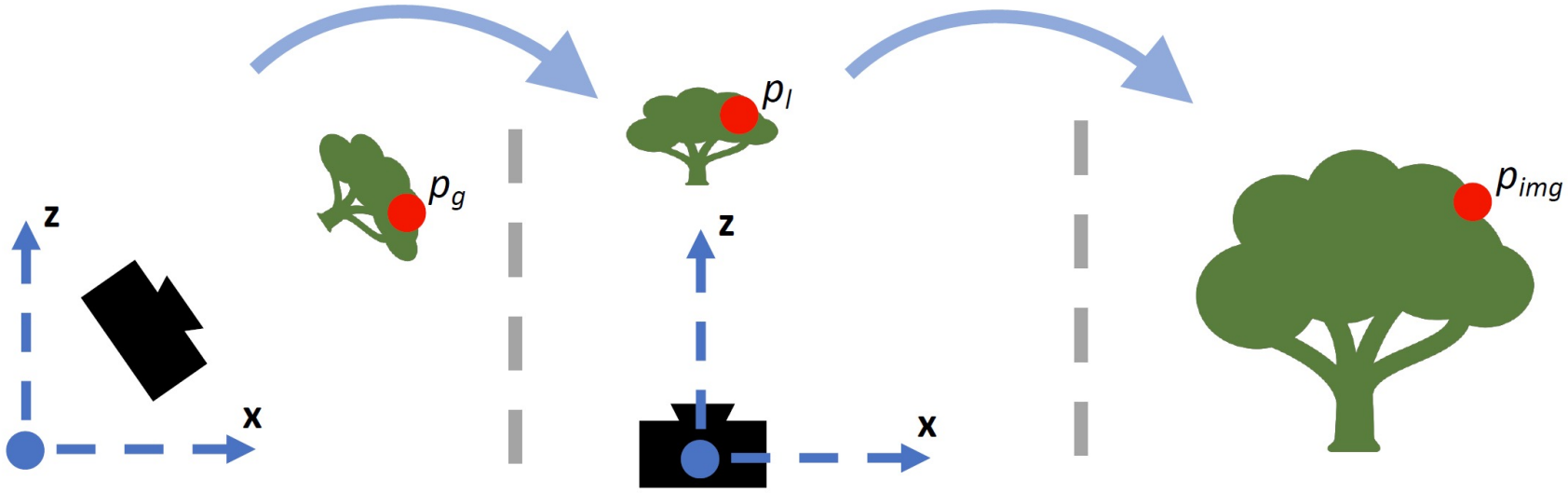
Coordinate frames + Transforms

Orientation + Location of the camera in the World

Extrinsics (R, T)

How the camera maps a point in 3D to image

Intrinsics (K)



World coordinates

Camera coordinates

Image coordinates

Figure credit: Peter Hedman

Camera: Specifics

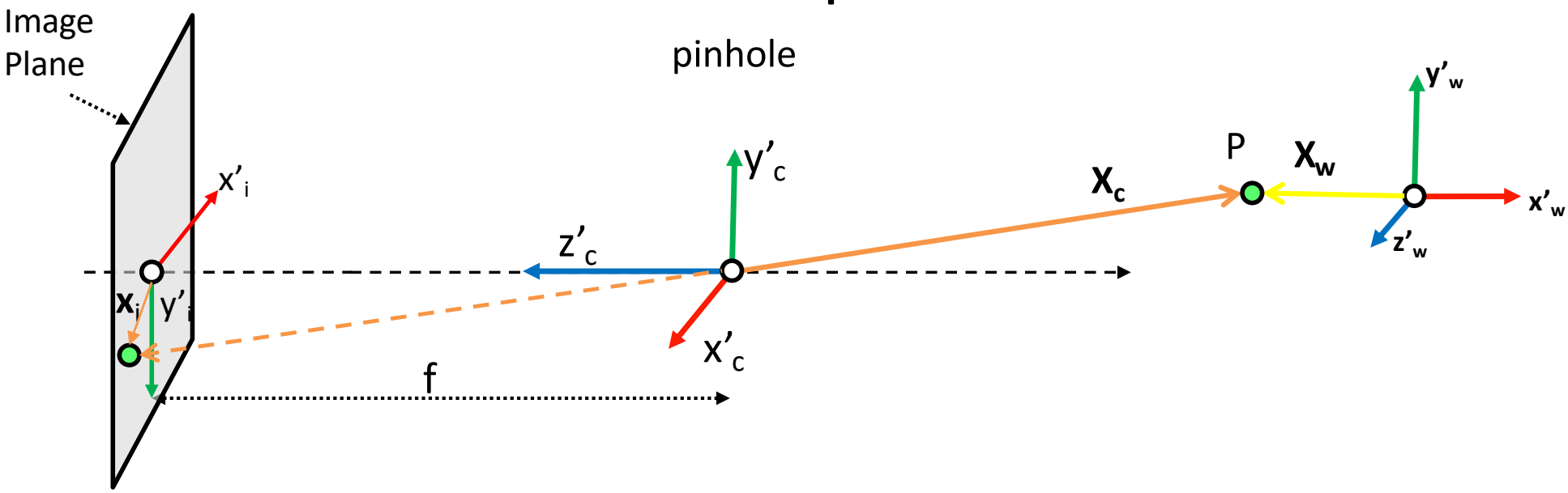


Image Coordinates

Camera Coordinates

World Coordinates

$$\mathbf{x}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad \mathbf{X}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \quad \mathbf{X}_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$$

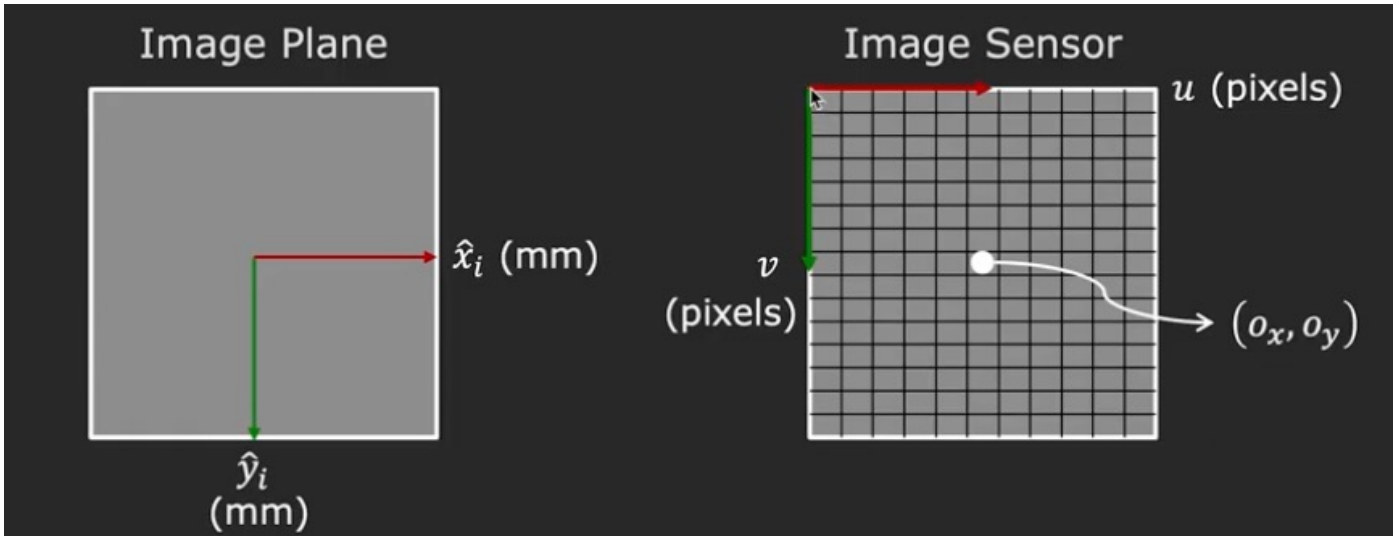
Perspective Projection

Coordinate Transformation

$$x_i = f\frac{x_c}{z_c}$$

Slide inspired by Shree Nayar

Image Plane to Image Sensor Mapping



1. Account for pixel density (pixel/mm) & aspect ratio by scalars: $[m_x, m_y]$

$$m_x x_i, m_y y_i$$

2. Usually the top left corner of the image is the origin. But in the image plane, where the optical axis pierces the plane is the origin, need to shift by: (o_x, o_y)

$$u_i = \alpha_x x_i + o_x = \alpha_x f \frac{x_c}{z_c} + o_x$$

where $[f_x, f_y] = [m_x f, m_y f]$

Pixel Coordinates:

$$u_i = \boxed{f_x} \frac{x_c}{z_c} + \boxed{o_x} \quad v_i = \boxed{f_y} \frac{y_c}{z_c} + \boxed{o_y}$$

With homogeneous coordinates

Perspective projection + Transformation to Pixel Coordinates:

$$u_i = f_x \frac{x_c}{z_c} + o_x \quad v_i = f_y \frac{y_c}{z_c} + o_y$$

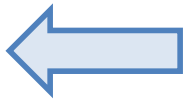
$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$

Intrinsic Matrix

Camera Transformation

Camera Coordinates

$$\mathbf{X}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$



Coordinate Transformation

World Coordinates

$$\mathbf{X}_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$$

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3\times 3} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

Extrinsic Matrix

Putting it all together

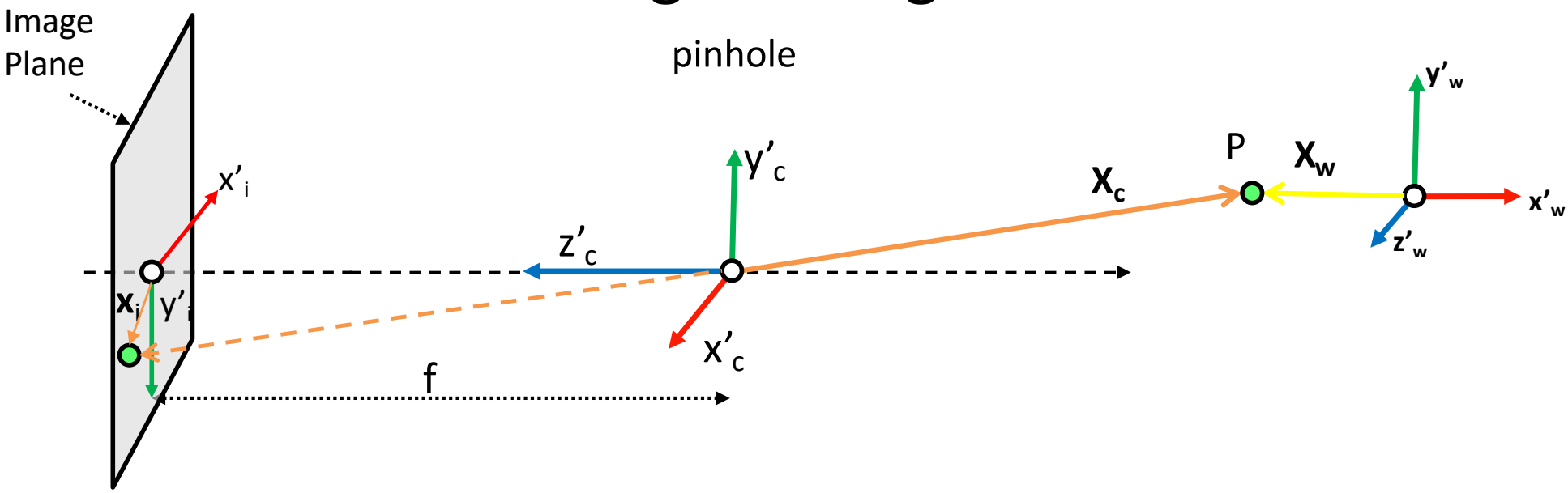


Image Coordinates

Camera Coordinates

World Coordinates

$$\mathbf{x}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad \mathbf{X}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \quad \mathbf{X}_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$$

Perspective Projection

Coordinate Transformation

$$\begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} R_{3\times 3} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}$$

Slide inspired by Shree Nayar

Projection Matrix

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R_{3\times 3} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

3 x 4 Projection matrix

Count the Degrees of Freedom:

Intrinsics: 4 + 1 (skew)

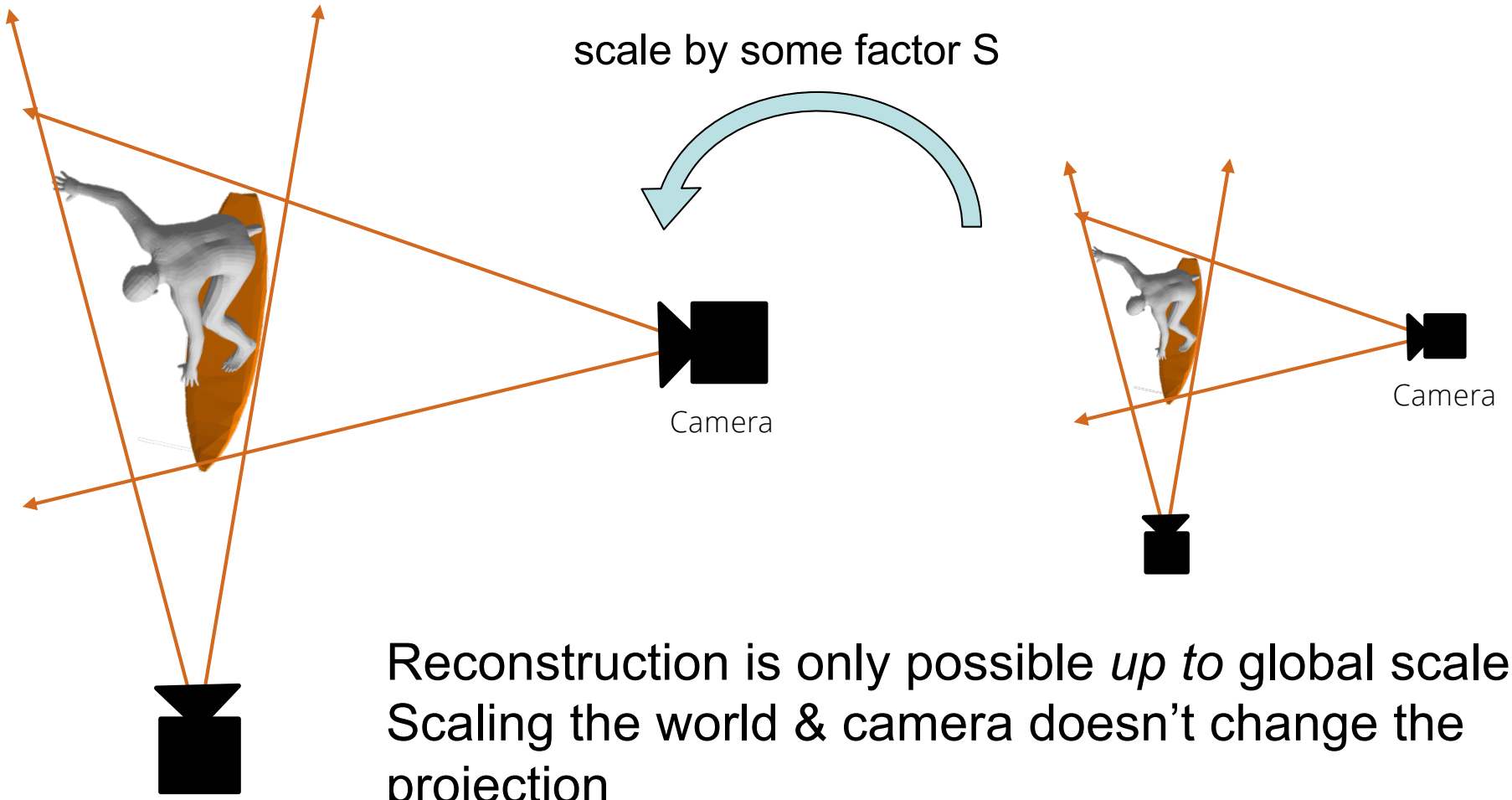
Extrinsic: 3 + 3 = 6

11 unknowns (up to scale)

For completeness, we need to add **skew** (this is 0 unless pixels are shaped like rhombi/parallelograms)

$$K = \begin{bmatrix} f_x & s & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Fundamental Scale Ambiguity



Reconstruction is only possible *up to* global scale
Scaling the world & camera doesn't change the projection
Unless you know something metric about the scene
e.g. surfboard is 2.1m

How to calibrate the camera?

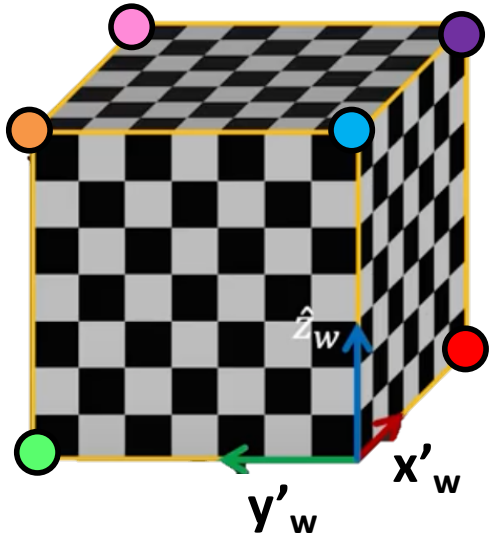
$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

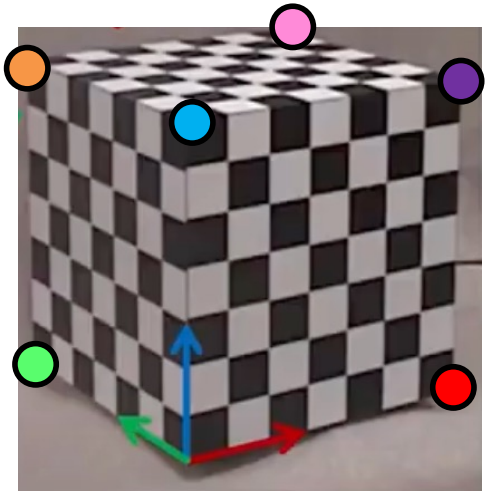
$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

If we know the points in 3D we can estimate the camera!!

Step 1: With a known 3D object

1. Take a picture of an object with known 3D geometry

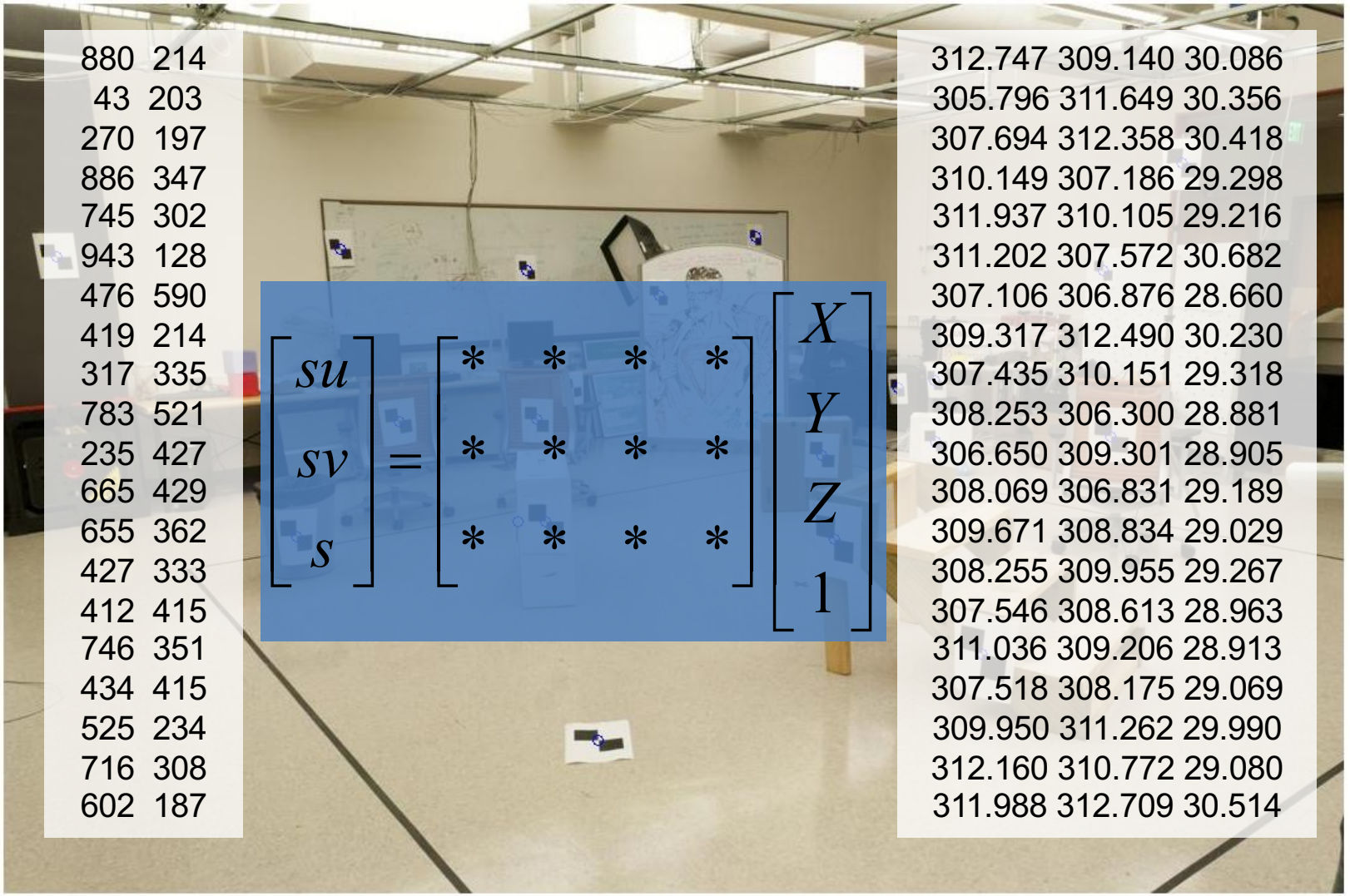




2. Identify correspondences

Slide from Shree Nayar

How do we calibrate a camera?



880	214
43	203
270	197
886	347
745	302
943	128
476	590
419	214
317	335
783	521
235	427
665	429
655	362
427	333
412	415
746	351
434	415
525	234
716	308
602	187

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

312.747	309.140	30.086
305.796	311.649	30.356
307.694	312.358	30.418
310.149	307.186	29.298
311.937	310.105	29.216
311.202	307.572	30.682
307.106	306.876	28.660
309.317	312.490	30.230
307.435	310.151	29.318
308.253	306.300	28.881
306.650	309.301	28.905
308.069	306.831	29.189
309.671	308.834	29.029
308.255	309.955	29.267
307.546	308.613	28.963
311.036	309.206	28.913
307.518	308.175	29.069
309.950	311.262	29.990
312.160	310.772	29.080
311.988	312.709	30.514

Method 1 – homogeneous linear system

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- Solve for m's entries using linear least squares

Ax=0 form

$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 & -u_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 & -v_1 \\ & & & & & & \vdots & & & & & \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -u_nX_n & -u_nY_n & -u_nZ_n & -u_n \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -v_nX_n & -v_nY_n & -v_nZ_n & -v_n \end{bmatrix} \begin{bmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

Can we factorize M back to K [R | T]?

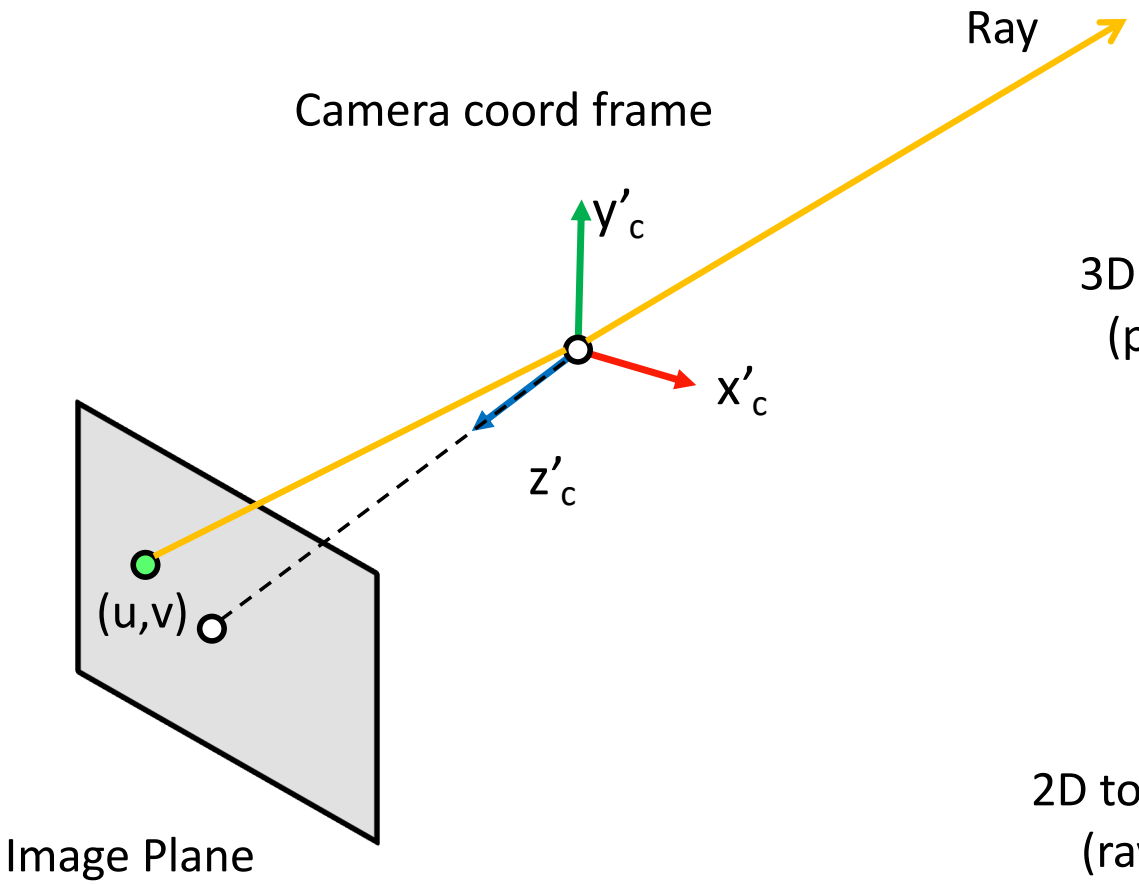
- Yes
- Why? because K and R have a very special form:

$$\begin{bmatrix} f_x & s & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

- Practically, use camera calibration packages (there is a good one in OpenCV)

Now that our cameras are calibrated, can we find the 3D scene point of a pixel?

You know we can't, but we know it'll be… on the ray!



3D to 2D:
(point)

$$u = f_x \frac{x_c}{z_c} + o_x$$

$$v = f_y \frac{y_c}{z_c} + o_y$$

2D to 3D:
(ray)
Back projection

$$x = \frac{z}{f_x}(u - o_x)$$

$$y = \frac{z}{f_y}(v - o_y)$$

$$z > 0$$

Slide credit: Shree Nayar

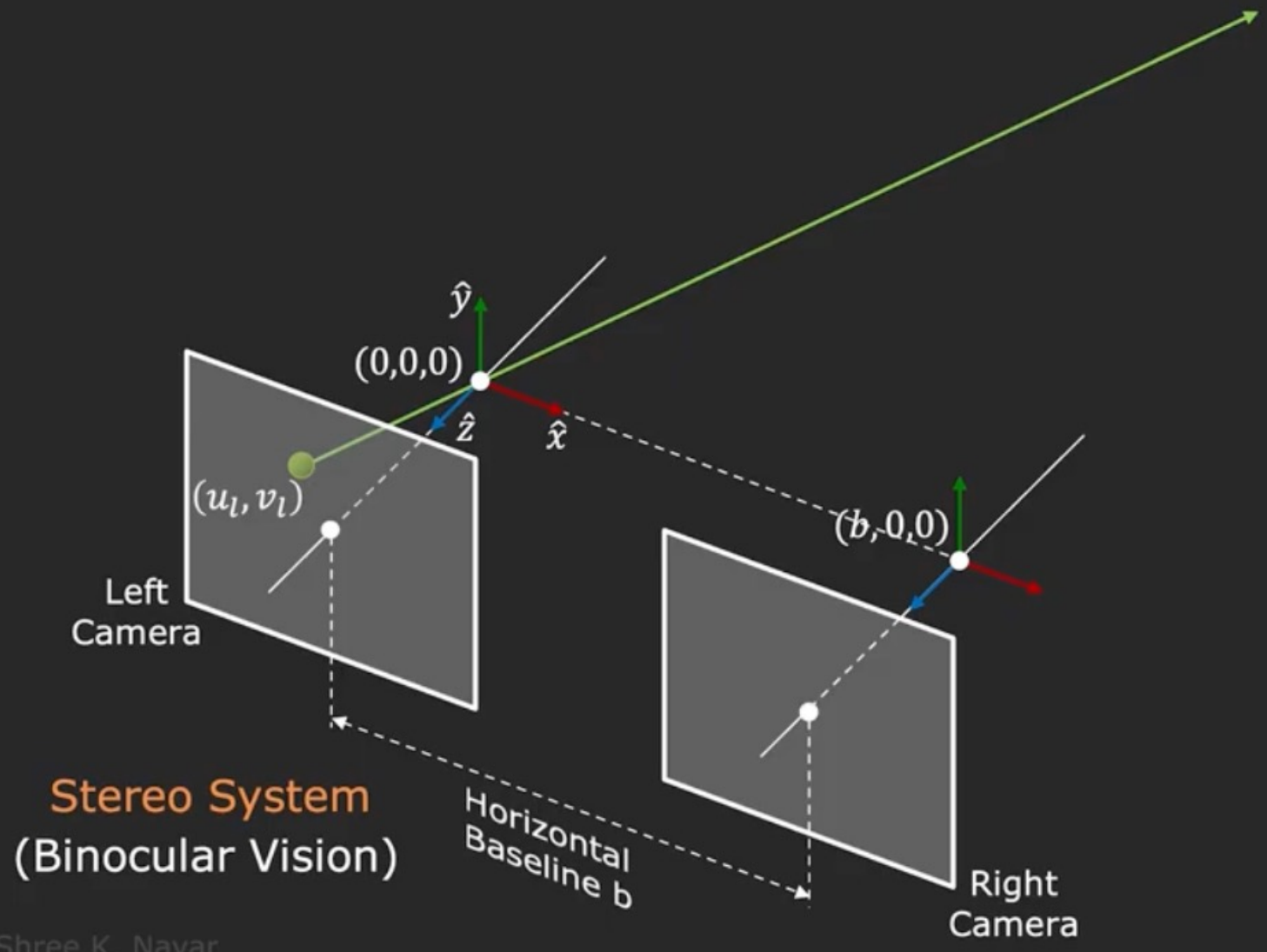
Simple Stereo Setup

- Assume **parallel** optical axes
- Two cameras are calibrated
- Find relative depth

Key Idea: difference in corresponding points to understand shape

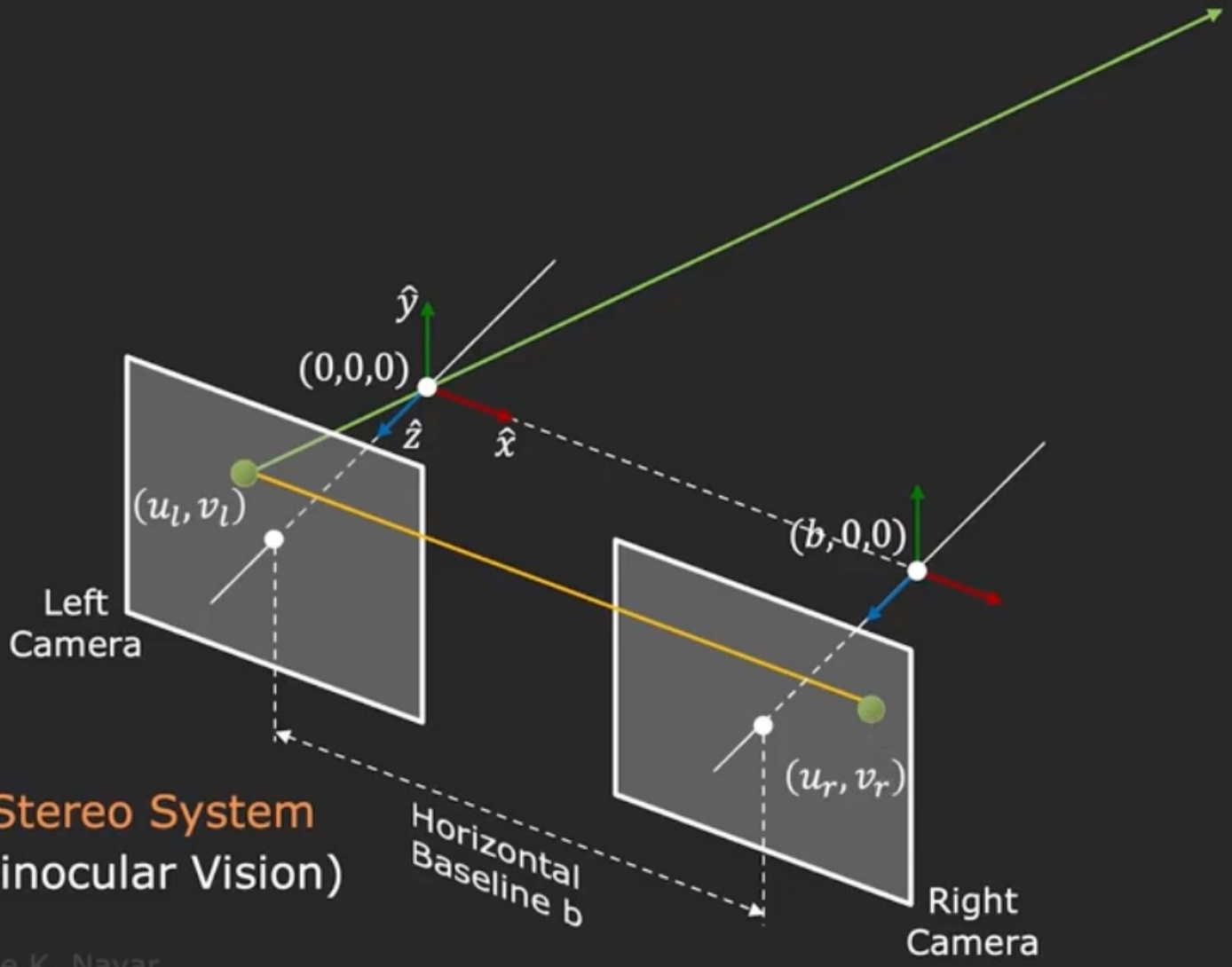
Slide credit: Noah Snavely

Triangulation using two cameras



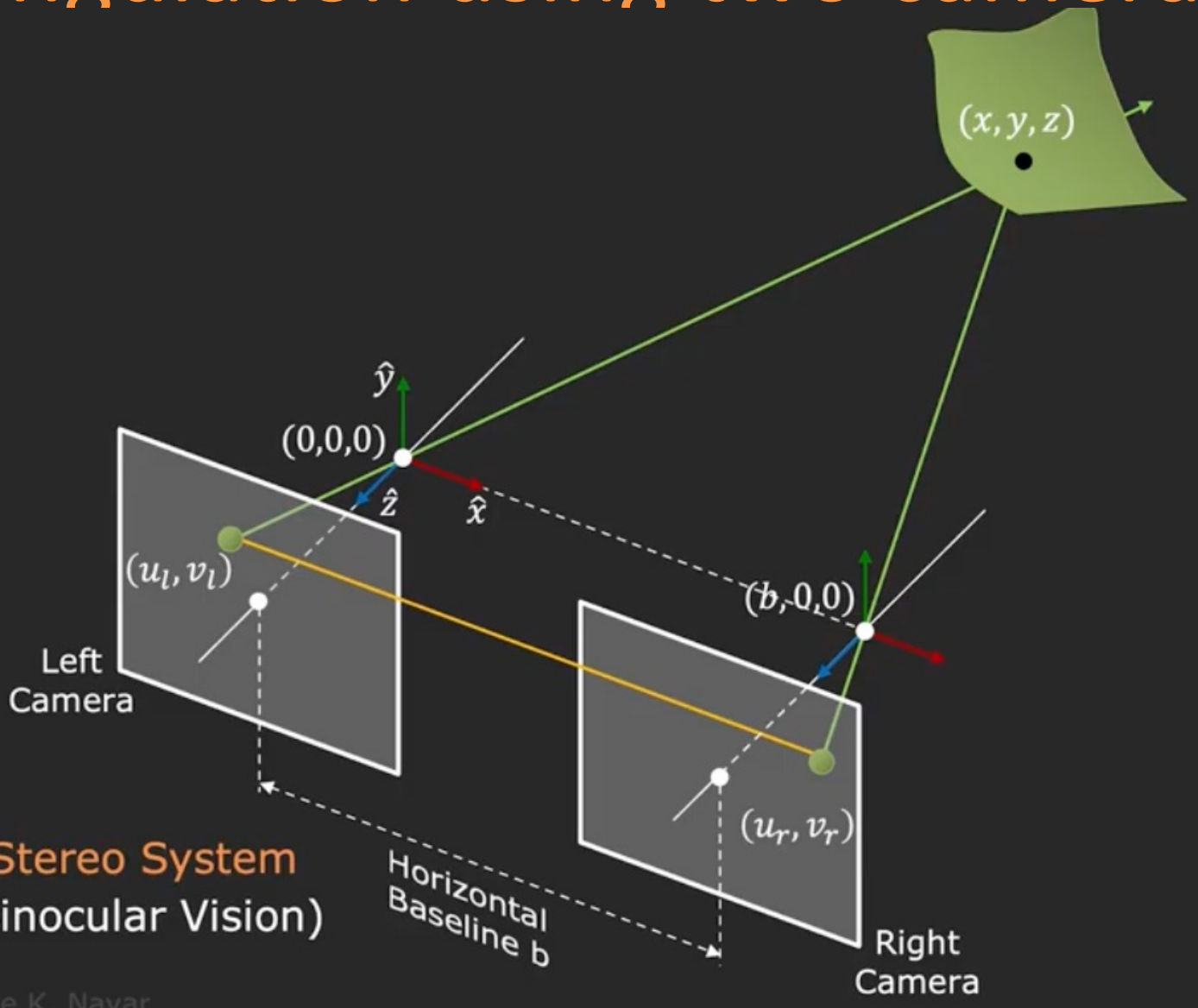
© 2020 Shree K. Nayar

Triangulation using two cameras



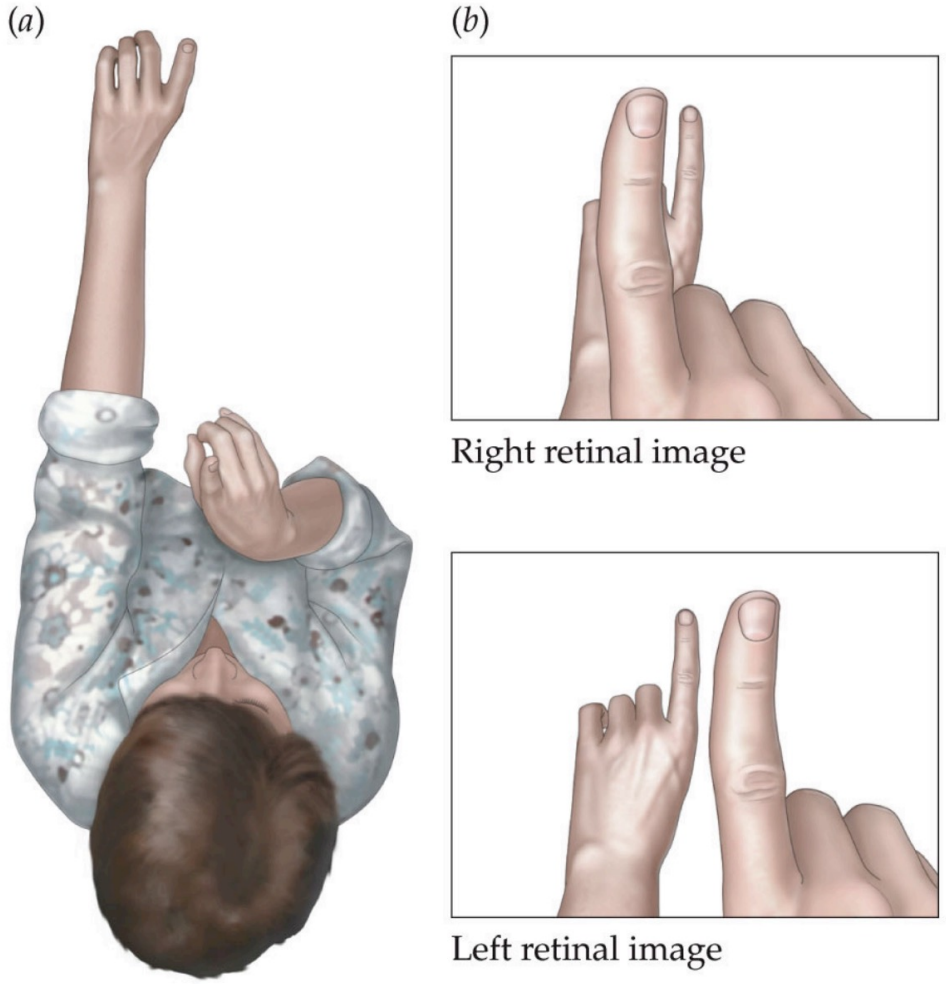
© 2020 Shree K. Nayar

Triangulation using two cameras

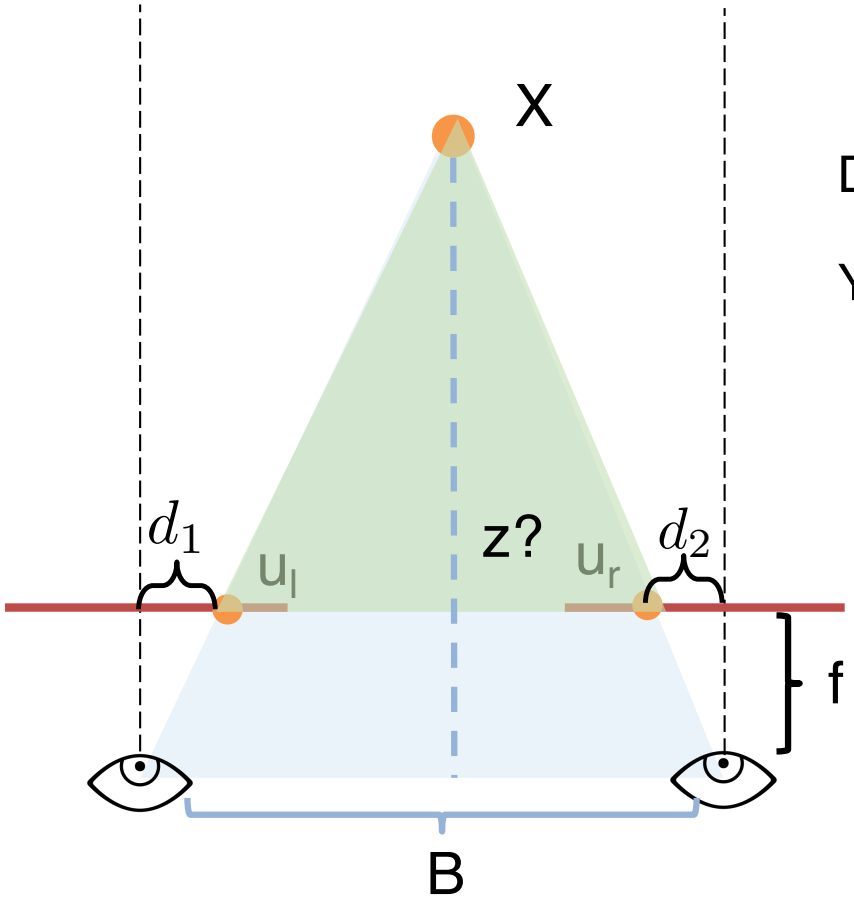


020 Shree K. Nayar

We are equipped with binocular vision. Let's try!



Solving for Depth in Simple Stereo



Do we have enough to know what is Z?

Yes, similar triangles!

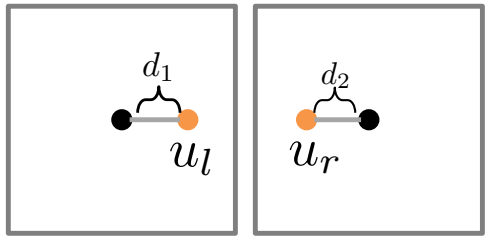
$$\frac{B-(u_l-u_r)}{z-f}=\frac{B}{z}$$

$$z=\frac{fB}{u_l-u_r}$$

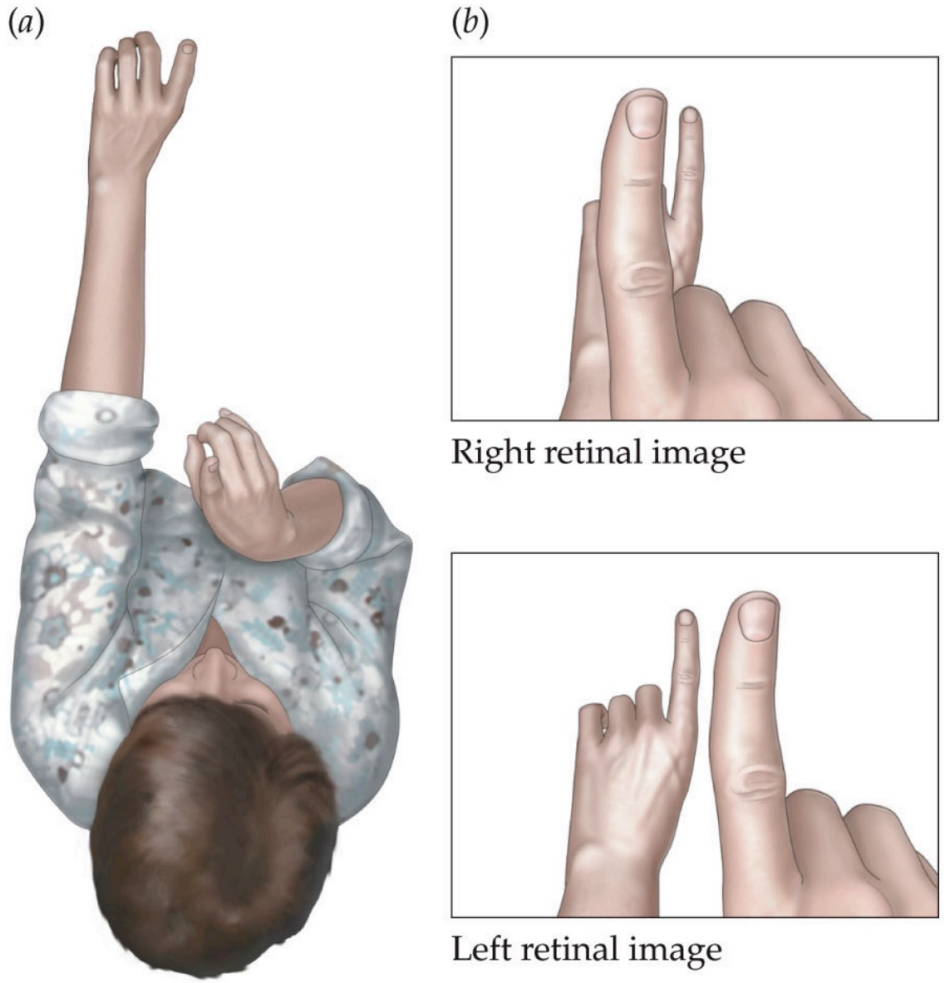
disparity (how much corrsp. pixels move)

Base of △ : $B-(d1+d_2)$

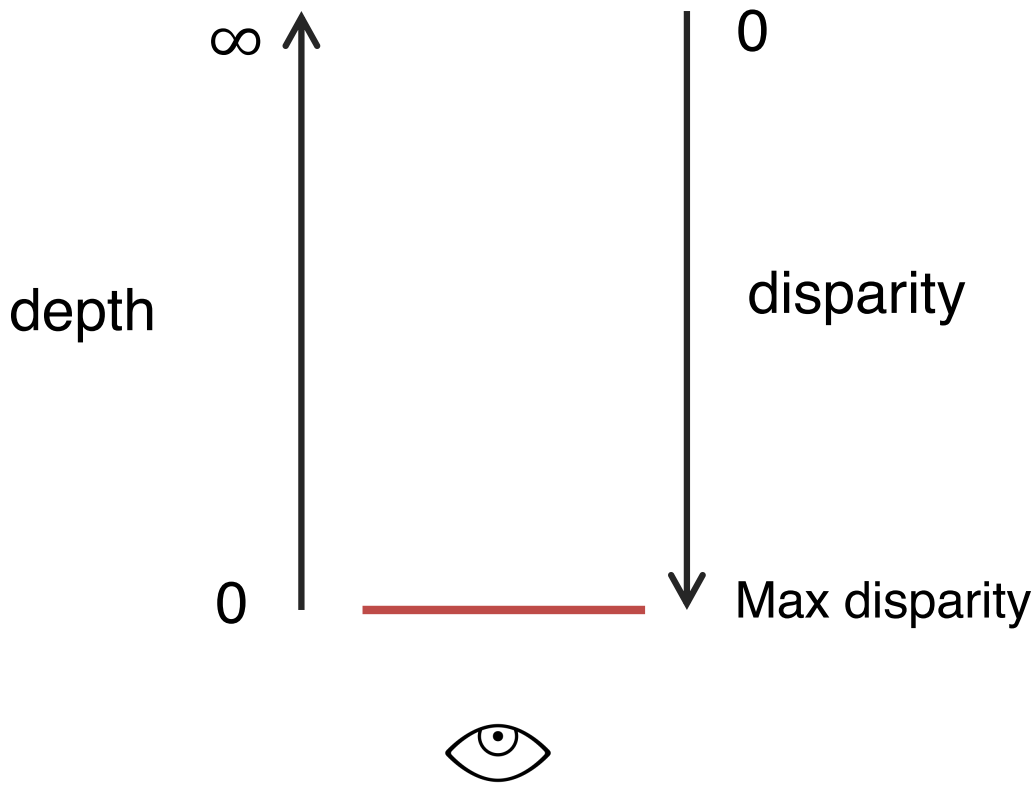
in image coordinates: $=B-(u_l-u_r)$



Try with your hands!



Depth is inversely proportional to disparity



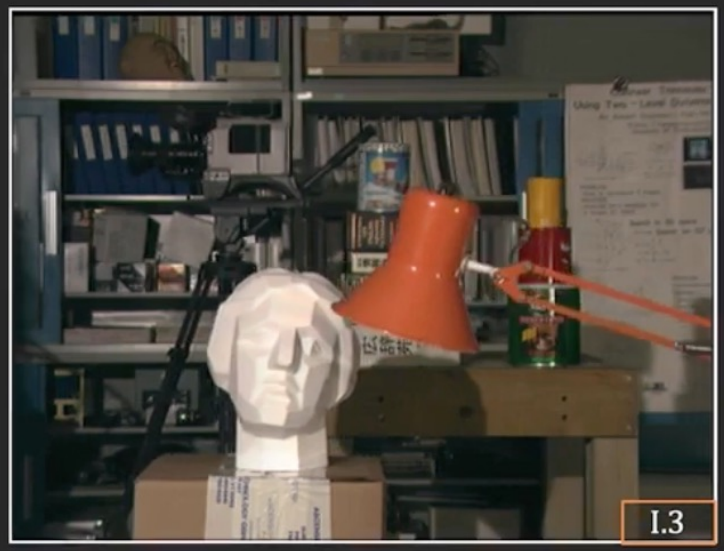
$$z = \frac{fB}{u_l - u_r}$$

disparity

$$z \propto \frac{1}{u_l - u_r} = \frac{1}{d}$$

Stereo Matching: Finding Disparities

Goal: Find the disparity between left and right stereo pairs.

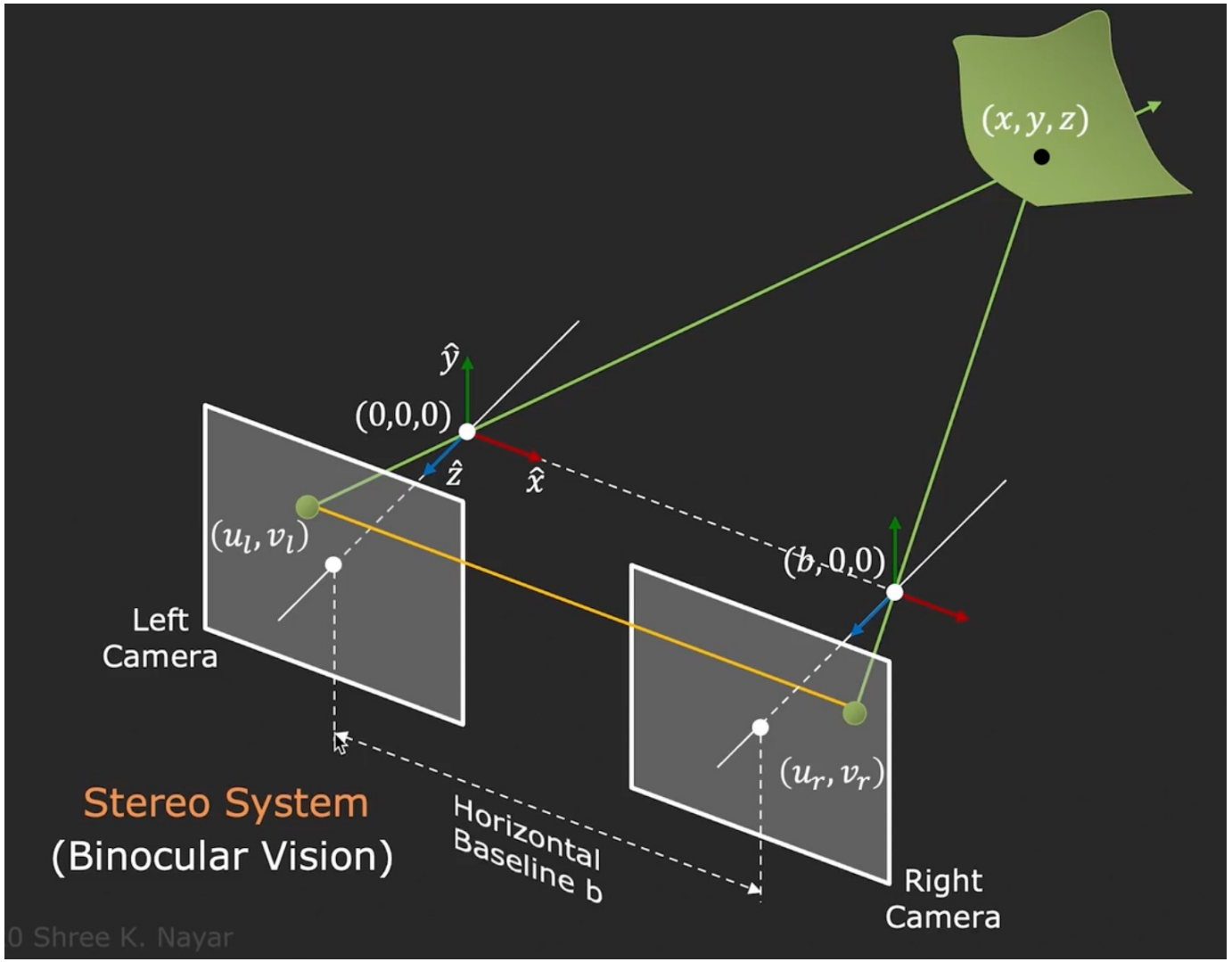


Left/Right Camera Images

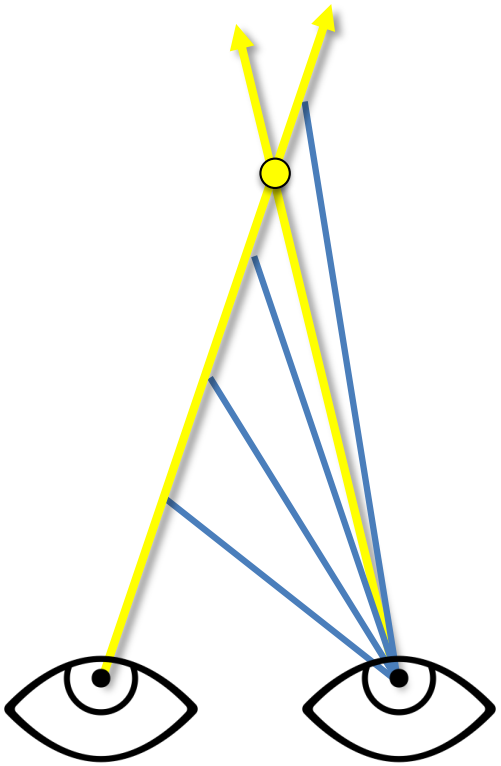
Disparity Map (Ground Truth)

Where is the corresponding point going to be?

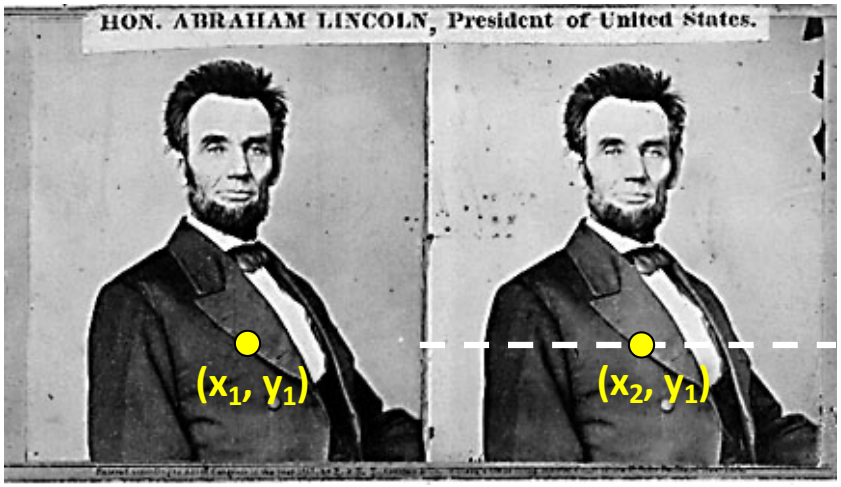
Hint



Epipolar Line



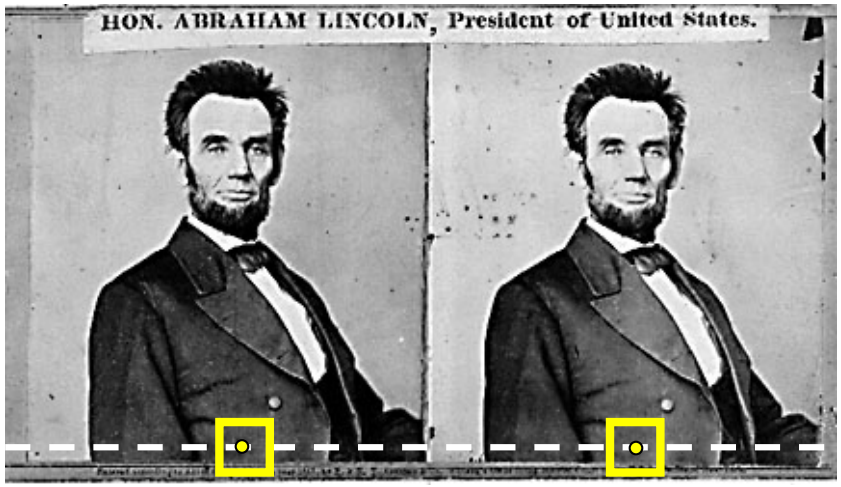
epipolar lines



Two images captured by a purely horizontal translating camera
(*rectified* stereo pair)

$x_1 - x_2$ = the *disparity* of pixel (x_1, y_1)

Your basic stereo algorithm



For every epipolar line:

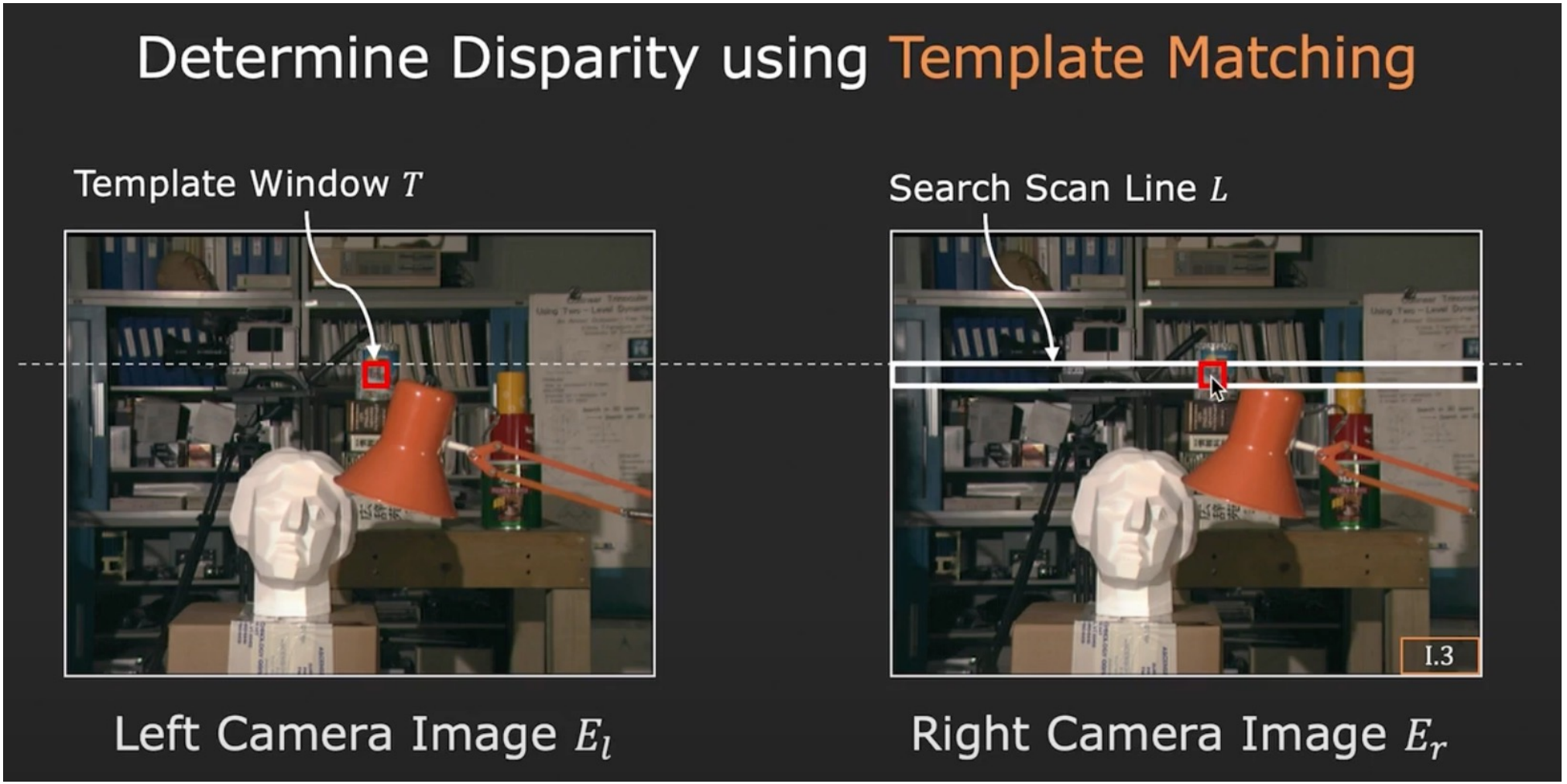
For each pixel in the left image

- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

Improvement: match ***windows,*** *+ clearly lots of matching strategies*

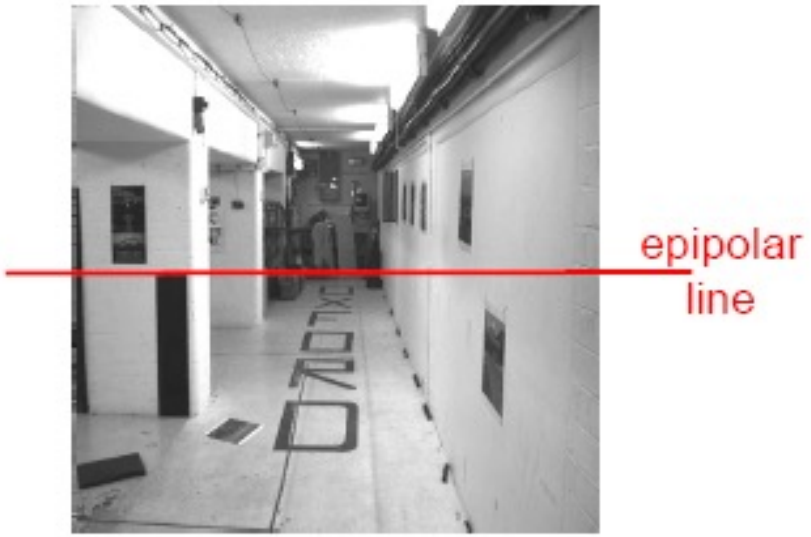
Slide credit: Noah Snavely

Your basic stereo algorithm



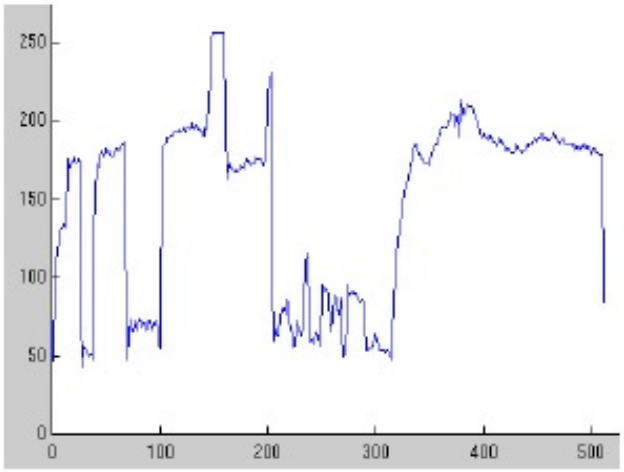
Correspondence problem

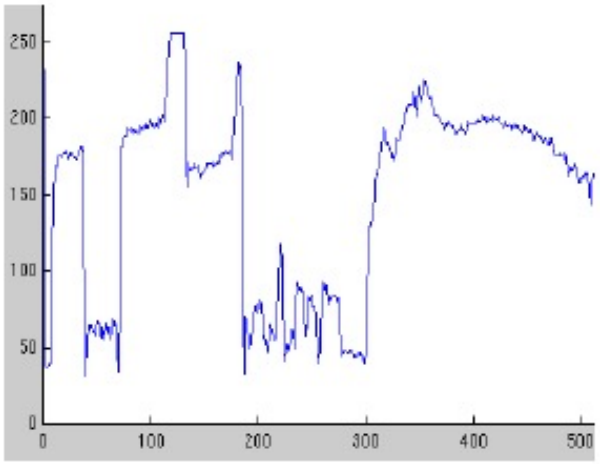
Parallel camera example – epipolar lines are corresponding rasters



Source: Andrew Zisserman

Intensity profiles

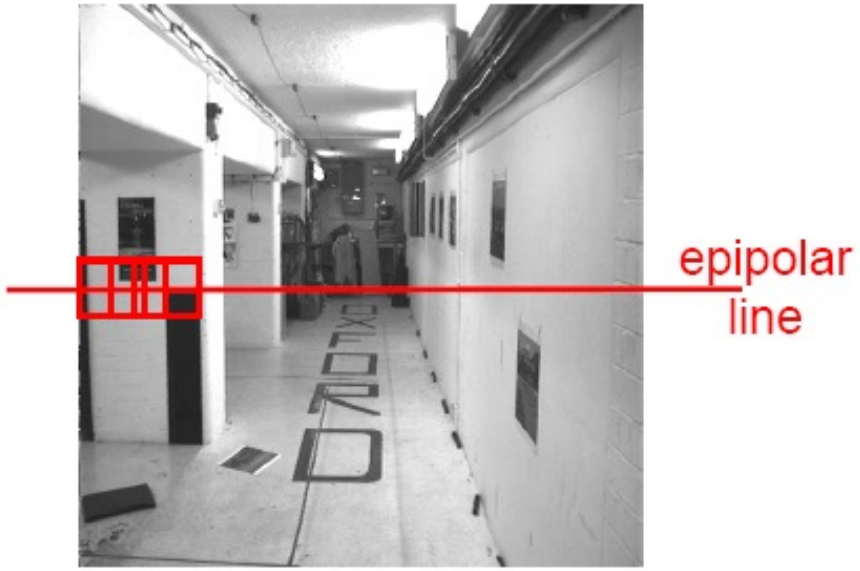




- Clear correspondence between intensities, but also noise and ambiguity

Source: Andrew Zisserman

Correspondence problem



Neighborhood of corresponding points are similar in intensity patterns.

Source: Andrew Zisserman

Normalized cross correlation

subtract mean: $A \leftarrow A - < A >, B \leftarrow B - < B >$

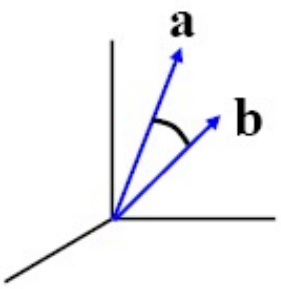
$$\text{NCC} = \frac{\sum_i \sum_j A(i,j)B(i,j)}{\sqrt{\sum_i \sum_j A(i,j)^2}\sqrt{\sum_i \sum_j B(i,j)^2}}$$

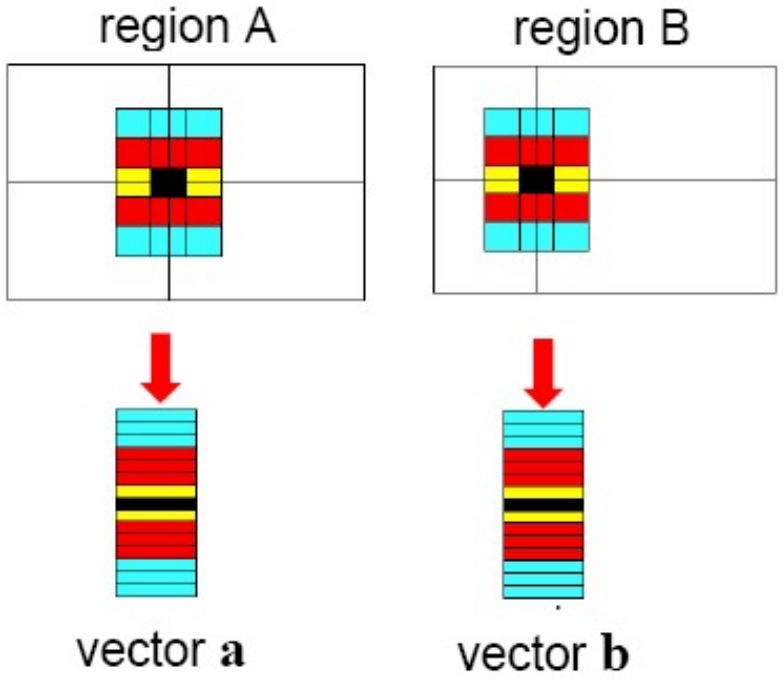
Write regions as vectors

$$\mathbf{A} \rightarrow \mathbf{a}, \quad \mathbf{B} \rightarrow \mathbf{b}$$

$$\text{NCC} = \frac{\mathbf{a}.\mathbf{b}}{|\mathbf{a}||\mathbf{b}|}$$

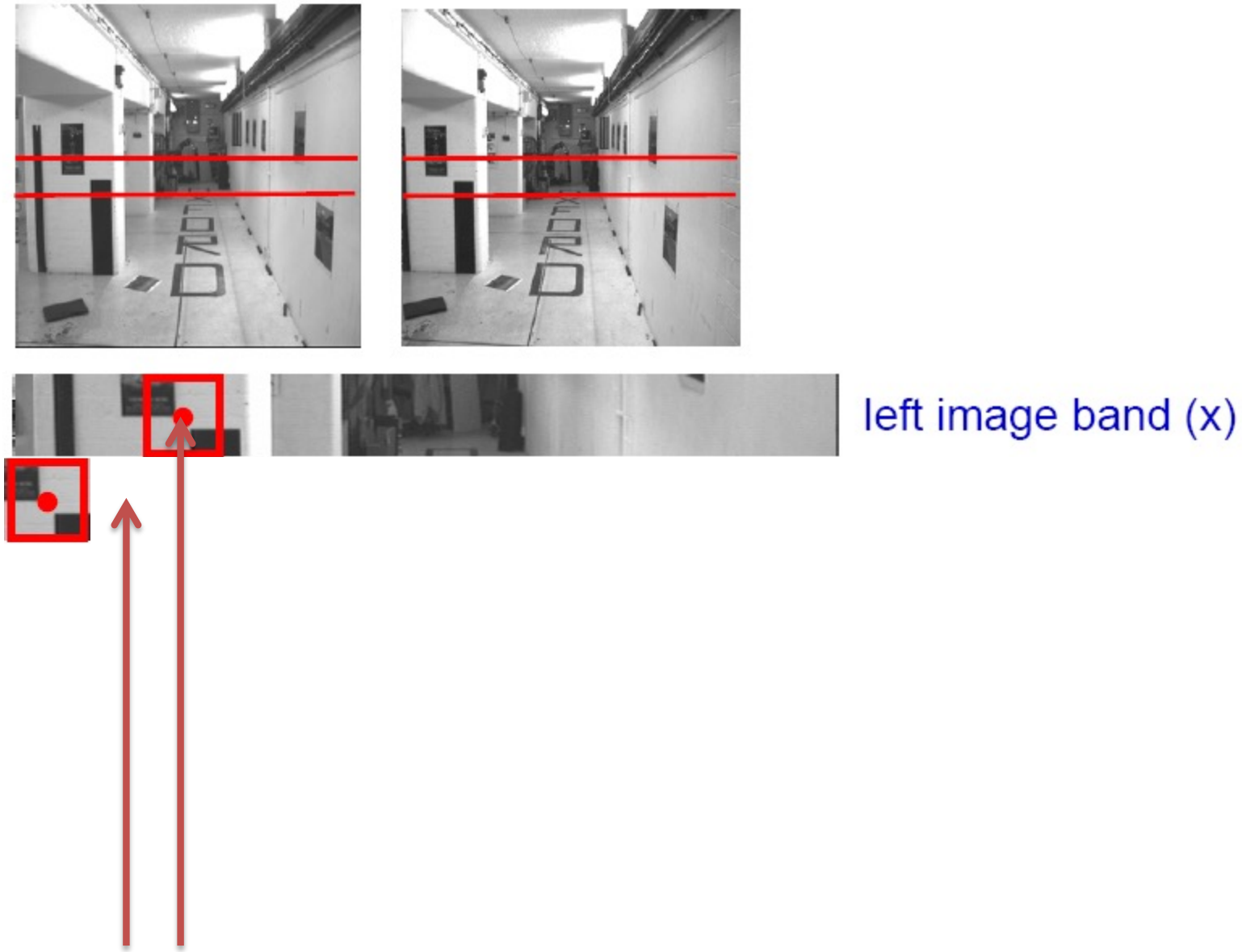
$$-1 \leq \text{NCC} \leq 1$$





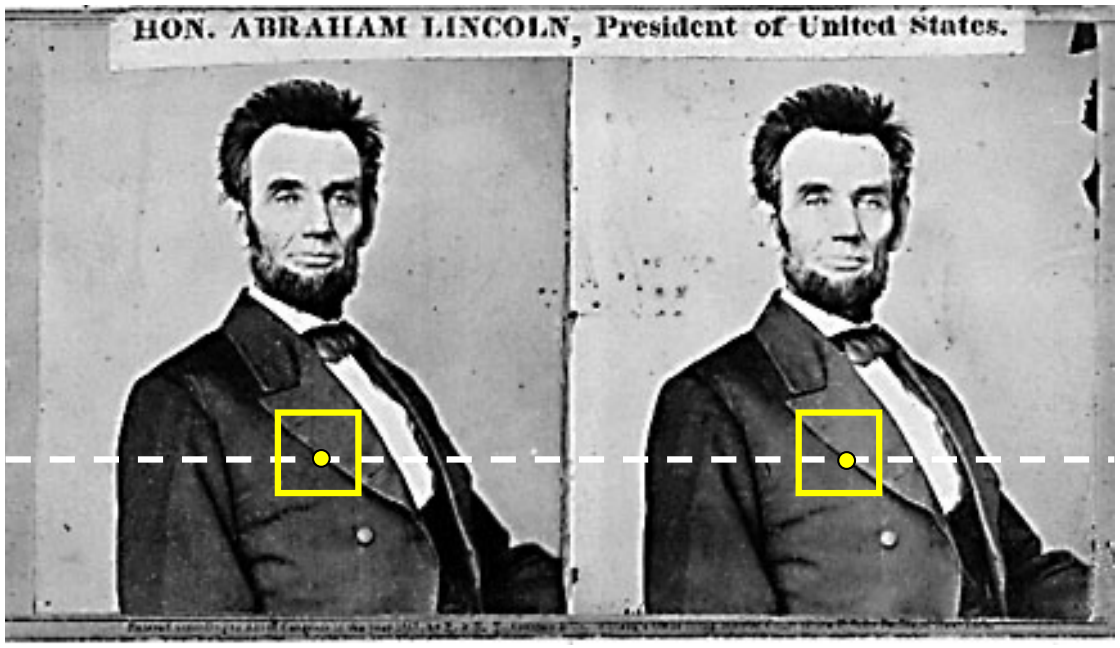
Source: Andrew Zisserman

Correlation-based window matching



Source: Andrew Zisserman

Dense correspondence search



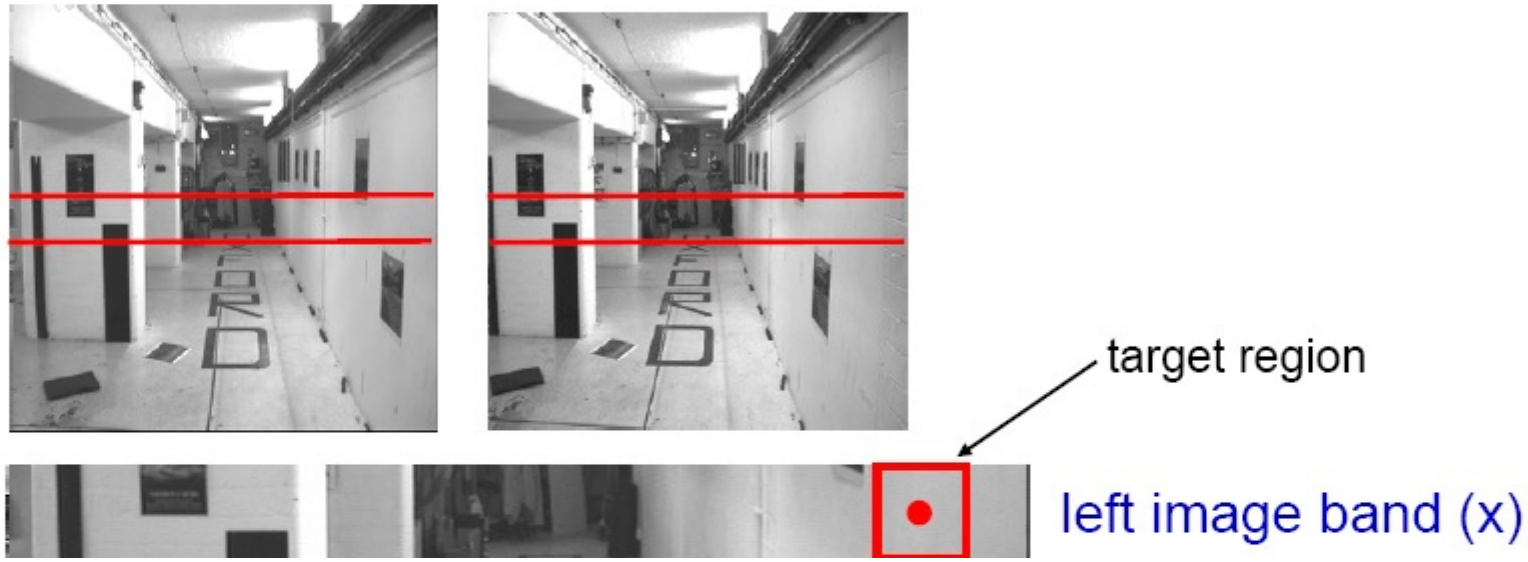
For each epipolar line

For each pixel / window in the left image

- compare with every pixel / window on same epipolar line in right image
- pick position with minimum match cost (e.g., SSD, correlation)

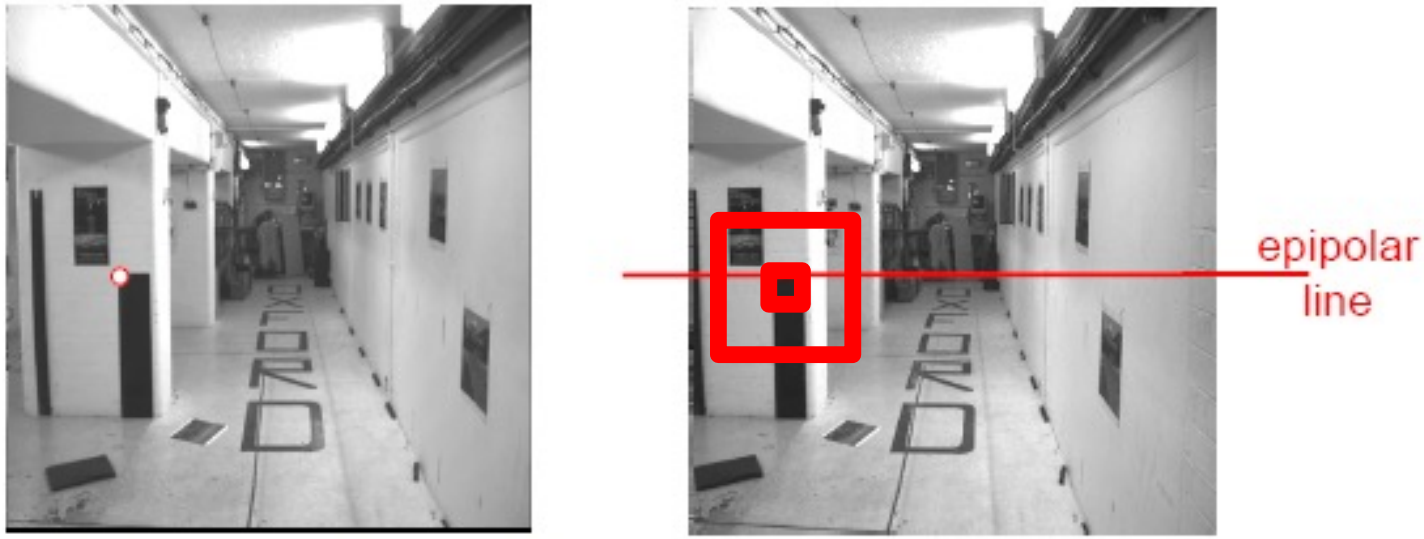
Adapted from Li Zhang

Textureless regions



Source: Andrew Zisserman

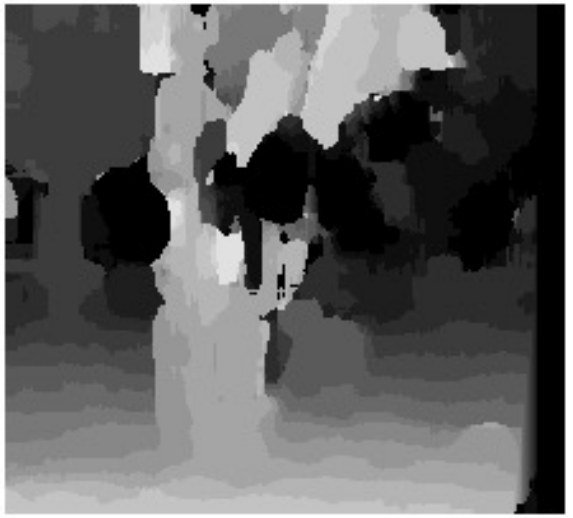
Effect of window size



Source: Andrew Zisserman

Effect of window size

W = 3



W = 20

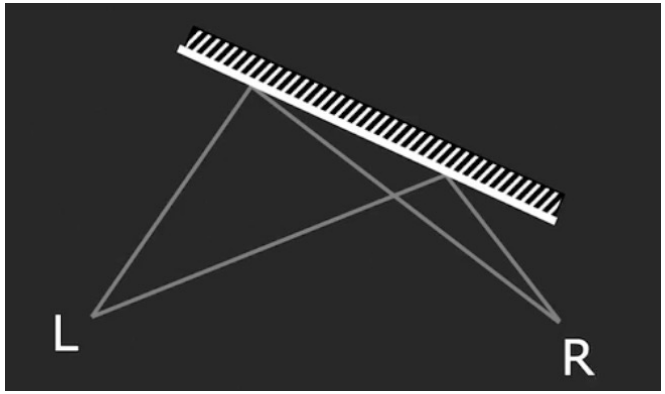
Want window large enough to have sufficient intensity variation, yet small enough to contain only pixels with about the same disparity.

Figures from Li Zhang

Issues with Stereo

- Surface must have non-repetitive texture

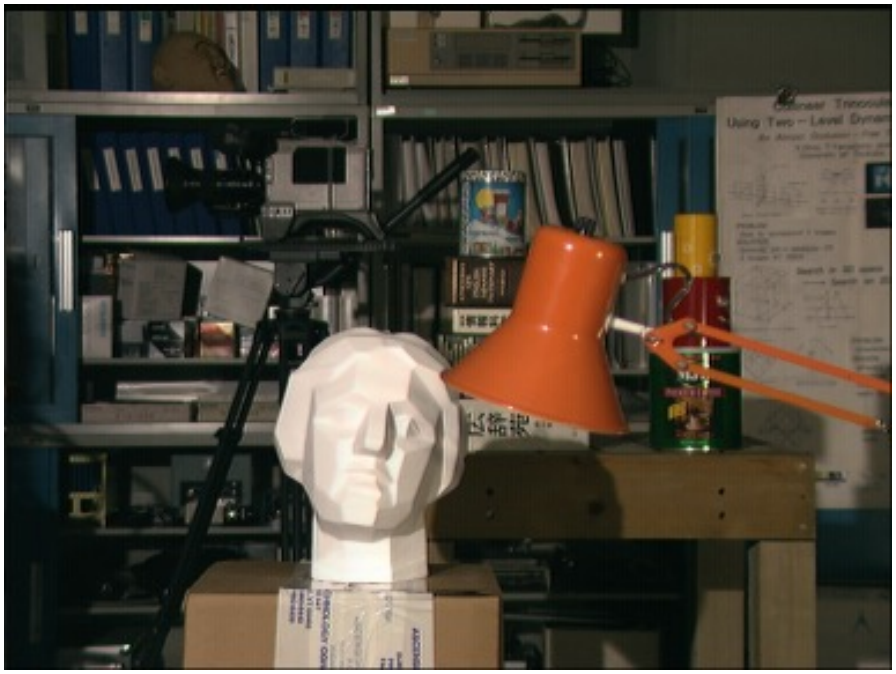
- Foreshortening effect makes matching a challenge



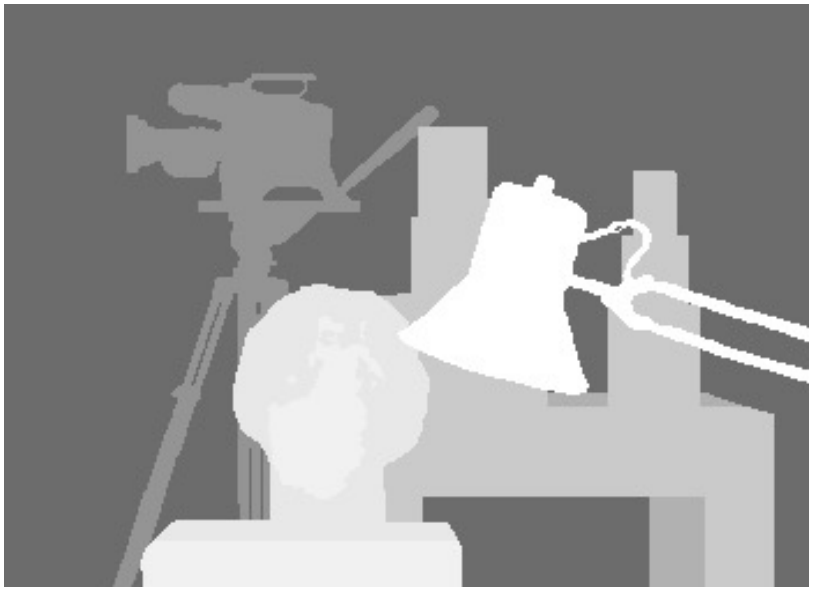
Slide Credit: Shree Nayar

Stereo Results

– Data from University of Tsukuba

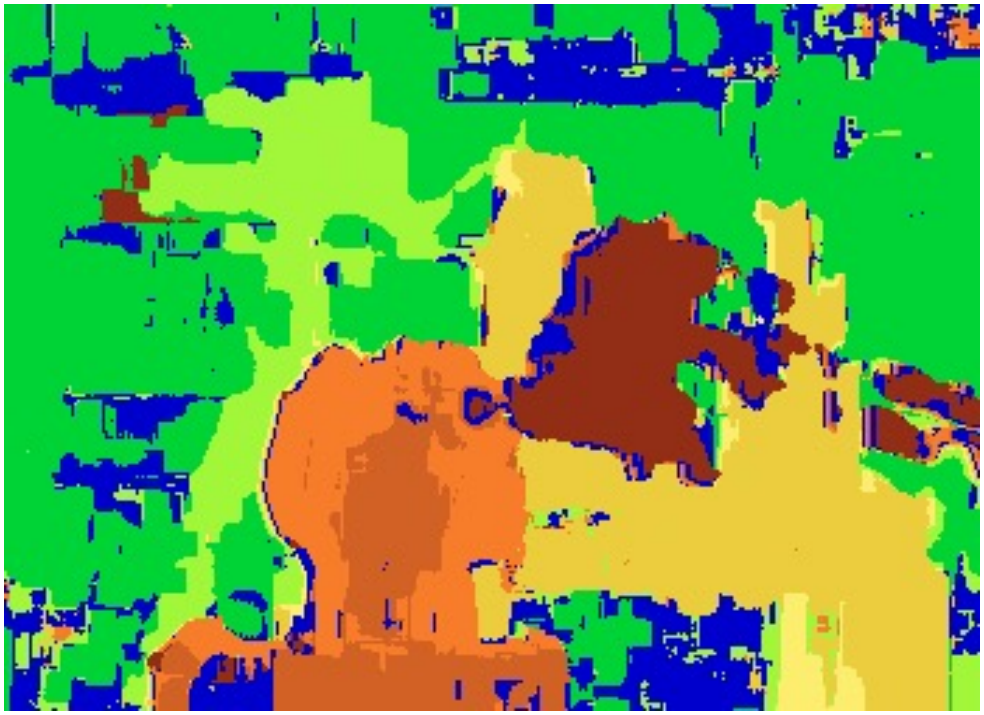


Scene



Ground truth

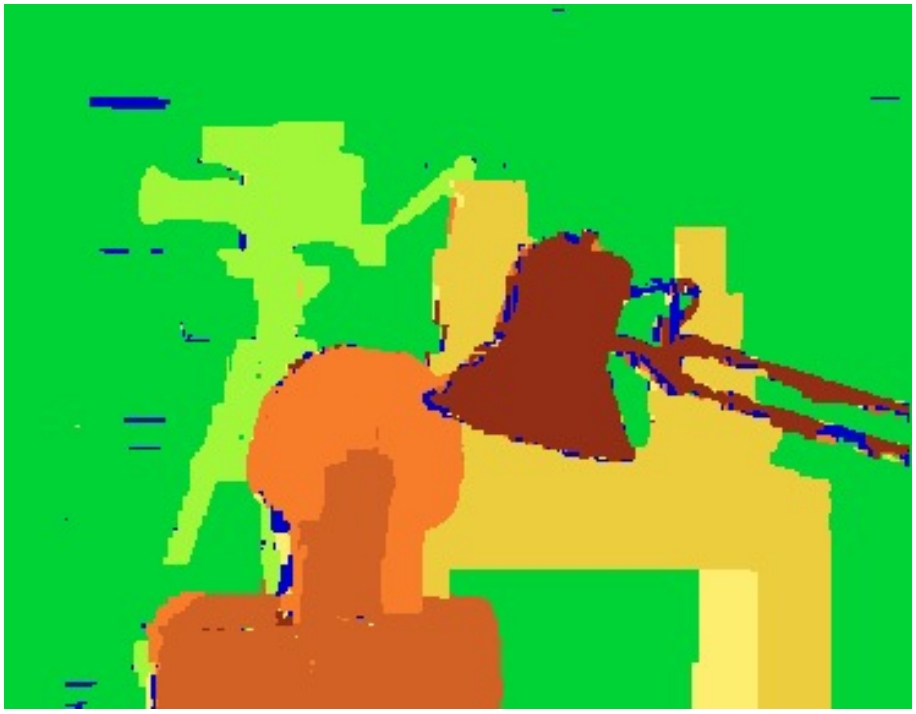
Results with Window Search



Window-based matching
(best window size)

Ground truth

Better methods exist...



Energy Minimization

Ground truth

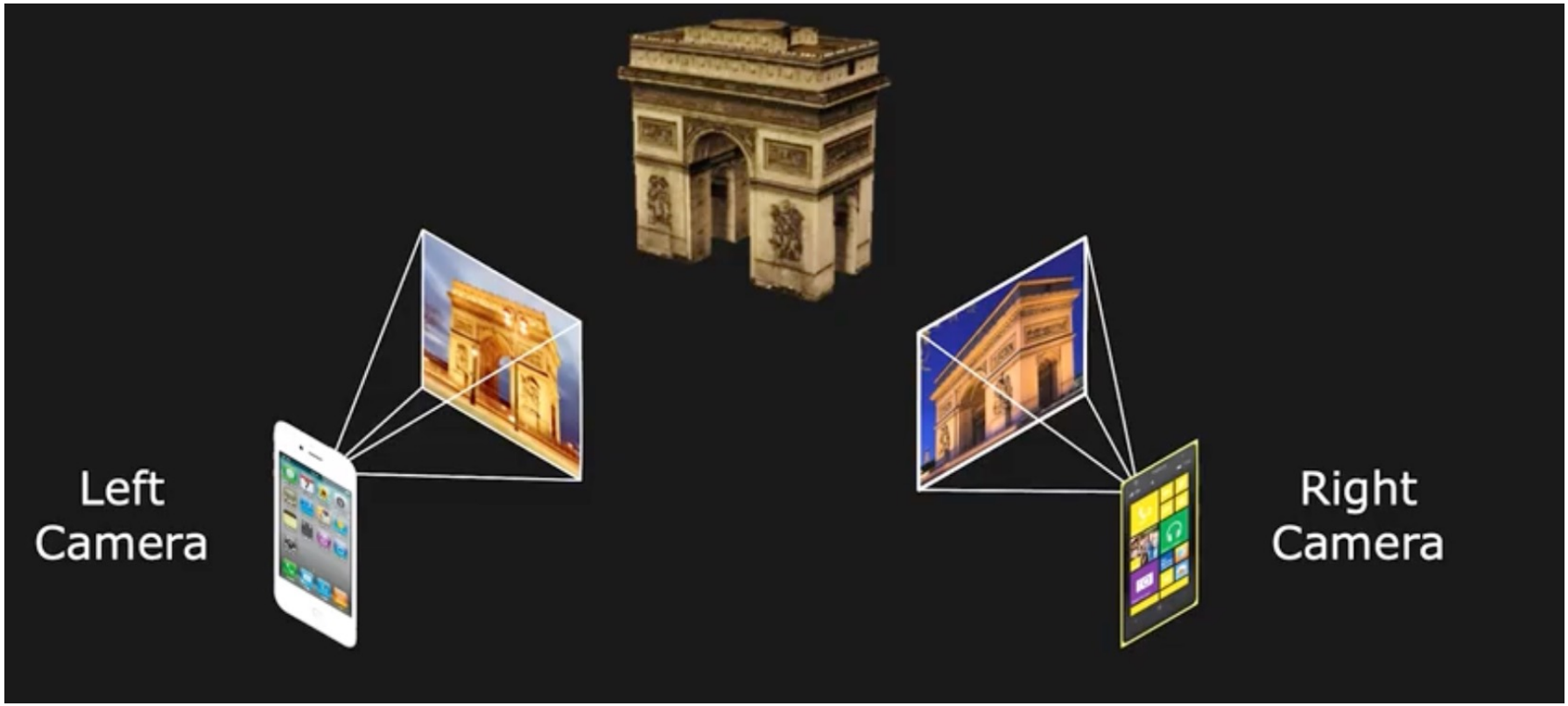
Boykov et al., Fast Approximate Energy Minimization via Graph Cuts, International Conference on Computer Vision, September 1999.

Summary

- With a simple stereo system, how much pixels move give information about the depth
- Correspondences let us measure the pixel disparity

Next: Uncalibrated Stereo

- From two arbitrary views

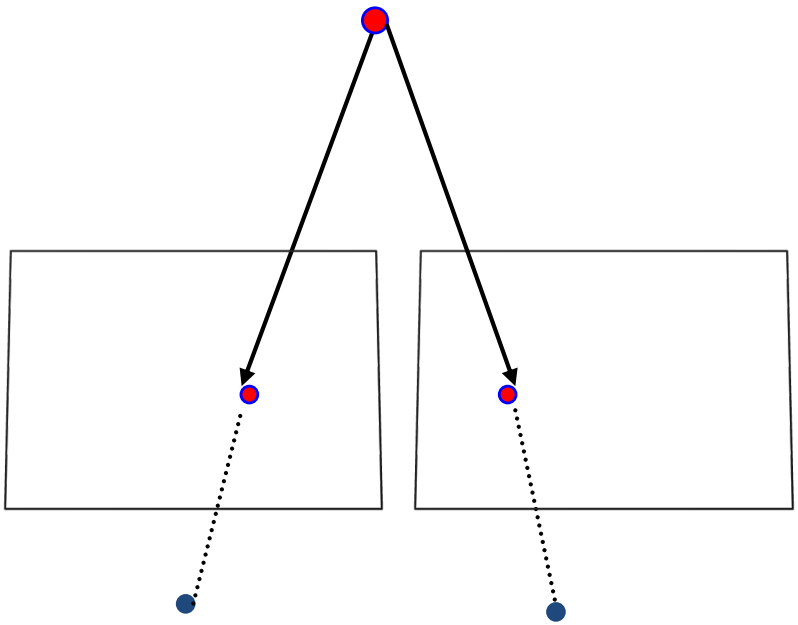


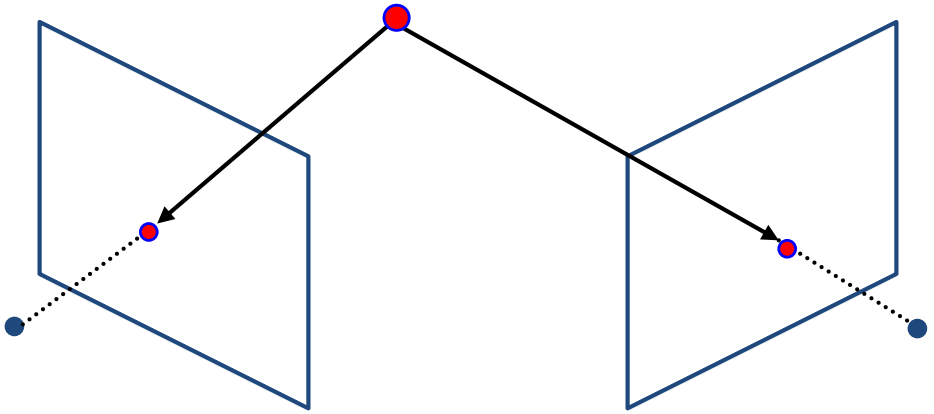
- Assume intrinsics are known (fx, fy, ox, oy)

Slide Credit: Shree Nayar

General case, with calibrated cameras

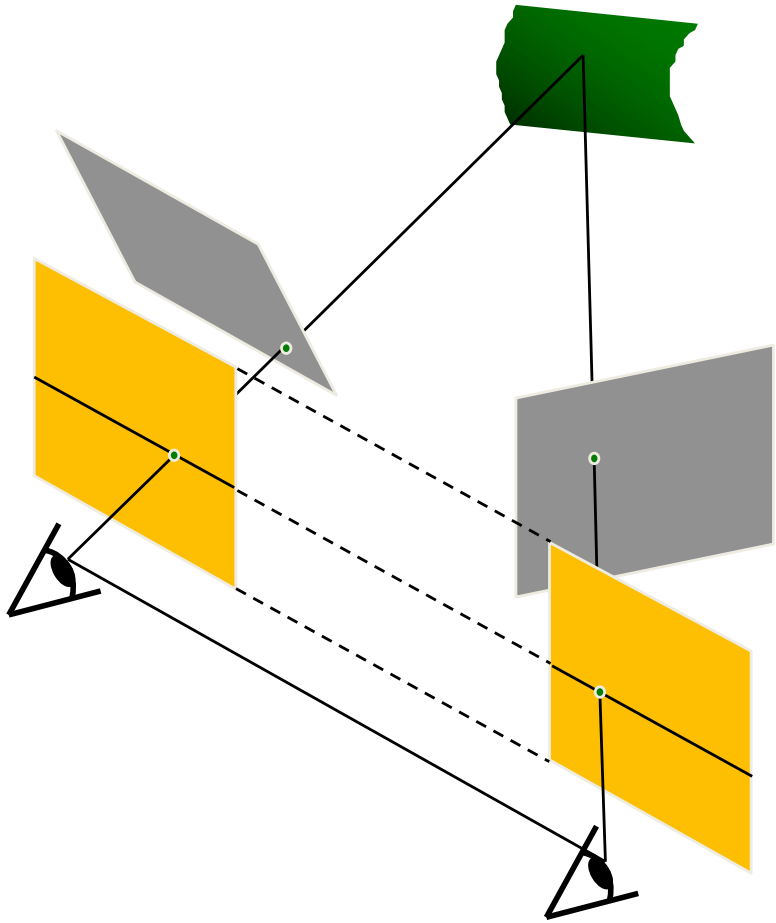
- The two cameras need not have parallel optical axes.





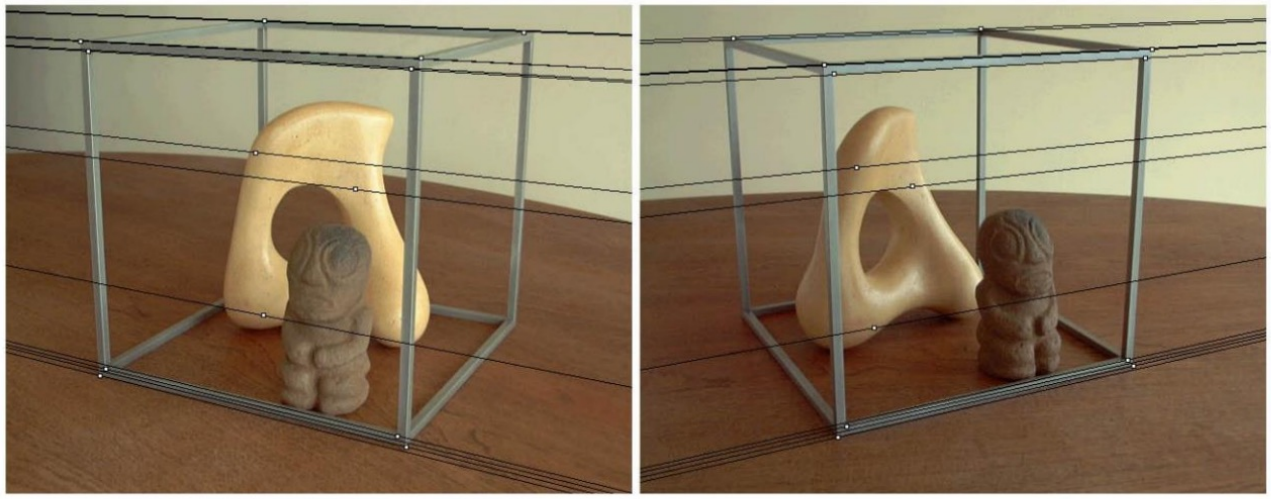
Option 1: Rectify via homography

- reproject image planes onto a common plane
  - plane parallel to the line between optical centers
- pixel motion is horizontal after this transformation
- two homographies, one for each input image reprojection
  - C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision. CVPR 1999.

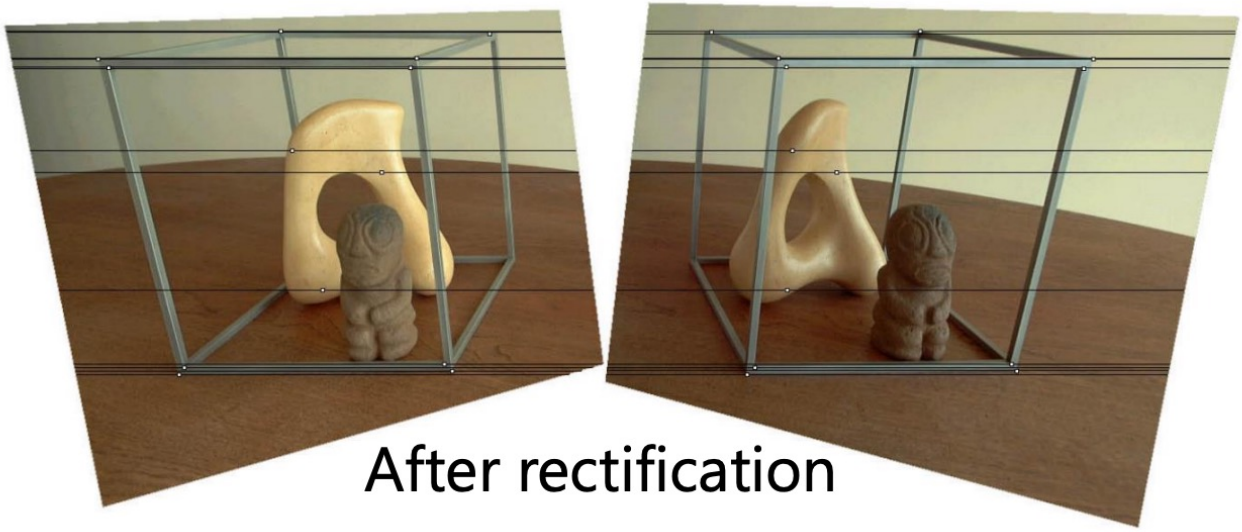


Slide courtesy of Noah Snavely

Option 1: Rectify via homography



Original stereo pair



After rectification

Slide courtesy of Noah Snavely

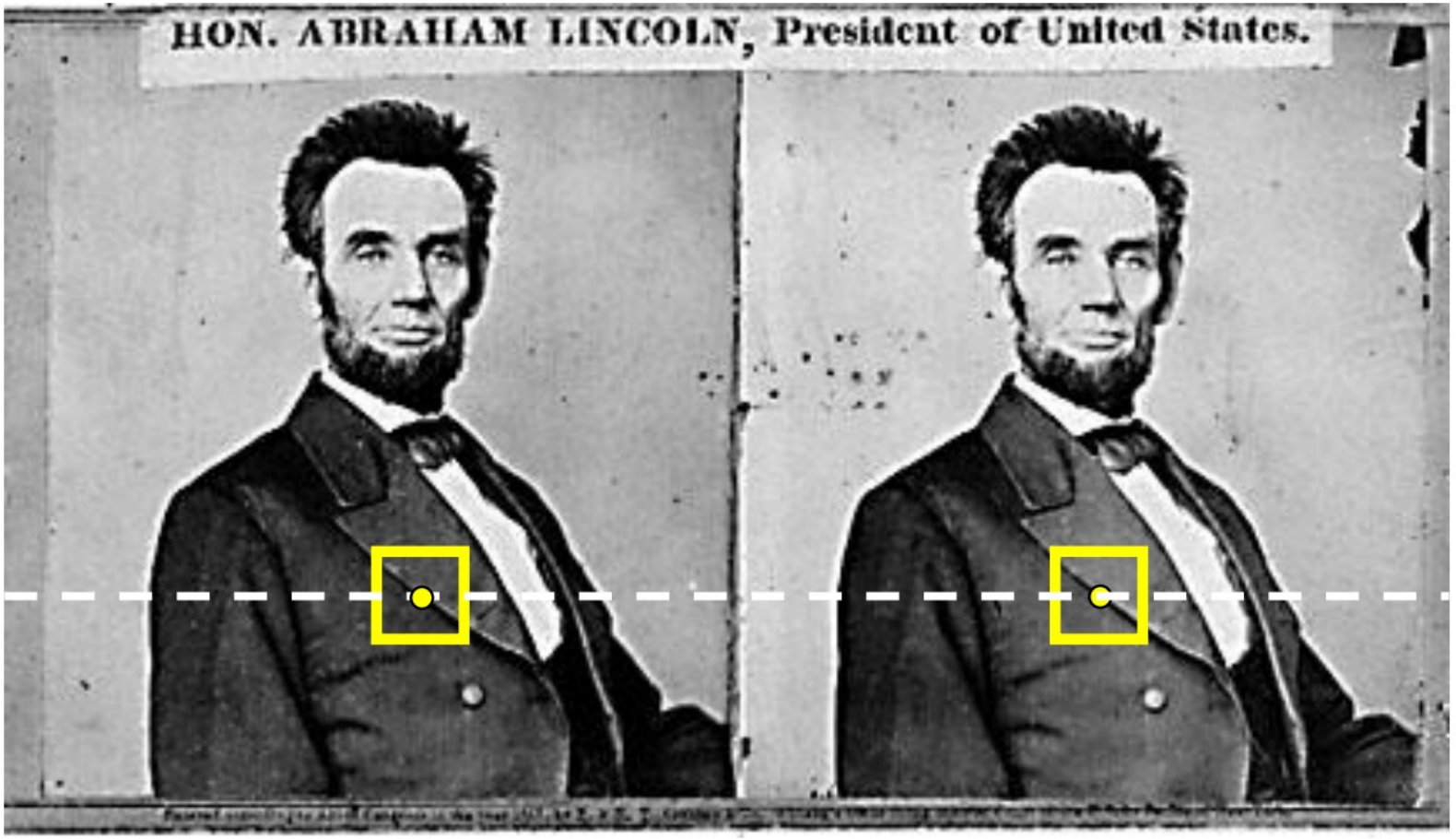
Option 2

1. Solve for correspondences
2. Estimate camera
3. Triangulate

Option 2

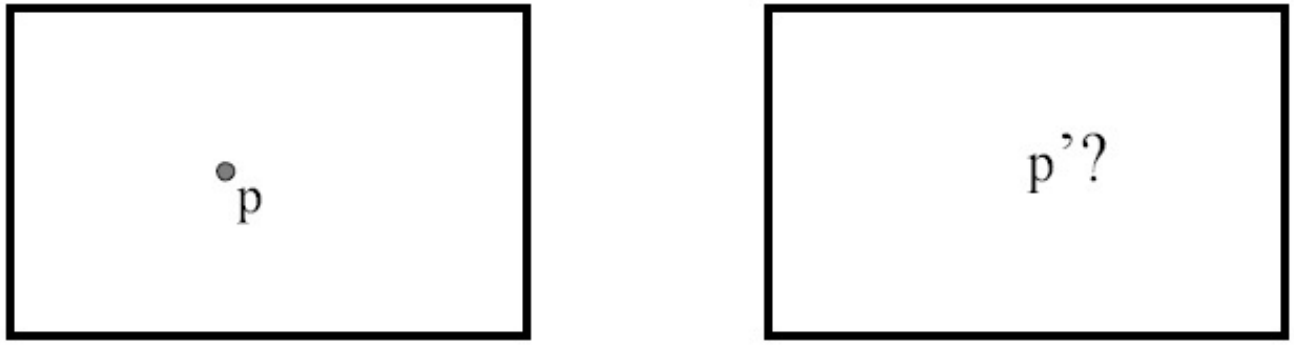
1. **Solve for correspondences**
2. **Estimate camera**
   - **What is the relationship between the camera + correspondences?**
3. Triangulate

Where do epipolar lines come from?



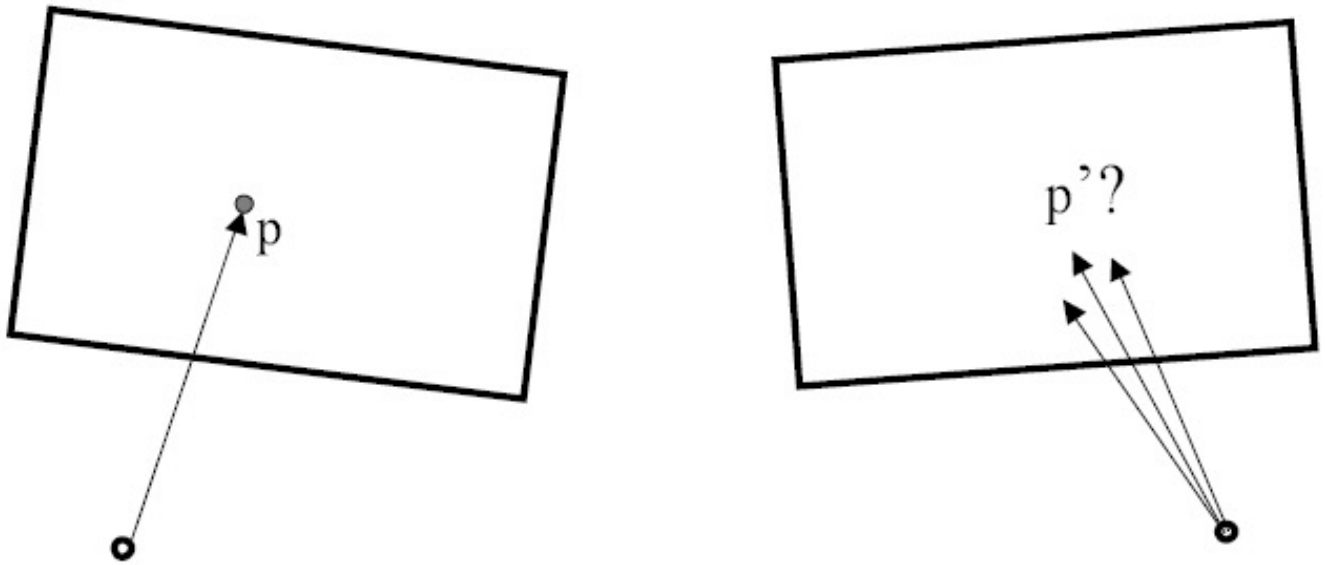
Slide courtesy of Noah Snavely

Stereo correspondence constraints



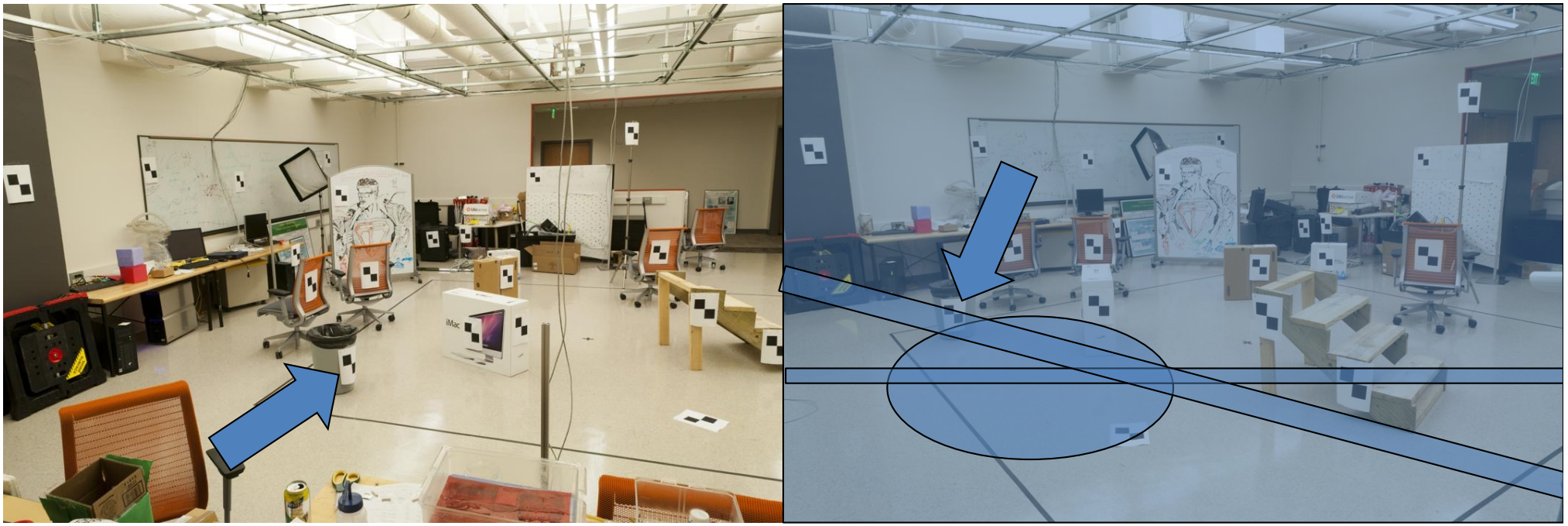
- Given p in left image, where can corresponding point p' be?

Stereo correspondence constraints



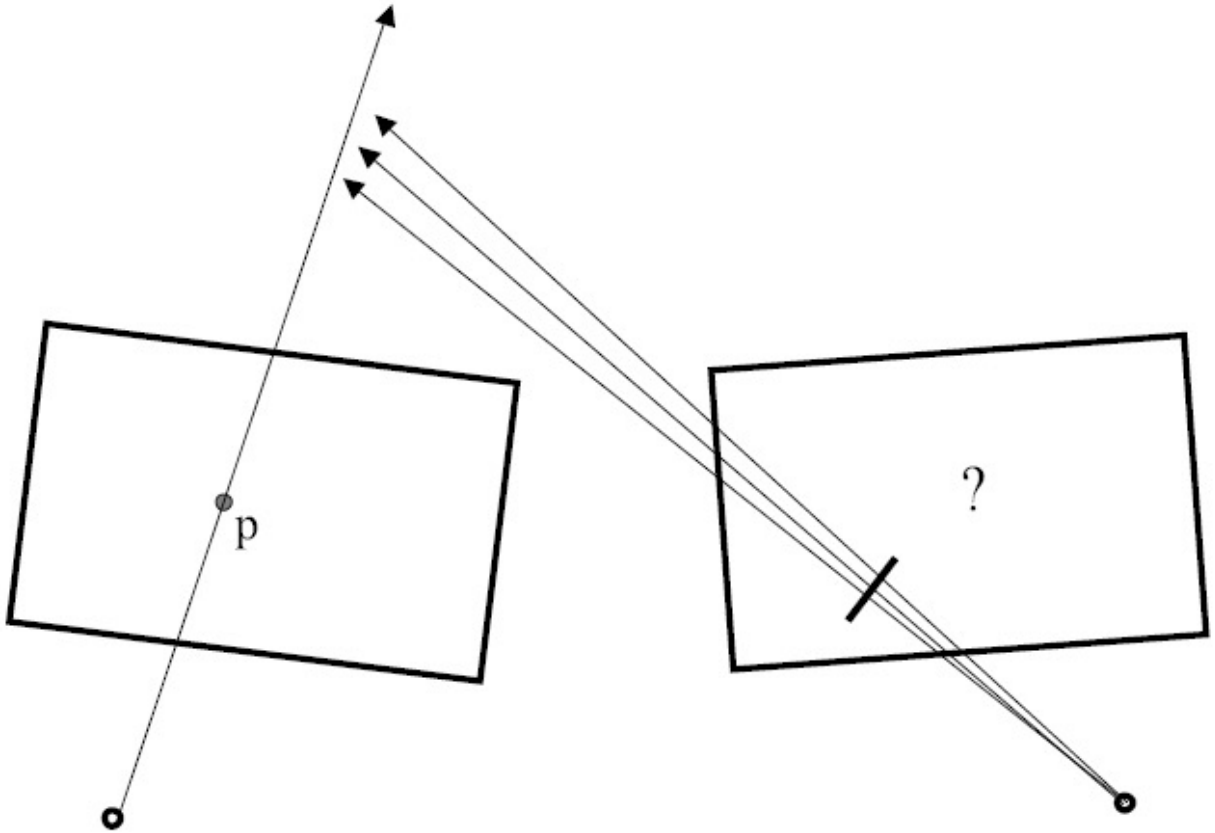
- Given p in left image, where can corresponding point p' be?

Where do we need to search?

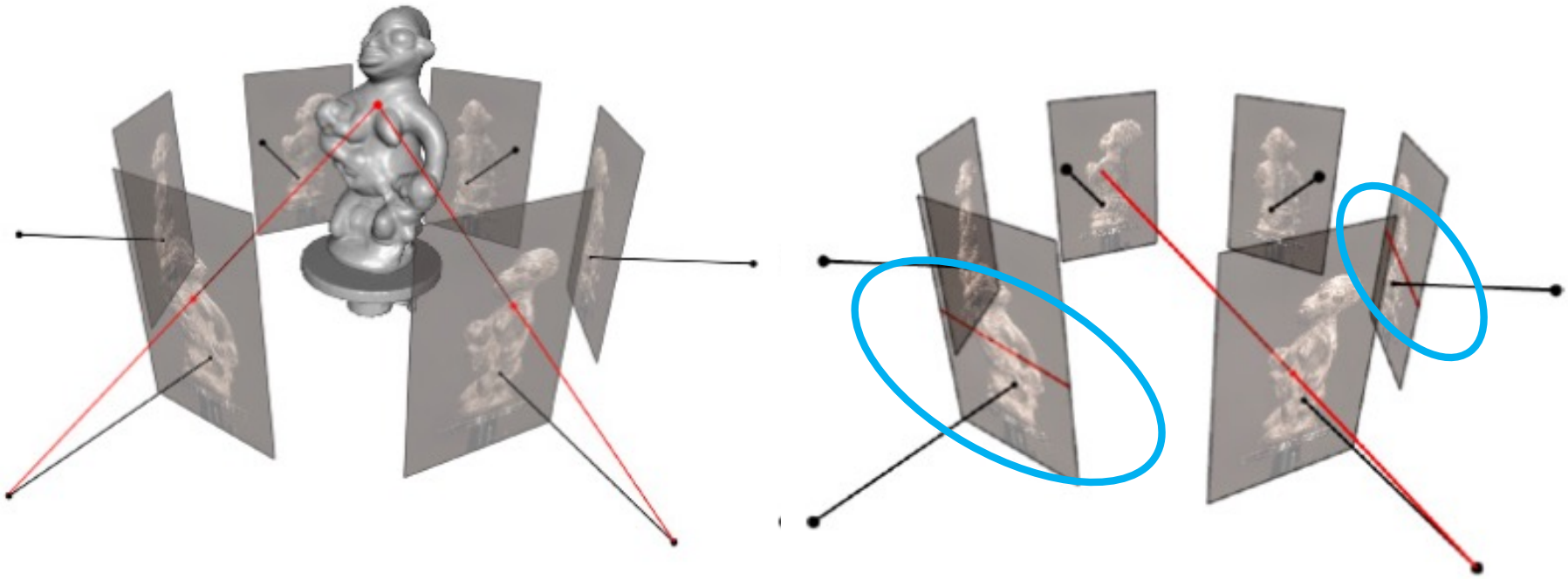


Slide by James Hays

Stereo correspondence constraints

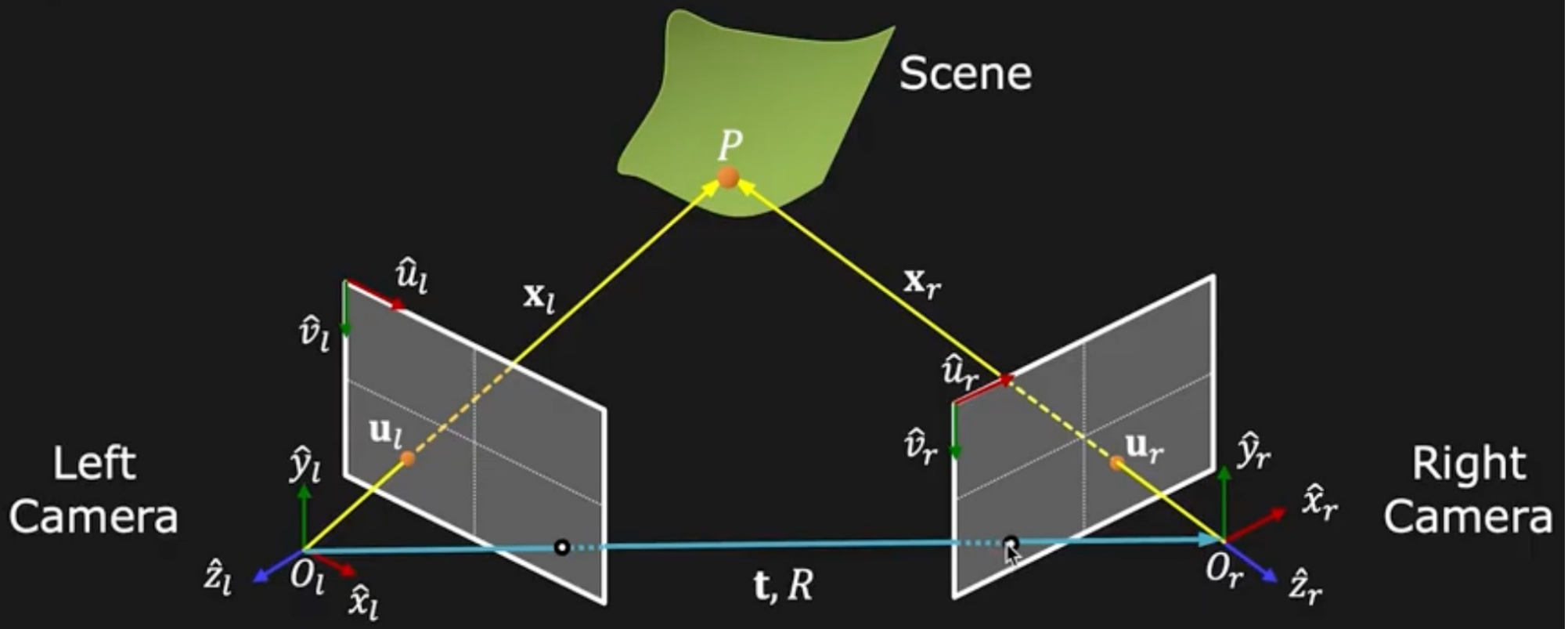


Epipolar Geometry



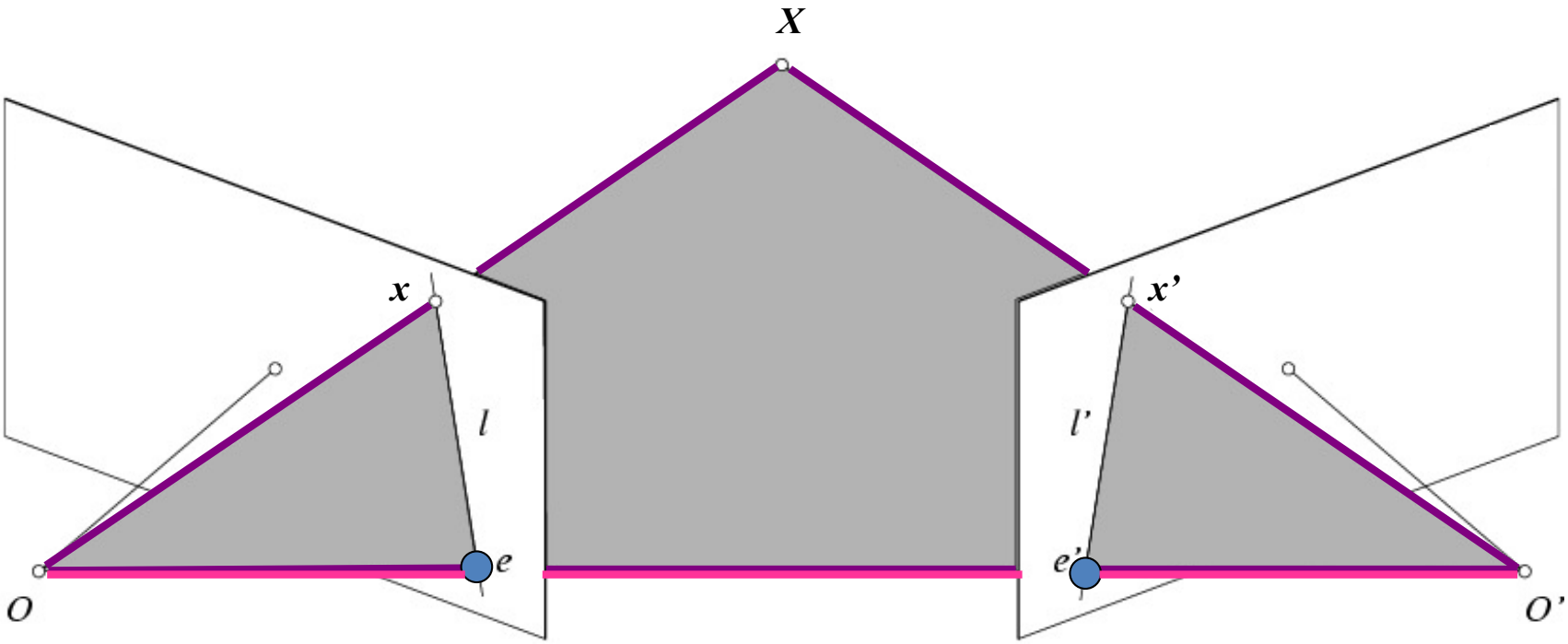
Figures by Carlos Hernandez

Epipolar Geometry: Epipoles



Epipole: Image point of origin/pinhole of one camera as viewed by the other camera.

Epipolar geometry



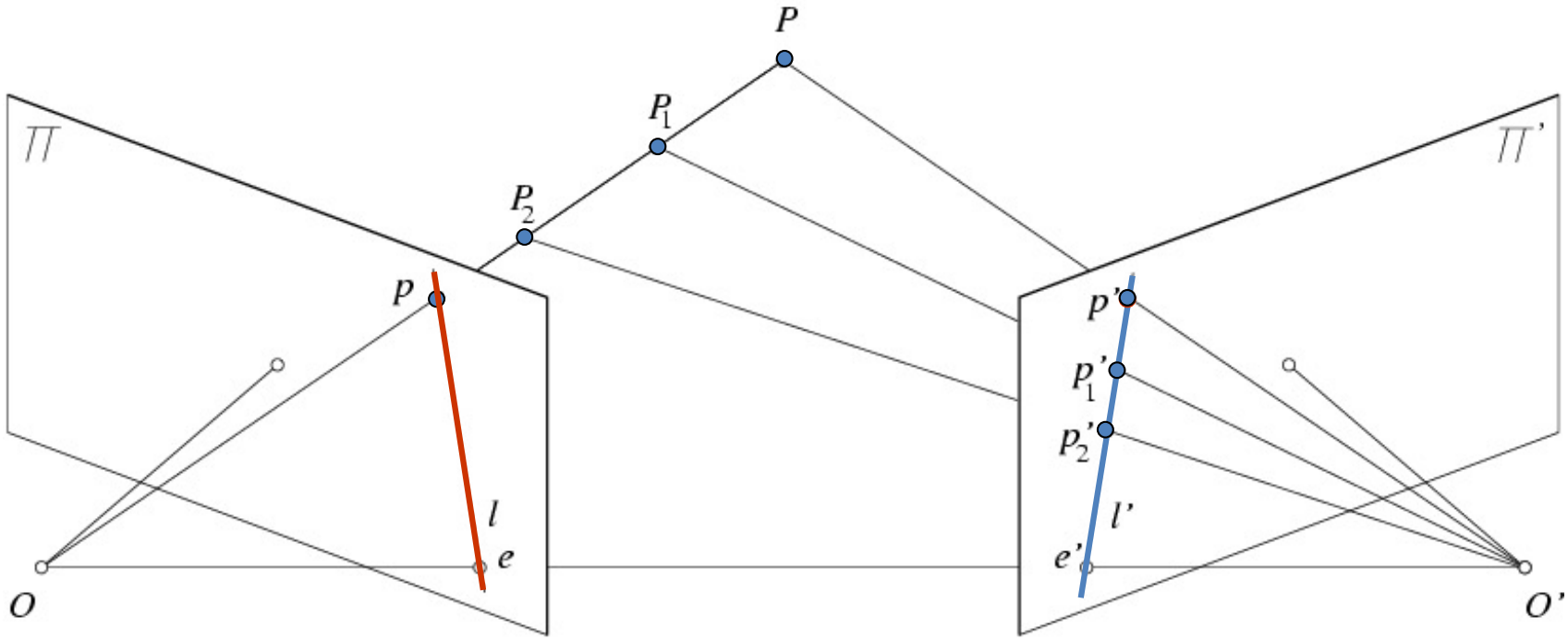
- **Baseline** – line connecting the two camera centers
- **Epipolar Plane** – plane containing baseline (1D family)
- **Epipoles**

= intersections of baseline with image planes
= projections of the other camera center
= vanishing points of the motion direction

The Epipole

Photo by Frank Dellaert

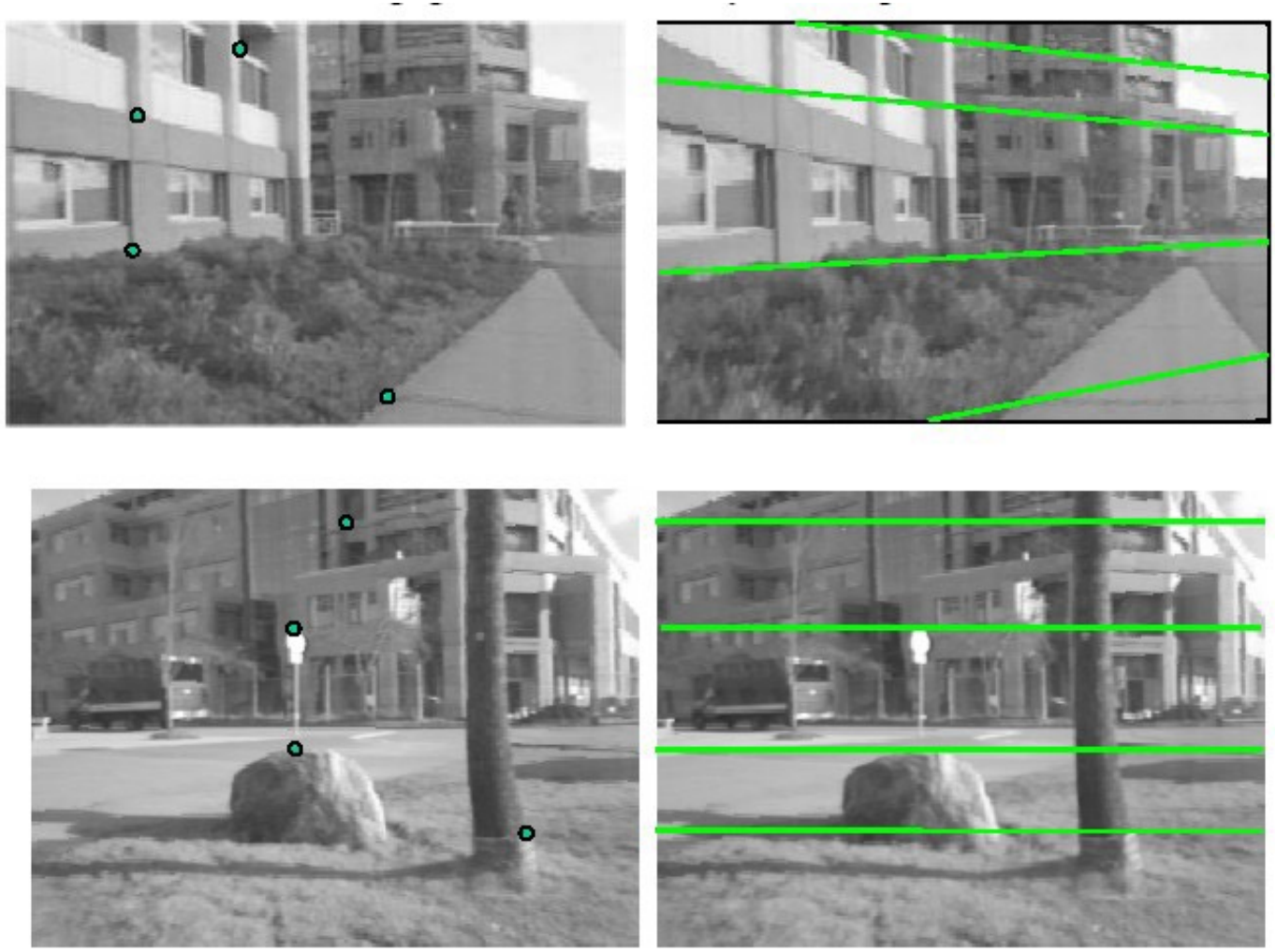
Epipolar constraint



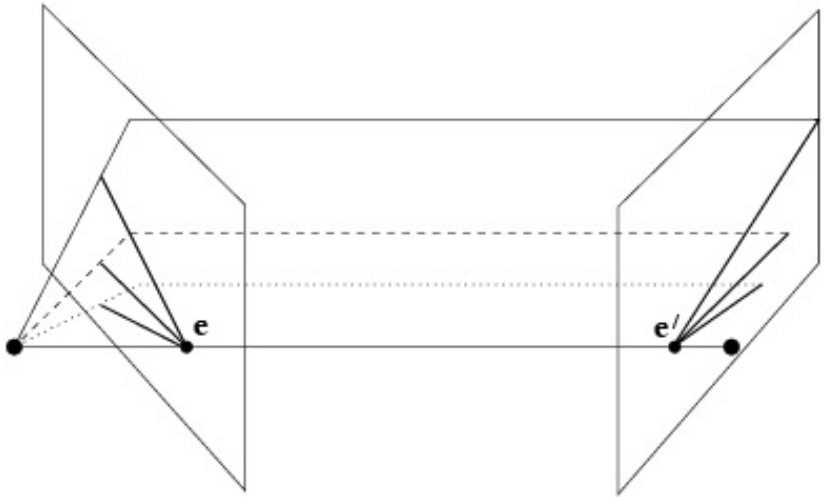
- Potential matches for *p* have to lie on the corresponding epipolar line *l'*.
- Potential matches for *p'* have to lie on the corresponding epipolar line *l*.

Source: M. Pollefeys

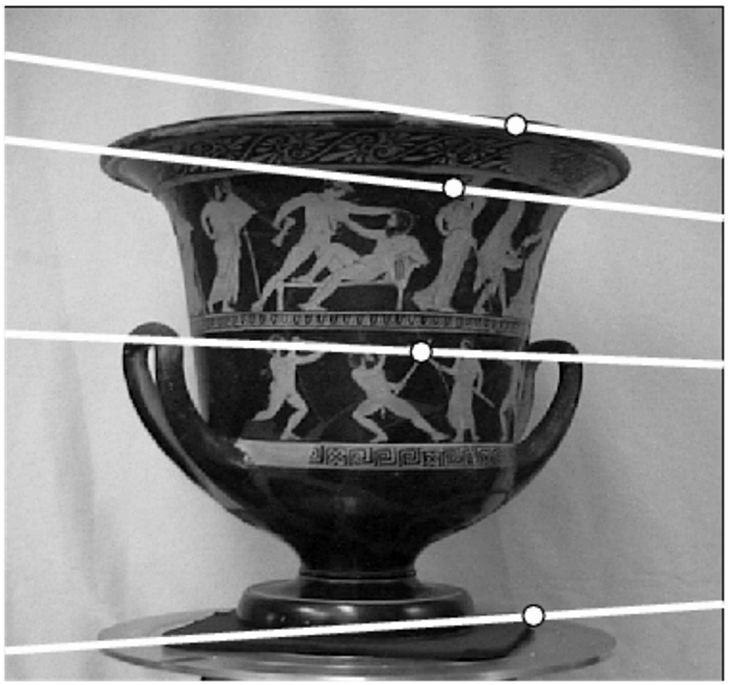
Example



Example: converging cameras



As position of 3d point varies, epipolar lines "rotate" about the baseline



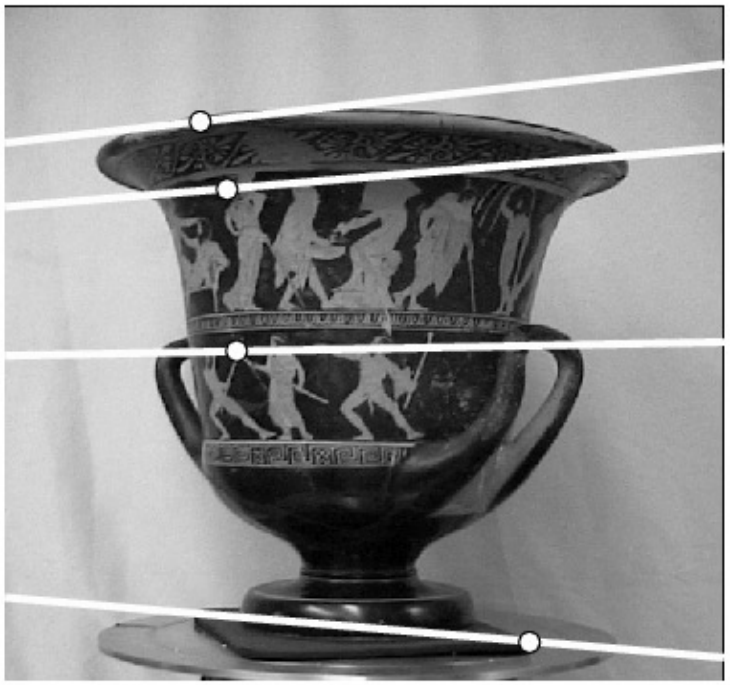


Figure from Hartley & Zisserman

Example: motion parallel with image plane

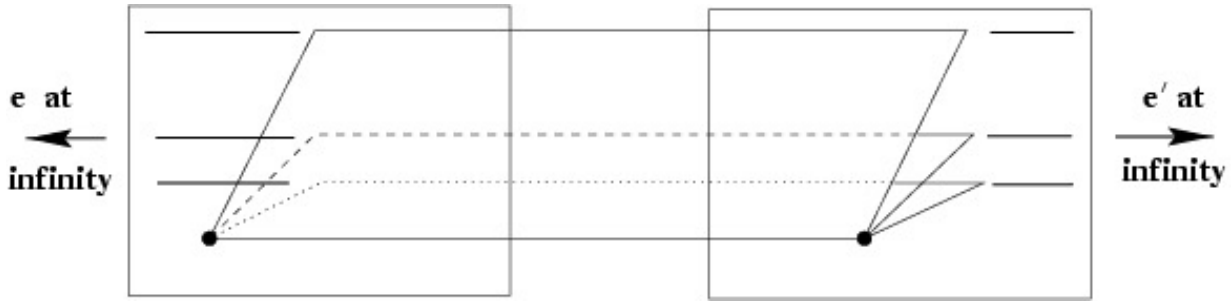
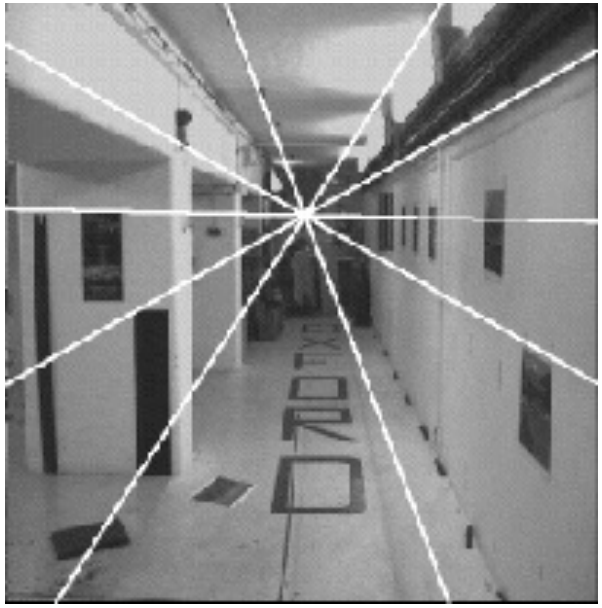
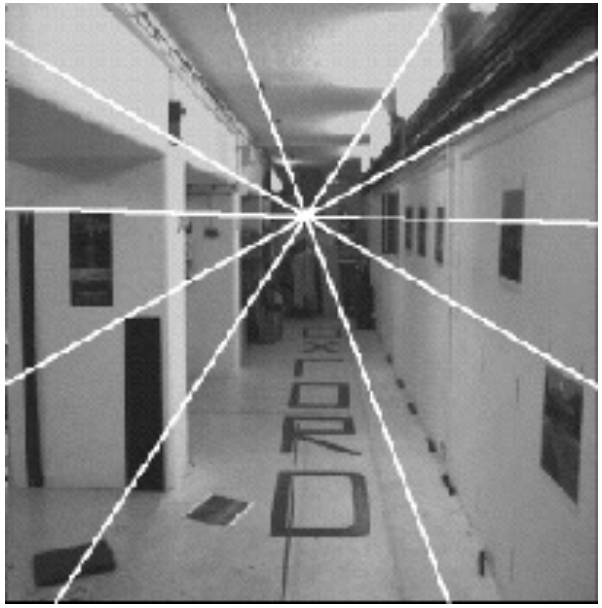
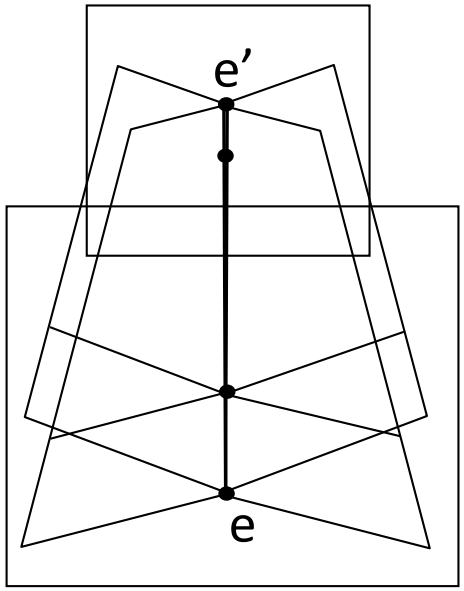


Figure from Hartley & Zisserman

Example: forward motion



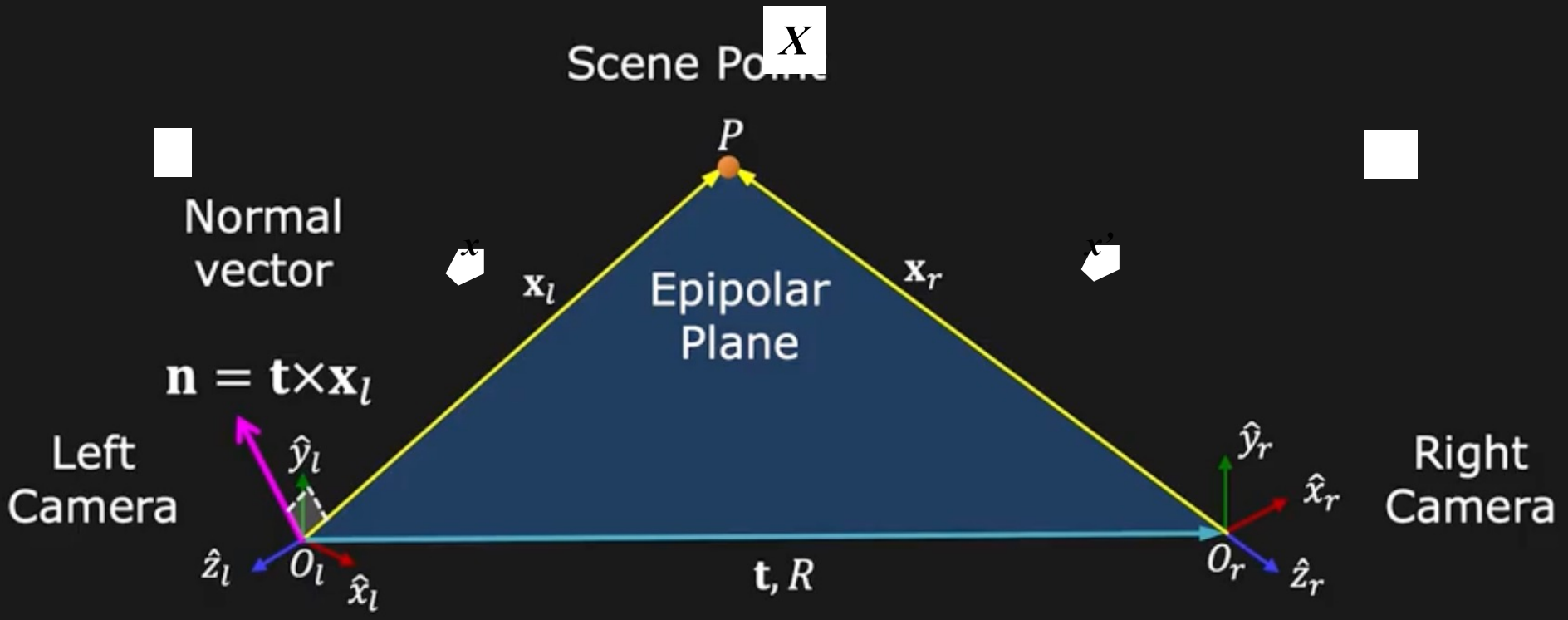




Epipole has same coordinates in both images.
Points move along lines radiating from e: “Focus of expansion”

Figure from Hartley & Zisserman

Epipolar Constraint



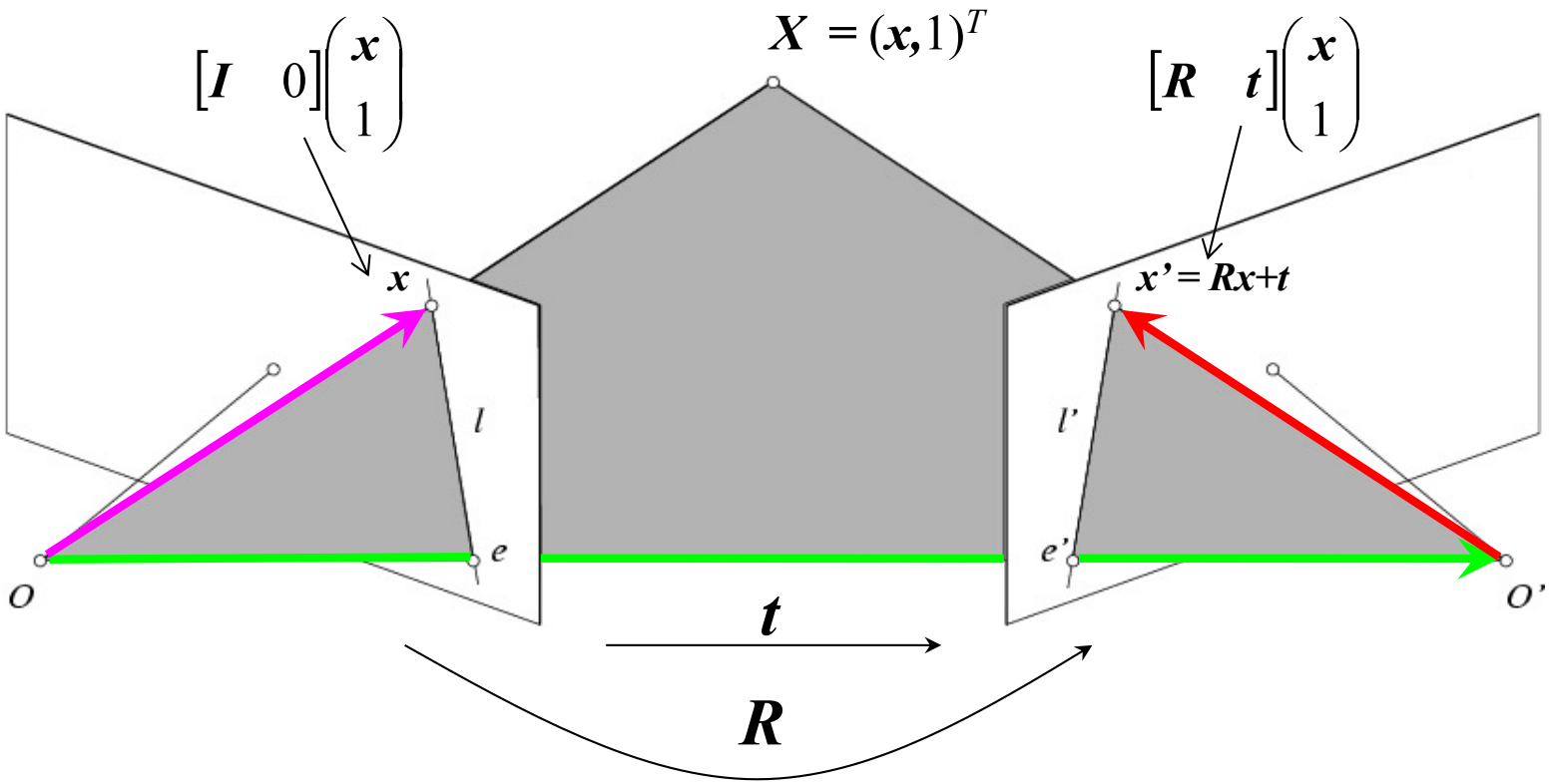
Vector normal to the epipolar plane: $\mathbf{n} = \mathbf{t}\times\mathbf{x}_l$

Dot product of $\mathbf{n}$ and $\mathbf{x}_l$ (perpendicular vectors) is zero:

$$\mathbf{x}_l \cdot (\mathbf{t}\times\mathbf{x}_l) = 0$$

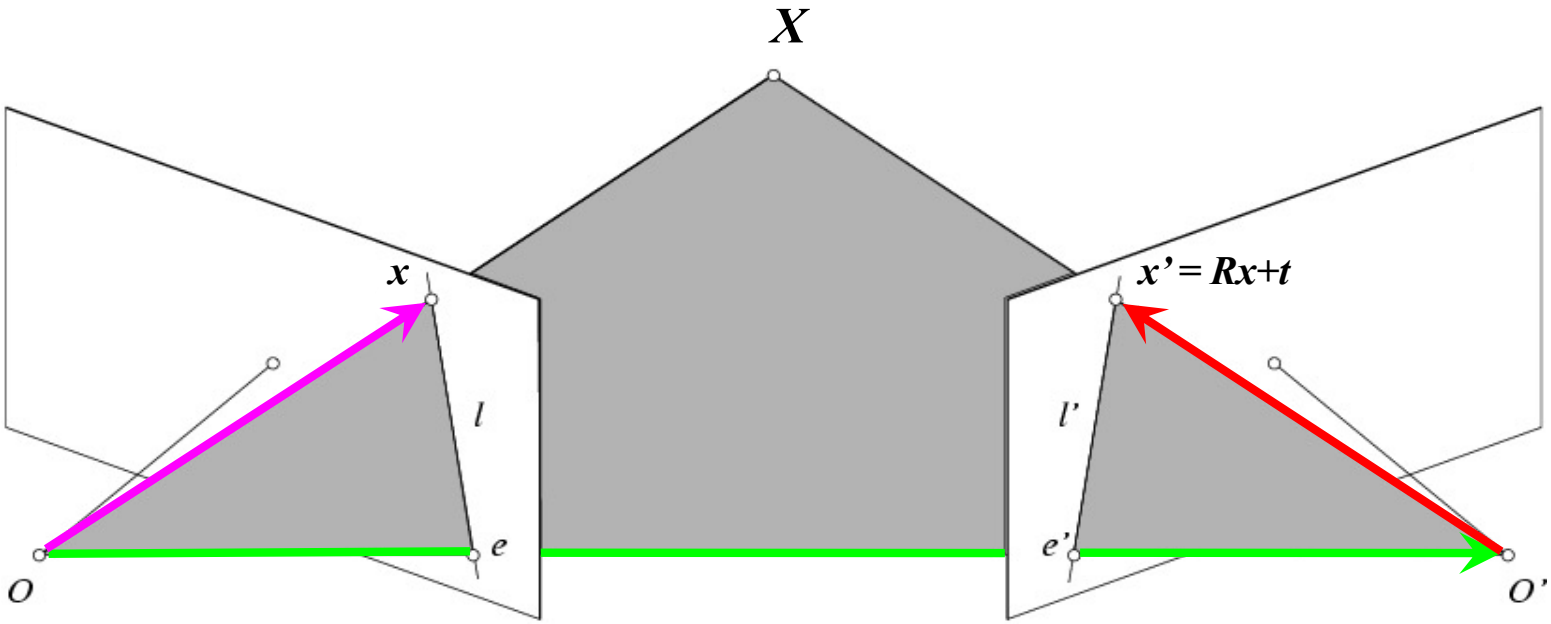
Slide Credit: Shree Nayar

Epipolar constraint: Calibrated case



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

Epipolar constraint: Calibrated case

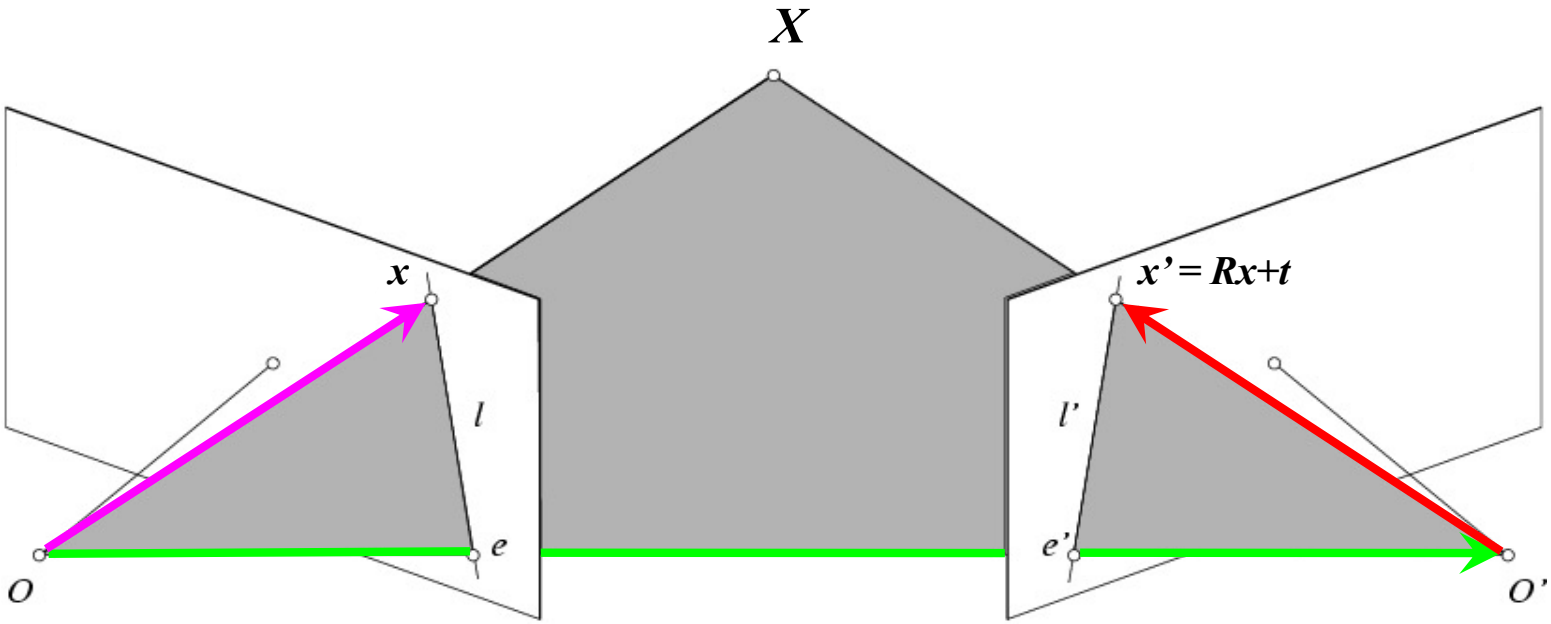


$$\boldsymbol{x}' \cdot [\boldsymbol{t} \times (\boldsymbol{R}\boldsymbol{x})] = 0 \quad \Rightarrow \quad \boldsymbol{x}'^{T} [\boldsymbol{t}_{\times}] \boldsymbol{R}\boldsymbol{x} = 0$$

Recall: $\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [\mathbf{a}_{\times}]\mathbf{b}$

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

Epipolar constraint: Calibrated case

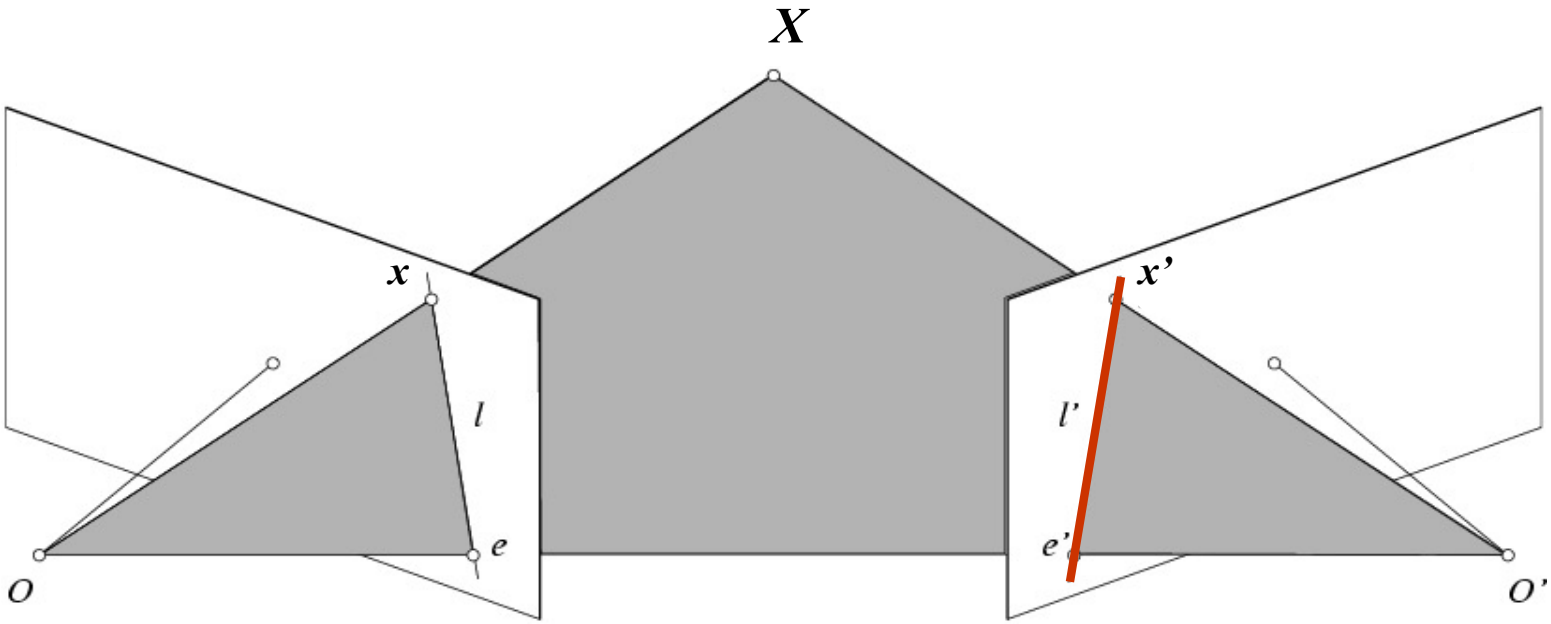


$$\boldsymbol{x}' \cdot [\boldsymbol{t} \times (\boldsymbol{R}\boldsymbol{x})] = 0 \quad \Rightarrow \quad \boldsymbol{x}'^{T} [\boldsymbol{t}_{\times}] \boldsymbol{R}\boldsymbol{x} = 0 \quad \Rightarrow \quad \boldsymbol{x}'^{T} \boldsymbol{E}\boldsymbol{x} = 0$$

Essential Matrix
(Longuet-Higgins, 1981)

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

Epipolar constraint: Calibrated case



$$\boldsymbol{x}'^{T} E \boldsymbol{x} = 0$$

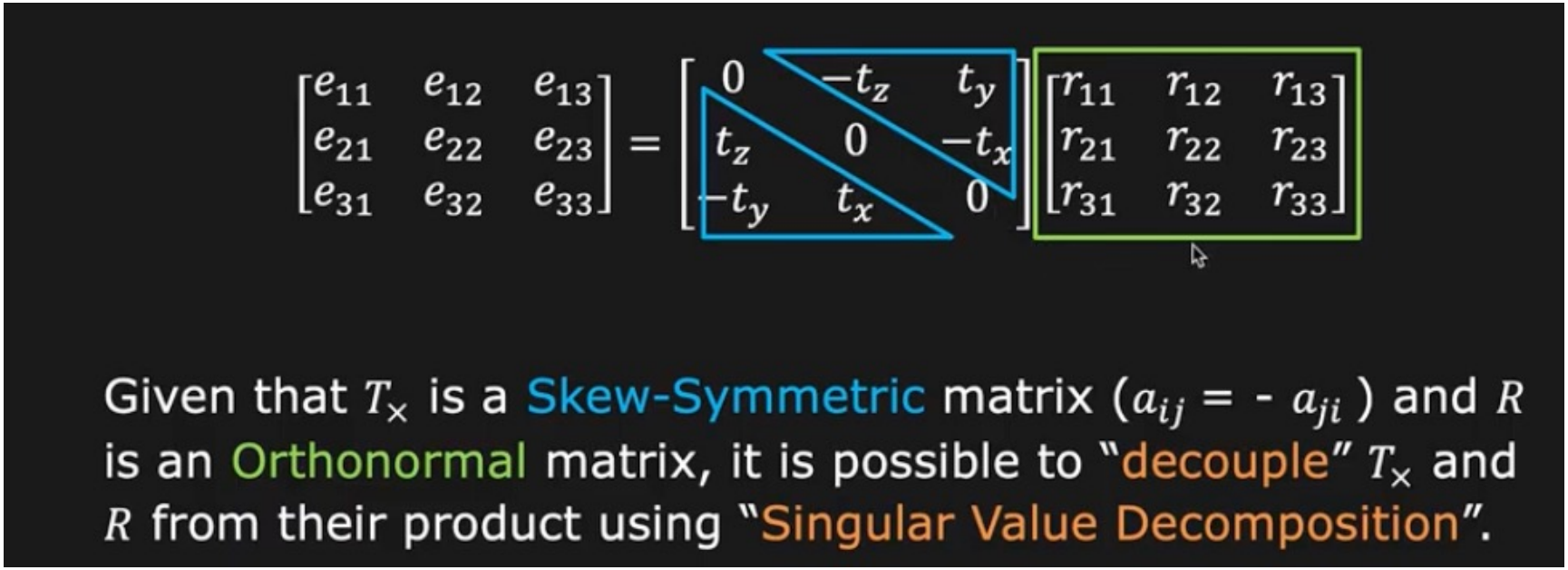
- ***E x*** is the epipolar line associated with ***x*** (***l'*** = ***E x***)
  - Recall: a line is given by $ax + by + c = 0$ or

$$\mathbf{l}^T \mathbf{x} = 0 \quad \text{where} \quad \mathbf{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

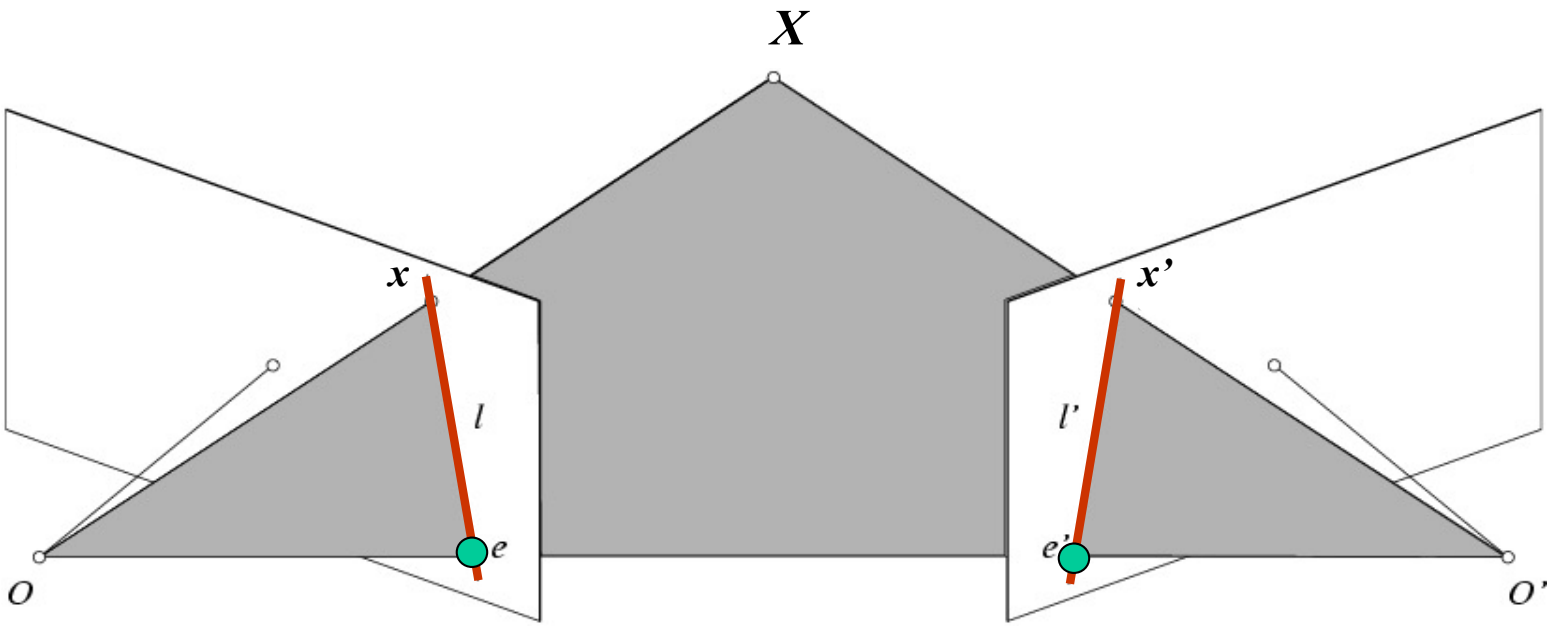
Essential matrix can be decomposed

$E = T_x R$

If we know E, we can recover t and R



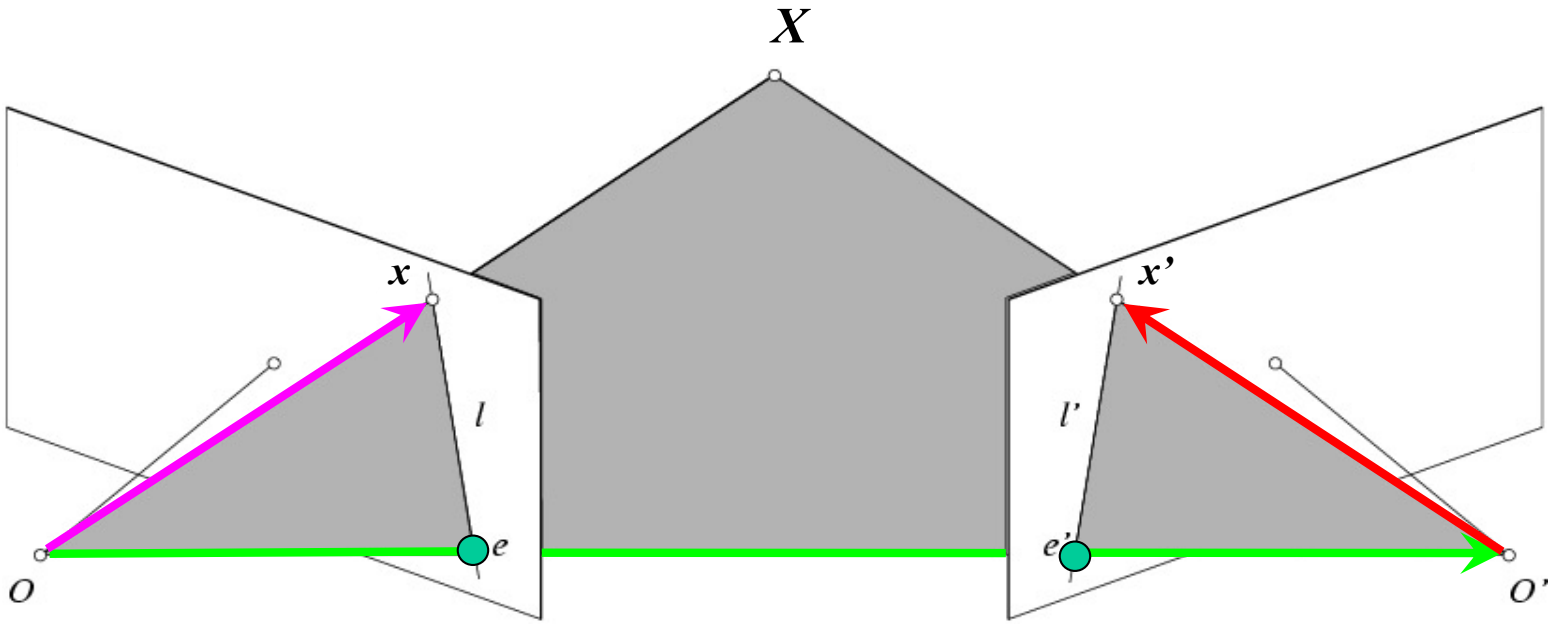
Epipolar constraint: Calibrated case



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

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

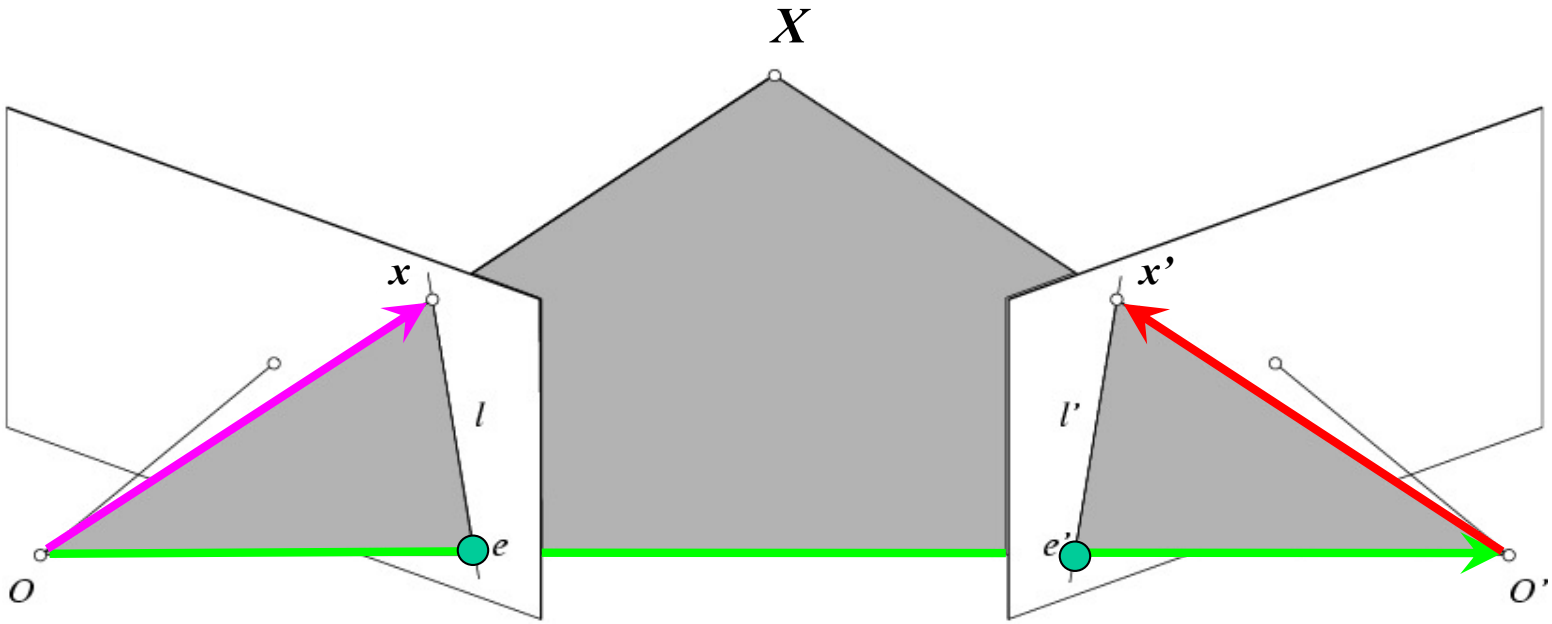
Epipolar constraint: Uncalibrated case



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

$$\hat{\boldsymbol{x}}'^{T}\boldsymbol{E}\hat{\boldsymbol{x}}=0 \qquad \hat{\boldsymbol{x}}=\boldsymbol{K}^{-1}\boldsymbol{x},\quad \hat{\boldsymbol{x}}'=\boldsymbol{K}'^{-1}\boldsymbol{x}'$$

Epipolar constraint: Uncalibrated case



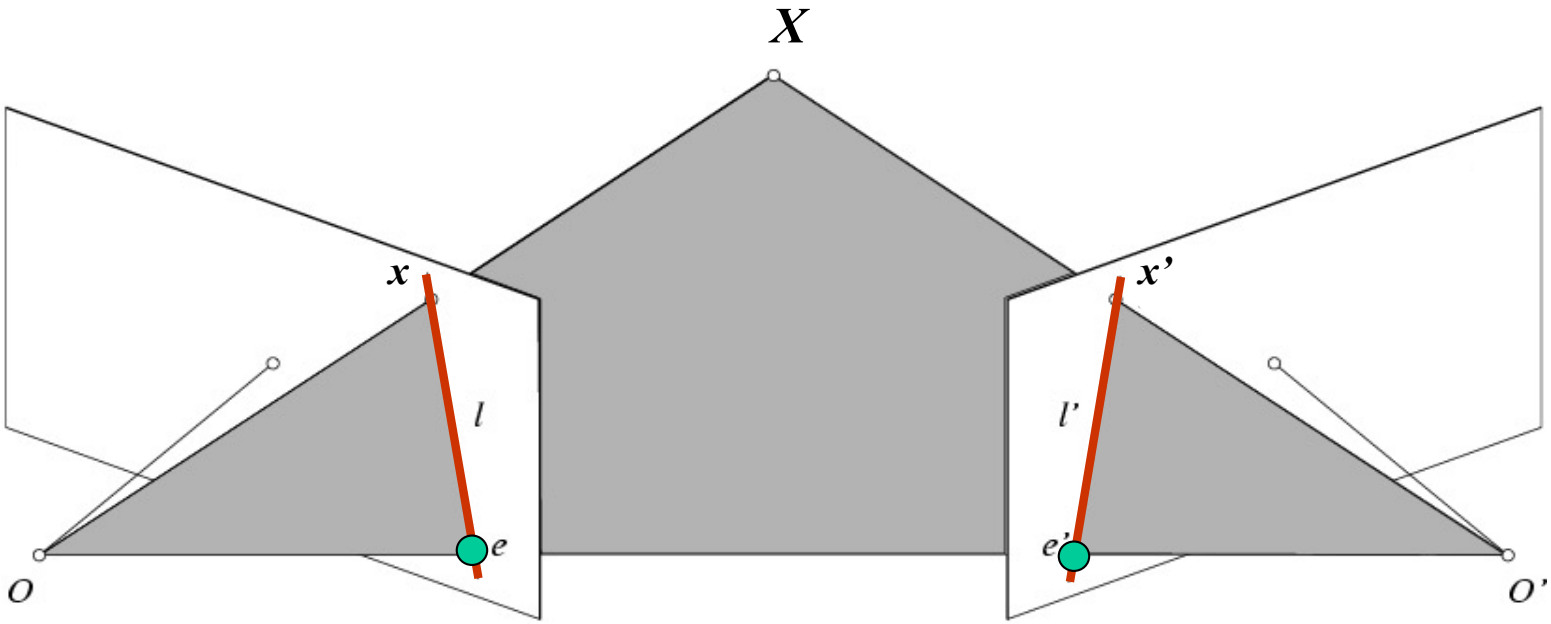
$$\hat{\boldsymbol{x}}'^{T}\boldsymbol{E}\hat{\boldsymbol{x}}=0 \quad \Rightarrow \quad \boldsymbol{x}'^{T}\boldsymbol{F}\boldsymbol{x}=0 \quad \text{with} \quad \boldsymbol{F}=\boldsymbol{K}'^{-T}\boldsymbol{E}\boldsymbol{K}^{-1}$$

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

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

Fundamental Matrix
(Faugeras and Luong, 1992)

Epipolar constraint: Uncalibrated case



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

- $\boldsymbol{F}\,\boldsymbol{x}$ is the epipolar line associated with $\boldsymbol{x}$ ($\boldsymbol{l}' = \boldsymbol{F}\,\boldsymbol{x}$)
- $\boldsymbol{F}^{T}\boldsymbol{x}'$ is the epipolar line associated with $\boldsymbol{x}'$ ($\boldsymbol{l} = \boldsymbol{F}^{T}\boldsymbol{x}'$)
- $\boldsymbol{F}\,\boldsymbol{e} = 0$ and $\boldsymbol{F}^{T}\boldsymbol{e}' = 0$
- $\boldsymbol{F}$ is singular (rank two)
- $\boldsymbol{F}$ has *seven* degrees of freedom

Recall the homography

Homography Matrix: 3x3

Fundamental Matrix is also 3x3.

Homography is a special case of the
Fundamental matrix, for planar scenes

How do we find E or F?

Relates 3D position (x_l, y_l, z_l) of scene point w.r.t left camera to its 3D position (x_r, y_r, z_r) w.r.t. right camera

$$\mathbf{x}_l^T E \mathbf{x}_r = 0$$

$$[x_l \quad y_l \quad z_l] \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} = 0$$

3D position in left camera coordinates

3x3 Essential Matrix

3D position in right camera coordinates

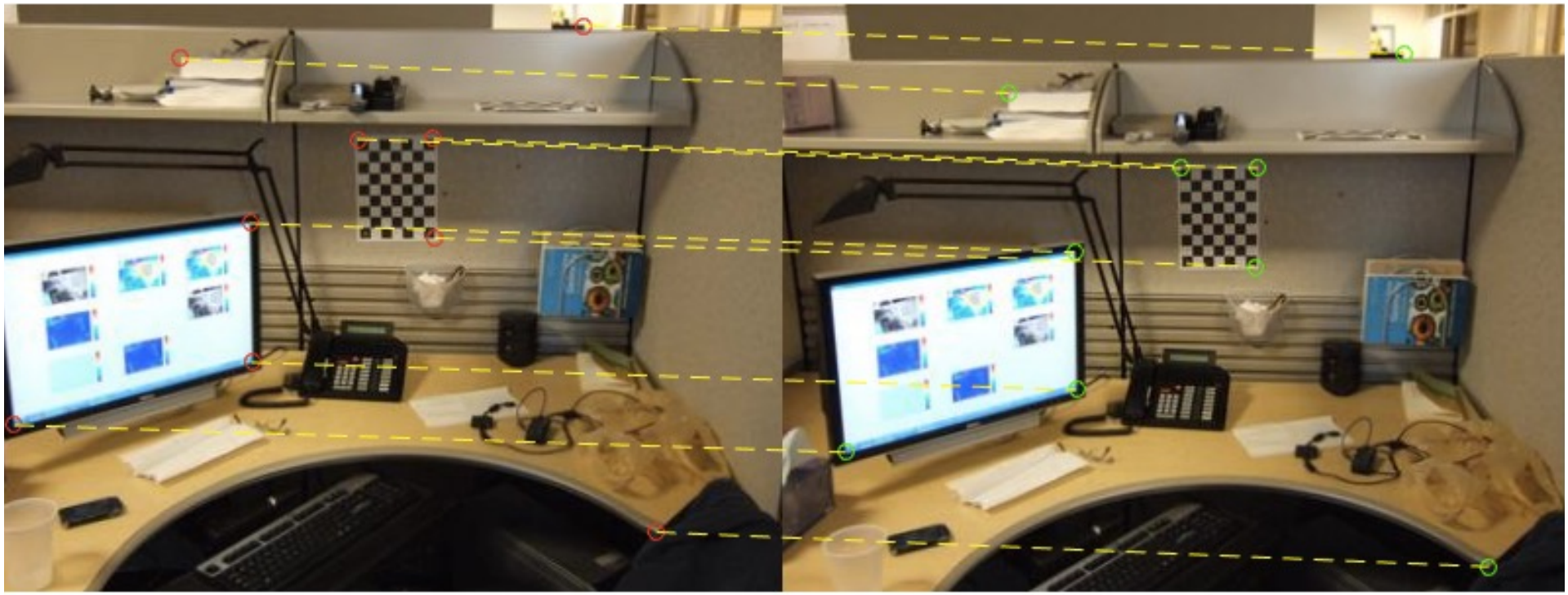
Unfortunately, we don't have $\mathbf{x}_l$ and $\mathbf{x}_r$.

But we know the image coordinates!

z's are still unknown, but they are > 0 and !=0

$$[u_l \quad v_l \quad 1]\, z_l\, K_l^{-1^T} \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} K_r^{-1}\, z_r \begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} = 0$$

Estimating the fundamental matrix



The eight-point algorithm

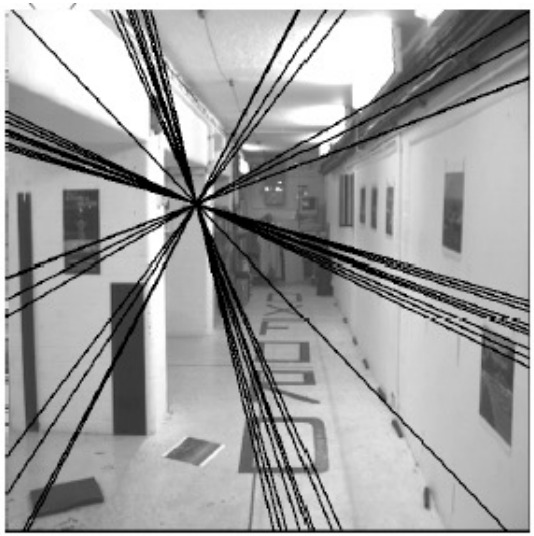
$$\boldsymbol{x} = (u, v, 1)^T, \quad \boldsymbol{x}' = (u', v', 1)$$

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

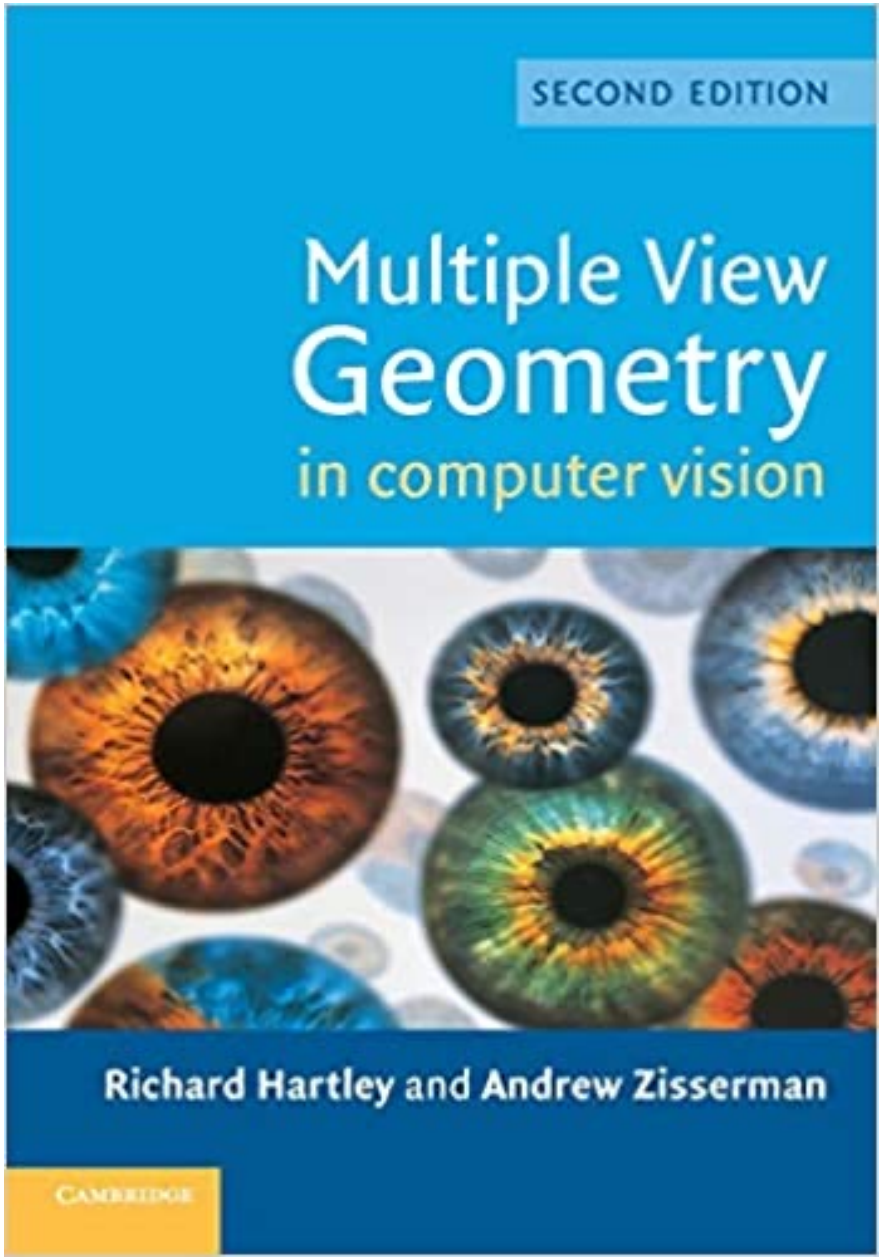
$$\begin{bmatrix} u'u & u'v & u' & v'u & v'v & v' & u & v & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

Solve homogeneous linear system using eight or more matches

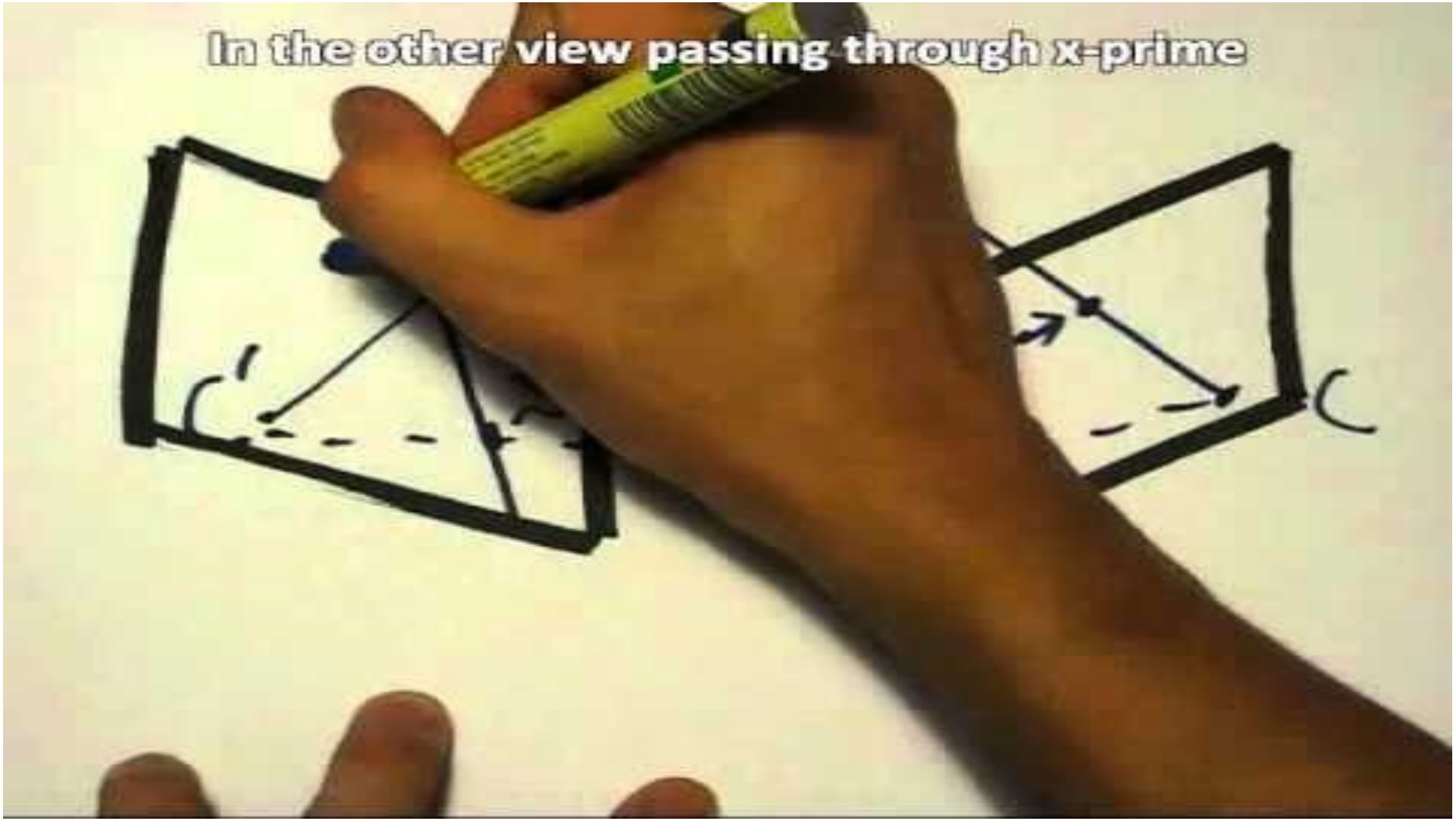
Enforce rank-2 constraint (take SVD of ***F*** and throw out the smallest singular value)



The Bible by Hartley & Zisserman



The Fundamental Matrix Song



http://danielwedge.com/fmatrix/
https://www.youtube.com/watch?time_continue=8&v=DgGV3I82NTk&feature=emb_title

Option 2: Depth from arbitrary two cameras

1. Solve for correspondences
2. Estimate camera
   - What is the relationship between the camera + correspondences? → Epipolar geometry!
3. **Triangulate**

Finally: computing depth by triangulation

Now we know about the camera, K_1, K_2 and [R t]:

Left Camera Imaging Equation

$$\begin{bmatrix} u_l \\ v_l \\ 1 \end{bmatrix} \equiv \begin{bmatrix} f_x^{(l)} & 0 & o_x^{(l)} & 0 \\ 0 & f_y^{(l)} & o_y^{(l)} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_l \\ y_l \\ z_l \\ 1 \end{bmatrix}$$

Right Camera Imaging Equation

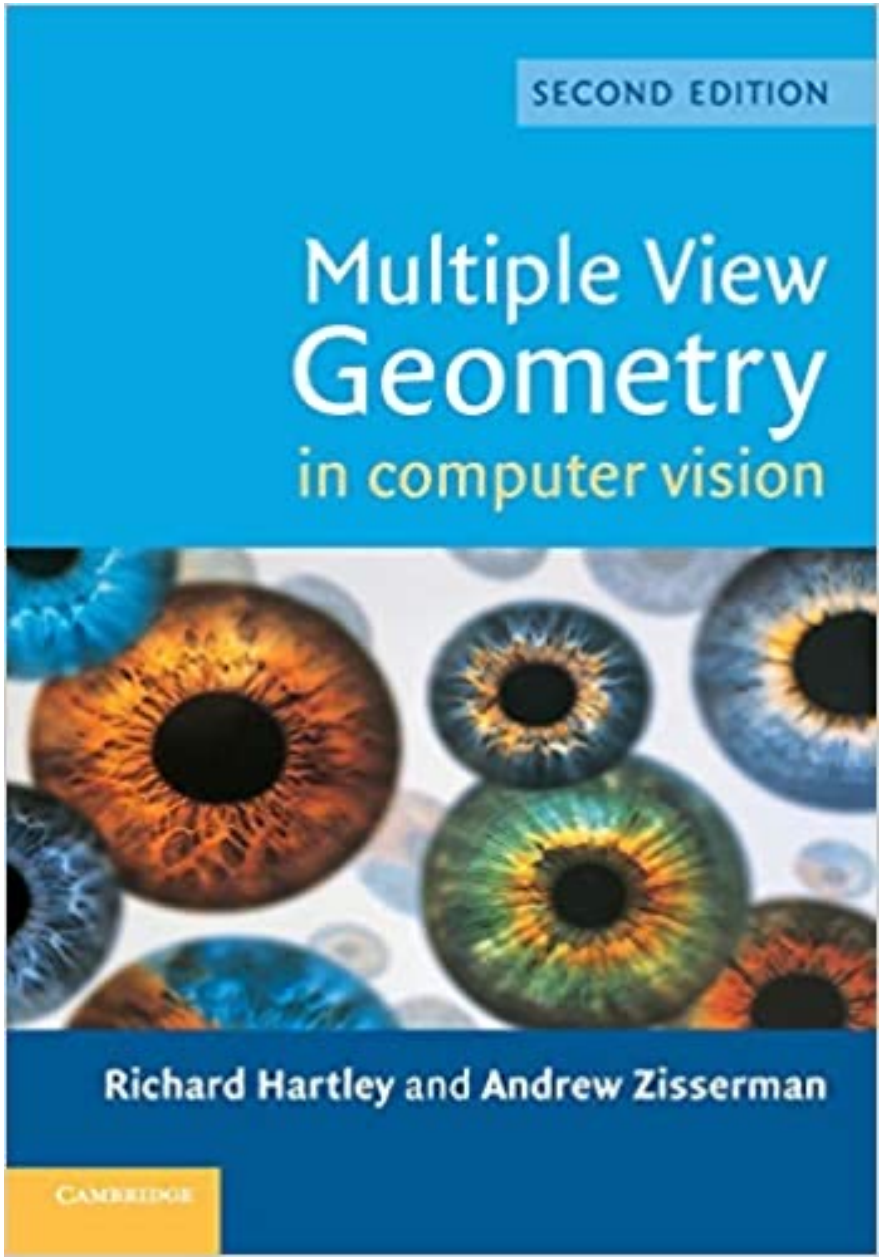
$$\begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} \equiv \begin{bmatrix} f_x^{(r)} & 0 & o_x^{(r)} & 0 \\ 0 & f_y^{(r)} & o_y^{(r)} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ z_r \\ 1 \end{bmatrix}$$

$$X_l = RX_r + t$$

Substitute X_l in terms of X_r

Then solve for X_r with least squares!

The Bible by Hartley & Zisserman



What about more than two views?

The geometry of three views is described by a 3 x 3 x 3 tensor called the *trifocal tensor*

The geometry of four views is described by a 3 x 3 x 3 x 3 tensor called the *quadrifocal tensor*

After this it starts to get complicated…

Slide courtesy of Noah Snavely

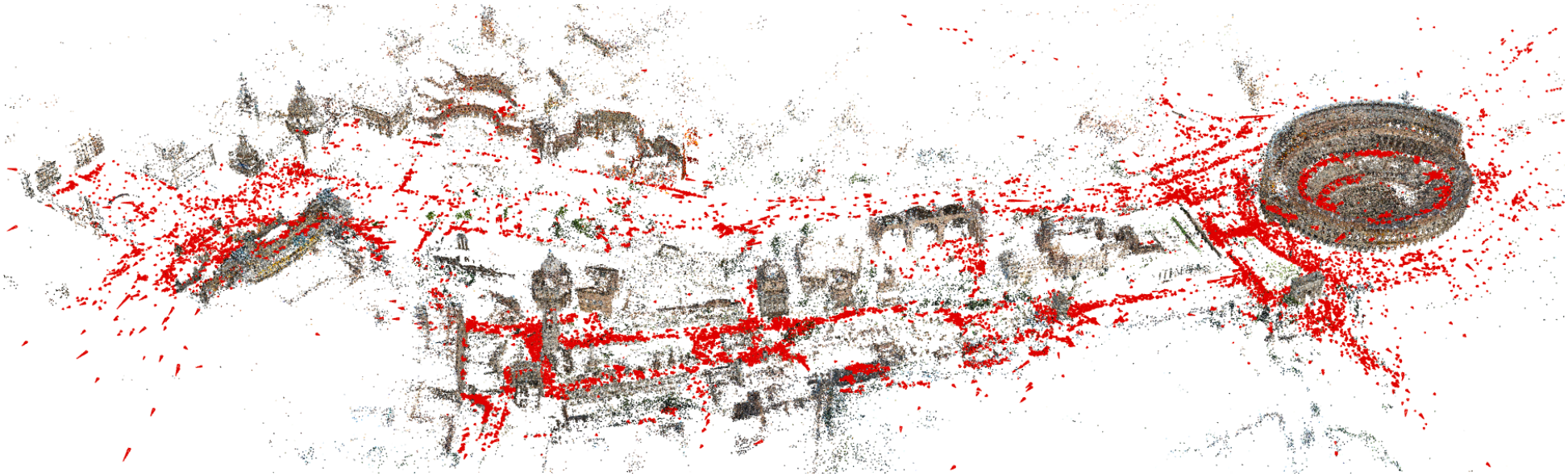
Large-scale structure from motion

Dubrovnik, Croatia. 4,619 images (out of an initial 57,845).
Total reconstruction time: 23 hours
Number of cores: 352

Building Rome in a Day, Agarwal et al. ICCV 2009

Slide courtesy of Noah Snavely

Large-scale structure from motion



Result using COLMAP: Schönberger et al. CVPR ’16