Lecture 17: Multi-view Geometry

Announcements

- New IA office hours, starting next week

Name	Office hour times
Andrew Owens	Fri. 3:00pm - 4:00pm
Haozhu Wang	Thu. 7:30pm - 8:30pm
Bingqi Sun	Mon. 4:30 - 5:30
Anthony Liang	Tue. 1:30 - 2:30pm

- Instead of raising your hand: send a message, then (after I answer) unmute and ask your question
- Please send us feedback!

Today

- Review image formation
- Epipolar geometry
- Image alignment

Recall: homogeneous coordinates

Representing translations:

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image coordinates

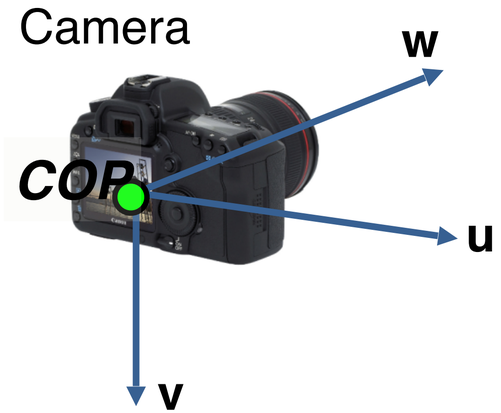
$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix}$$

Converting from homogeneous coordinates:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

Source: N. Snavely

Recall: camera parameters



"The World"

Three important coordinate systems:

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

How do we project a given world point (x, y, z) to an image point?

Source: N. Snavely

Recall: camera parameters

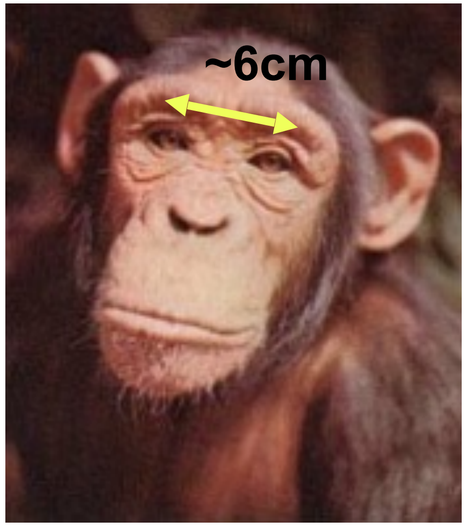
$$\mathbf{\Pi} = \begin{bmatrix} f & s & c_x \\ 0 & \alpha f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3\times 3} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3\times 3} & \mathbf{T}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix}$$

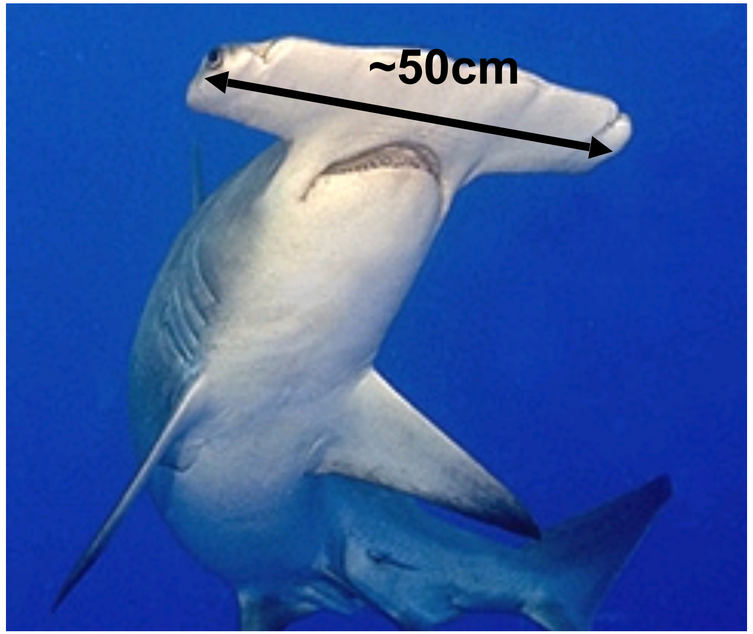
intrinsics rotation translation

Source: N. Snavely

Estimating depth from multiple views

Stereo vision





Source: Torralba, Isola, Freeman

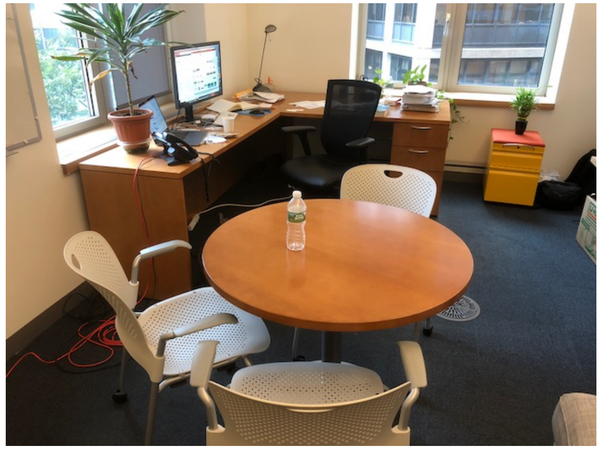
1, 2, N eyes

Source: Torralba, Isola, Freeman

1, 2, N eyes

Source: Torralba, Isola, Freeman

1, 2, N eyes



Source: Torralba, Isola, Freeman

Depth without objects

Random dot stereograms (Bela Julesz)

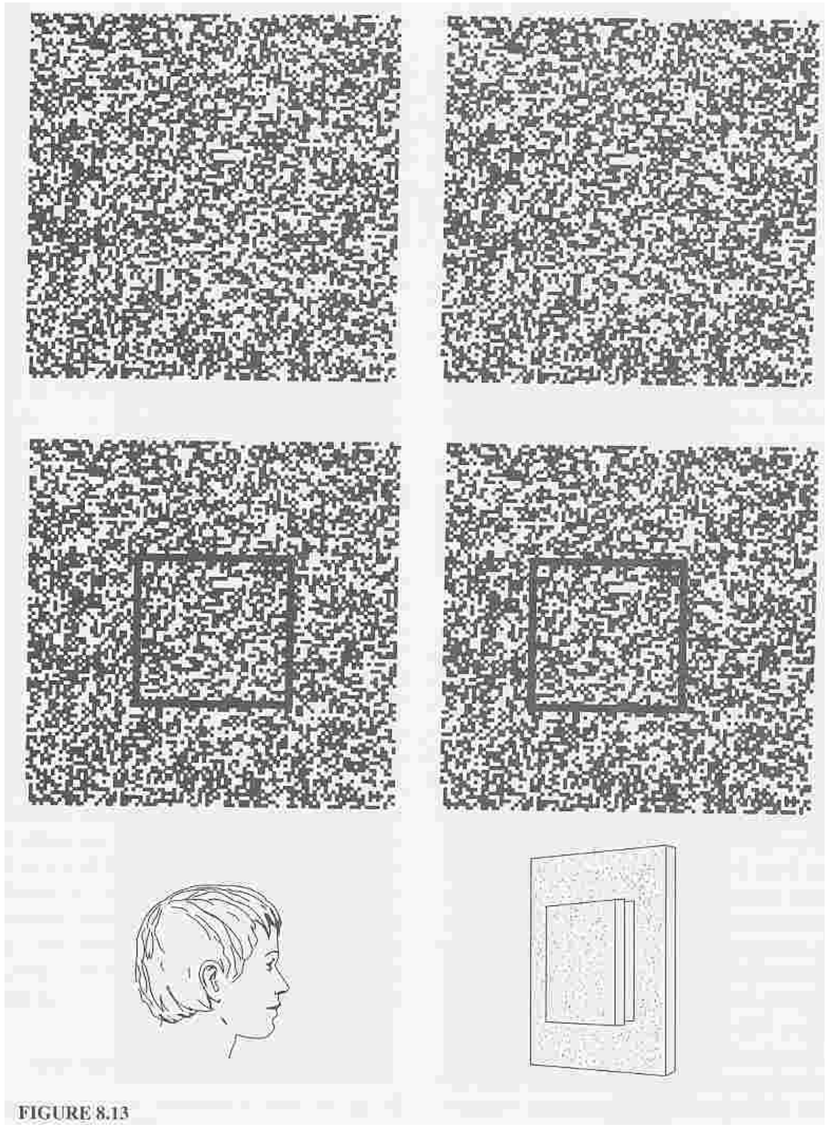
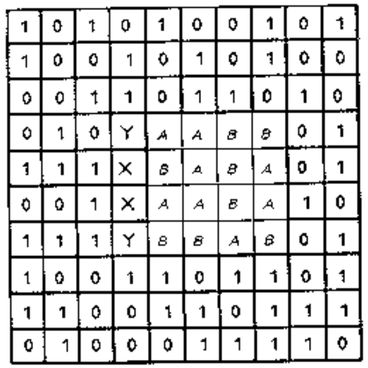
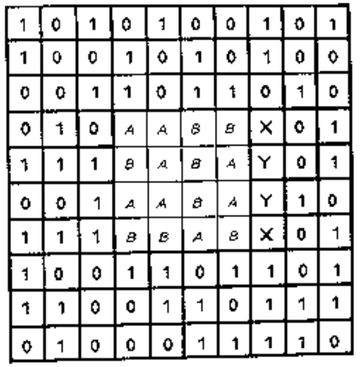


FIGURE 8.13





Julesz, 1971

Source: Torralba, Isola, Freeman

Brewster-type stereoscope, 1870

More details

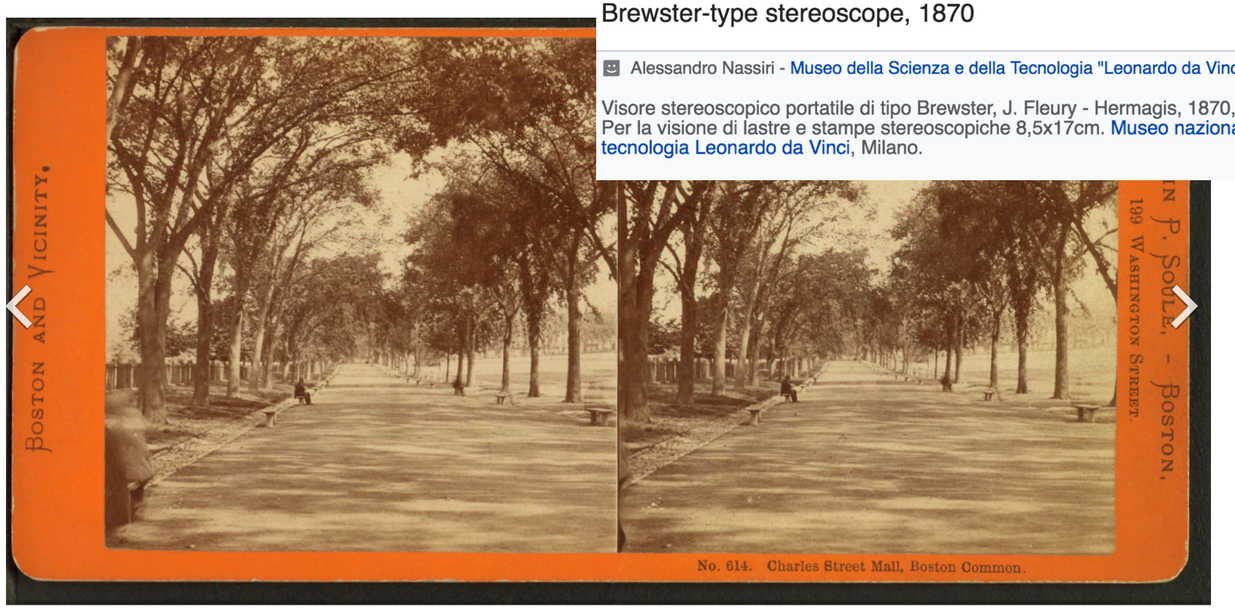
Alessandro Nassiri - Museo della Scienza e della Tecnologia "Leonardo da Vinci"

Visore stereoscopico portatile di tipo Brewster, J. Fleury - Hermagis, 1870, con messa a fuoco manuale. Per la visione di lastre e stampe stereoscopiche 8,5x17cm. Museo nazionale della scienza e della tecnologia Leonardo da Vinci, Milano.

CC BY-SA 4.0

File: IGB 006055 Visore stereoscopico portatile Museo scienza e tecnologia Milano.jpg

Created: 1 July 2014



View of Boston, c. 1860; an early stereoscopic card for viewing a scene from nature

More details

Soule, John P., 1827-1904 -- Photographer - This image is available from the New York Public Library's Digital Library under the digital ID G90F336_113F: digitalgallery.nypl.org → digitalcollections.nypl.org

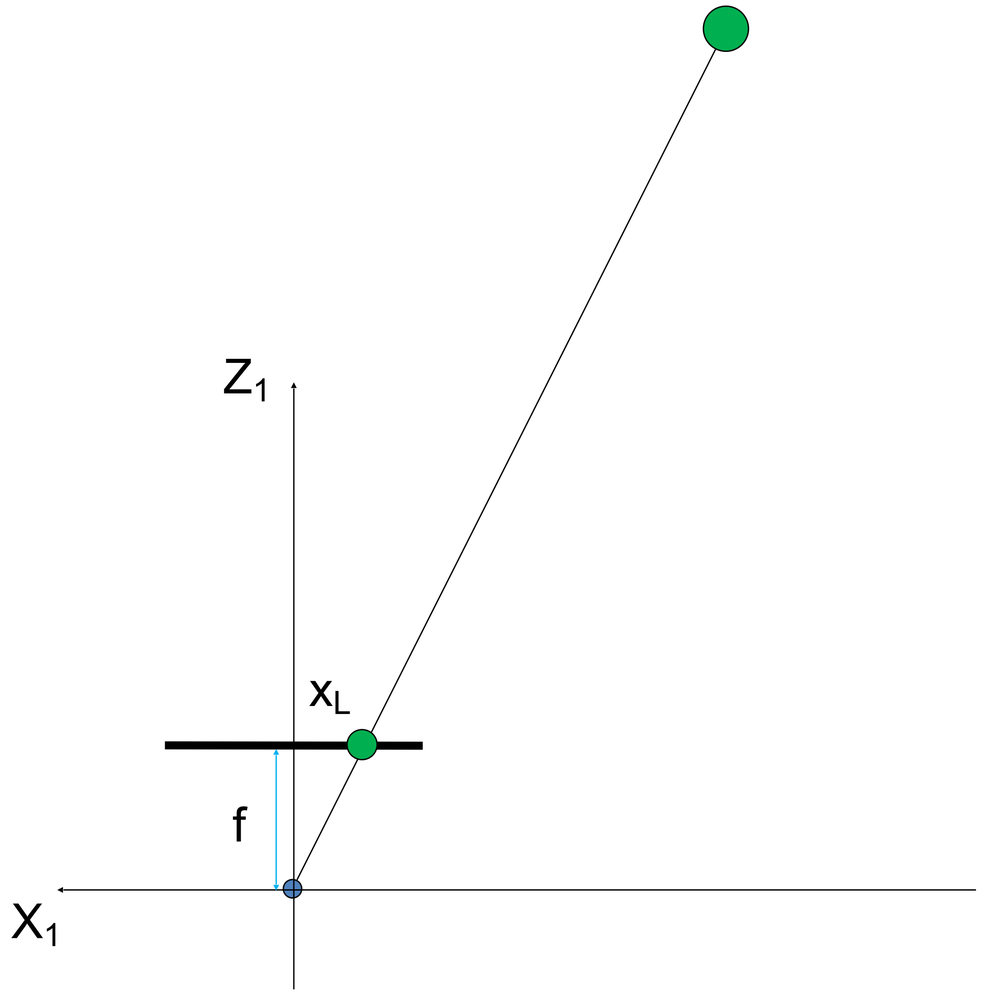
Public Domain

File: Charles Street Mall, Boston Common, by Soule, John P., 1827-1904 3.jpg

Created: Coverage: 1860?-1890?. Source Imprint: 1860?-1890?. Digital item published 7-28-2005; updated 4-23-2009.

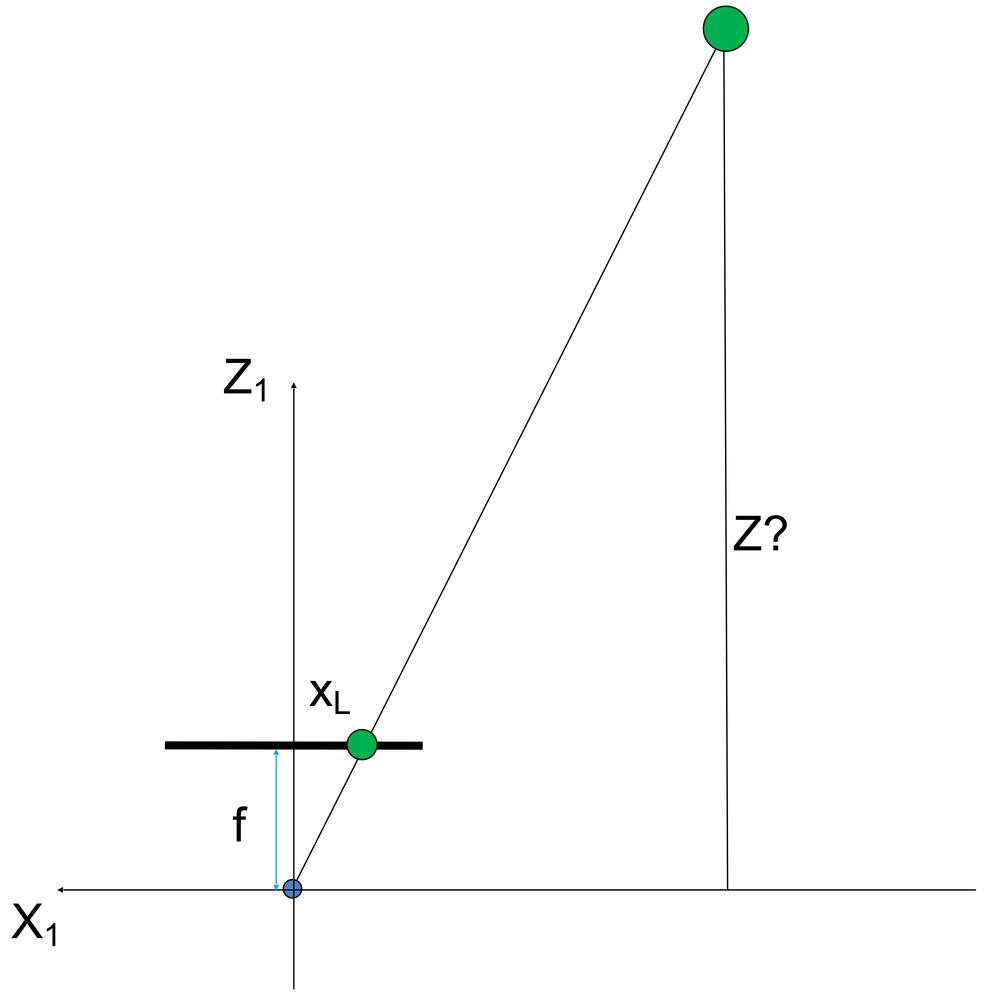
Source: Torralba, Isola, Freeman

Geometry for a simple stereo system



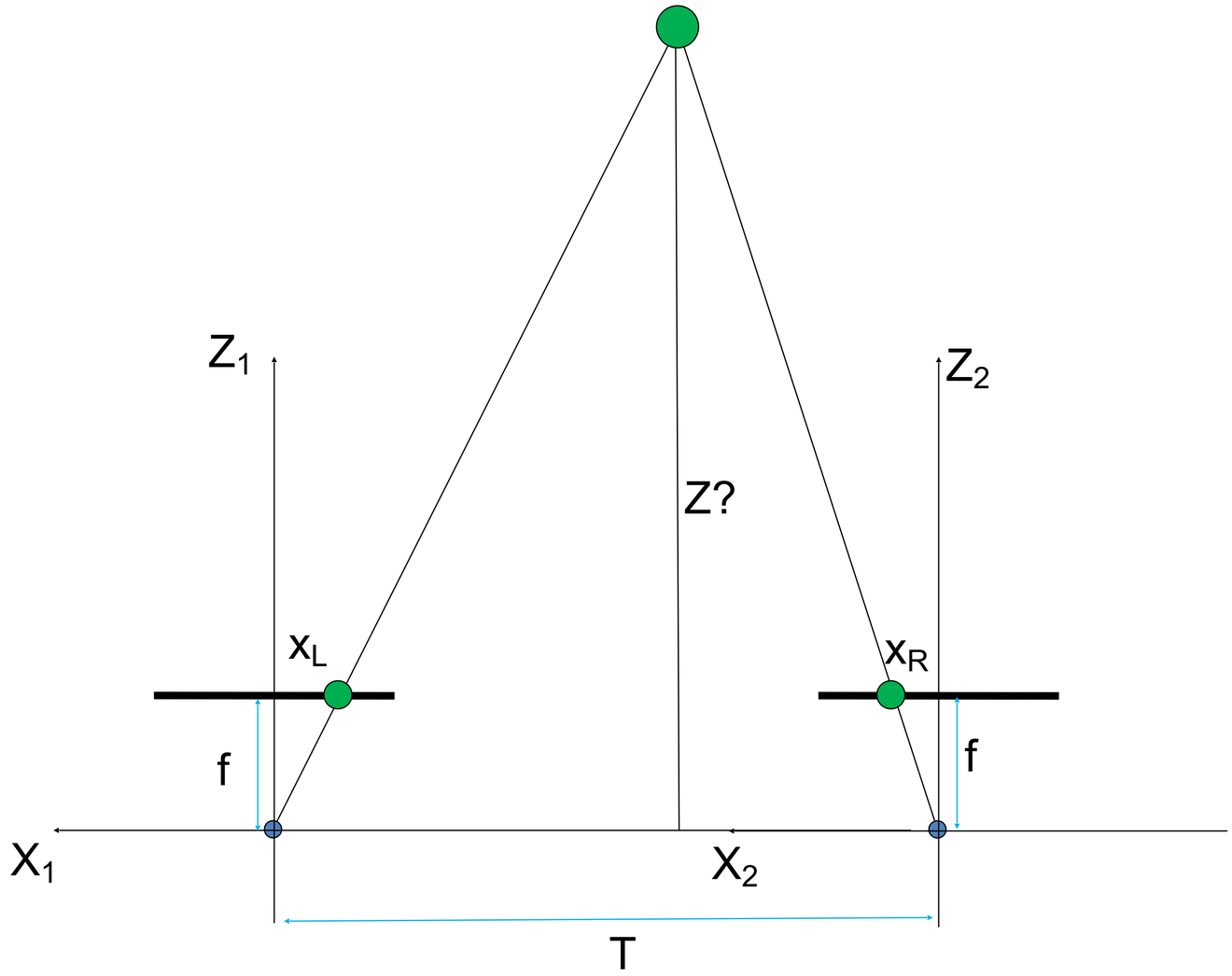
Source: Torralba, Isola, Freeman

Geometry for a simple stereo system



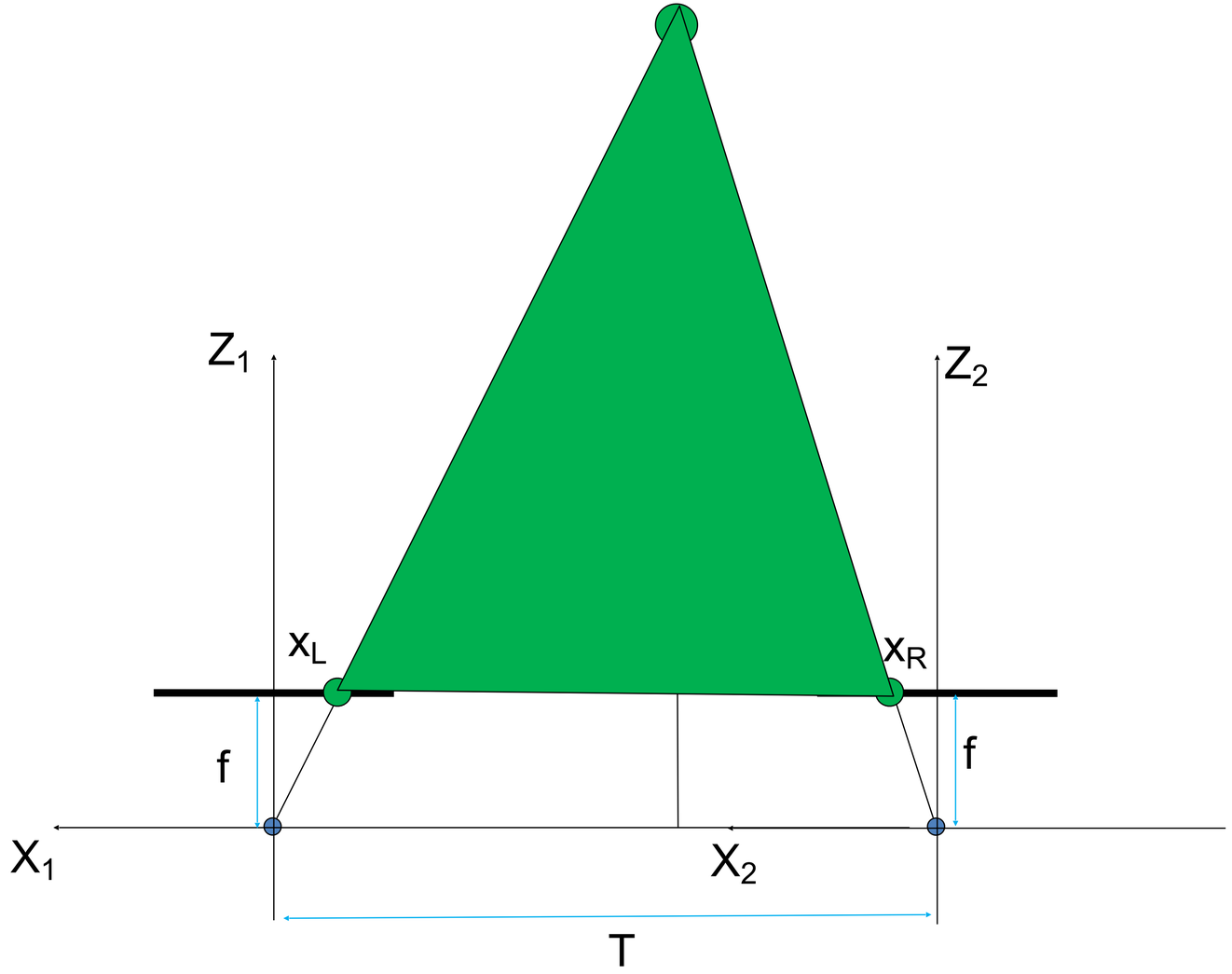
Source: Torralba, Isola, Freeman

Geometry for a simple stereo system



Source: Torralba, Isola, Freeman

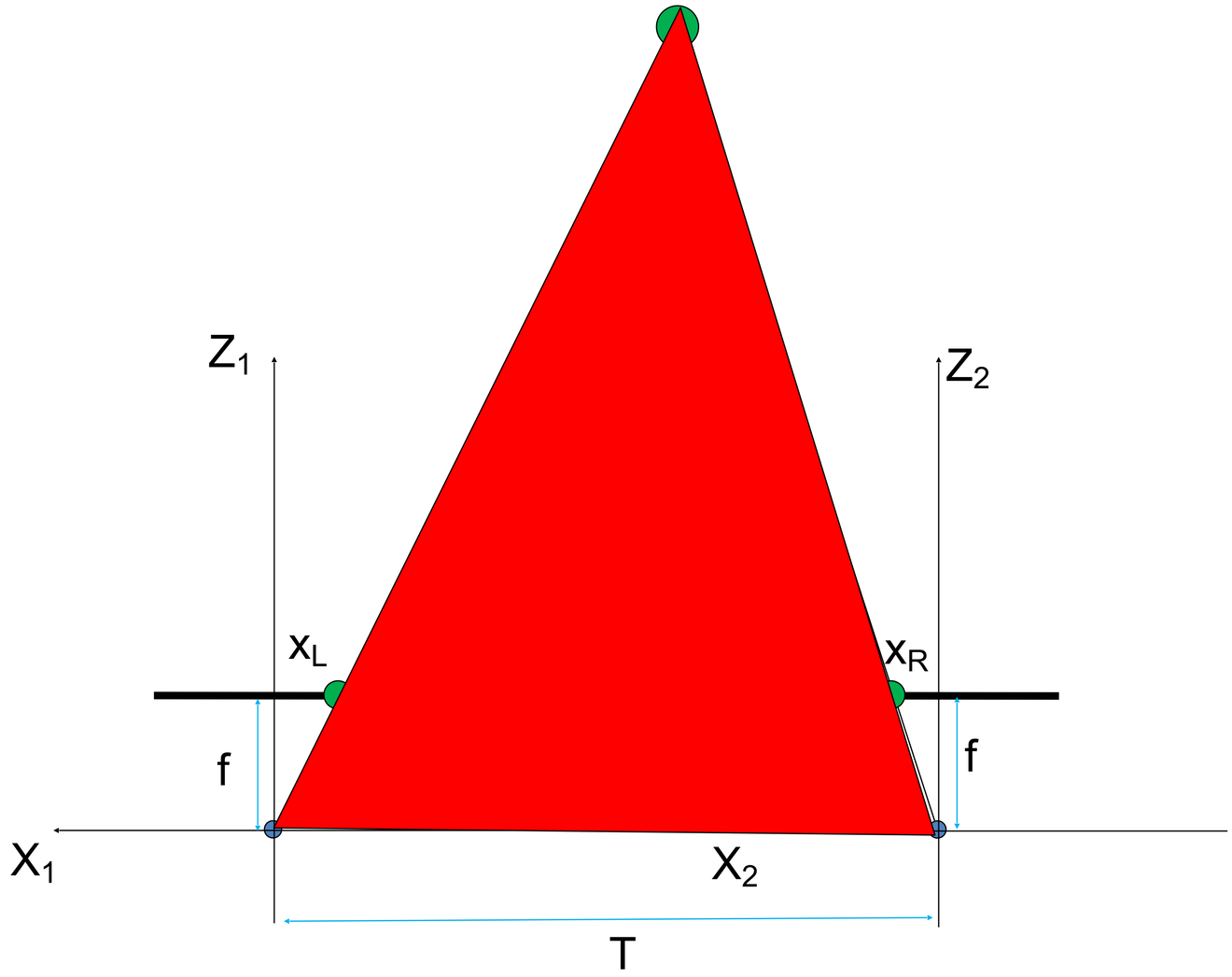
Geometry for a simple stereo system



Similar triangles

Source: Torralba, Isola, Freeman

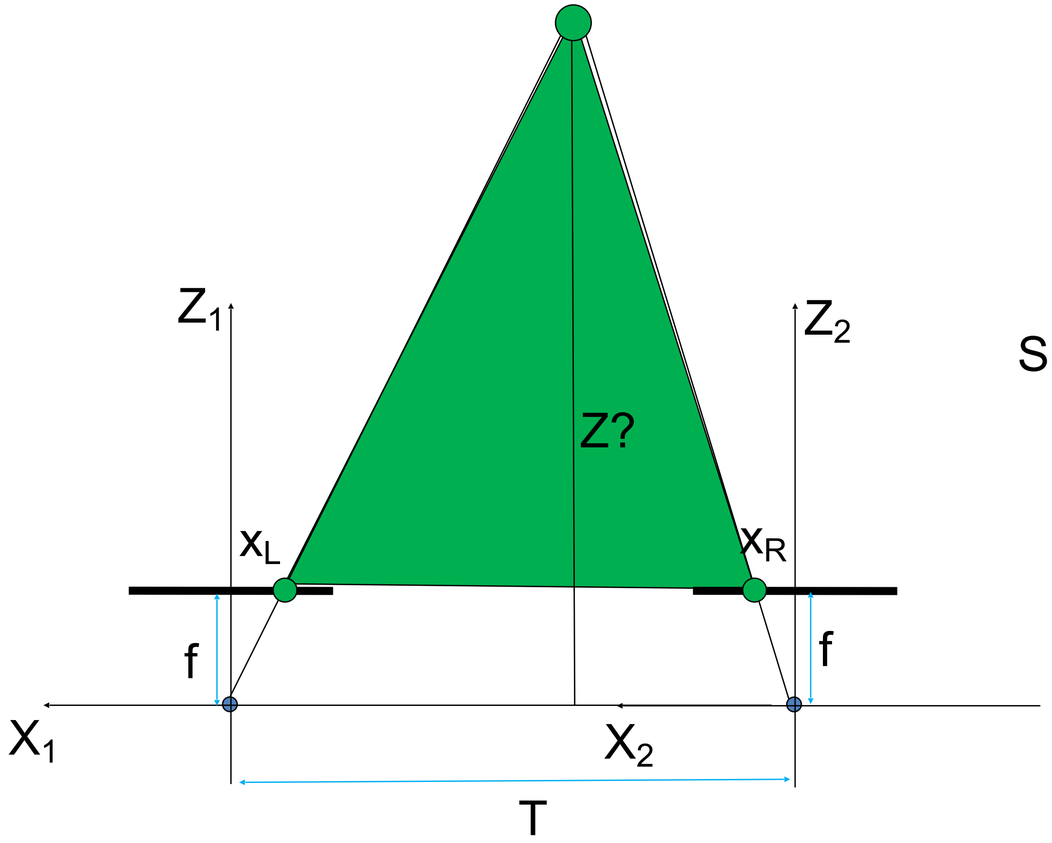
Geometry for a simple stereo system



Similar triangles

Source: Torralba, Isola, Freeman

Geometry for a simple stereo system

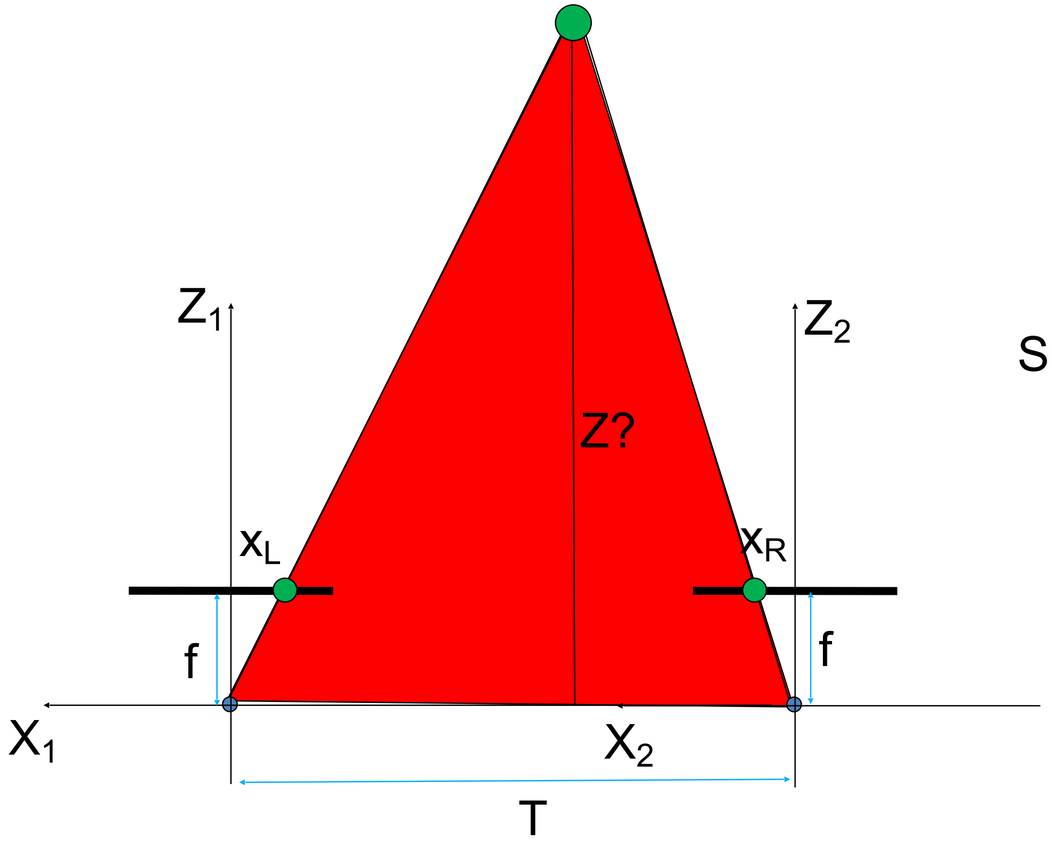


Similar triangles:

$$\frac{T+X_L-X_R}{Z-f} =$$

Source: Torralba, Isola, Freeman

Geometry for a simple stereo system

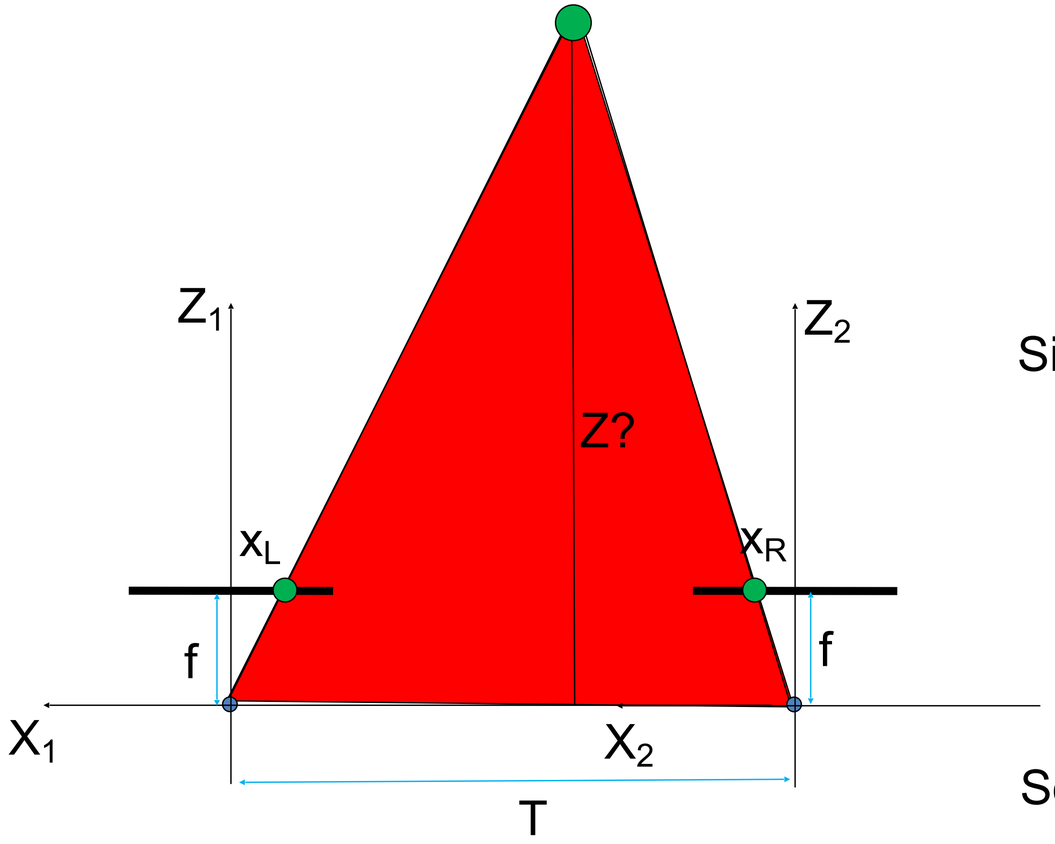


Similar triangles:

$$\frac{T+X_L-X_R}{Z-f} = \frac{T}{Z}$$

Source: Torralba, Isola, Freeman

Geometry for a simple stereo system



Similar triangles:

$$\frac{T+X_L-X_R}{Z-f} = \frac{T}{Z}$$

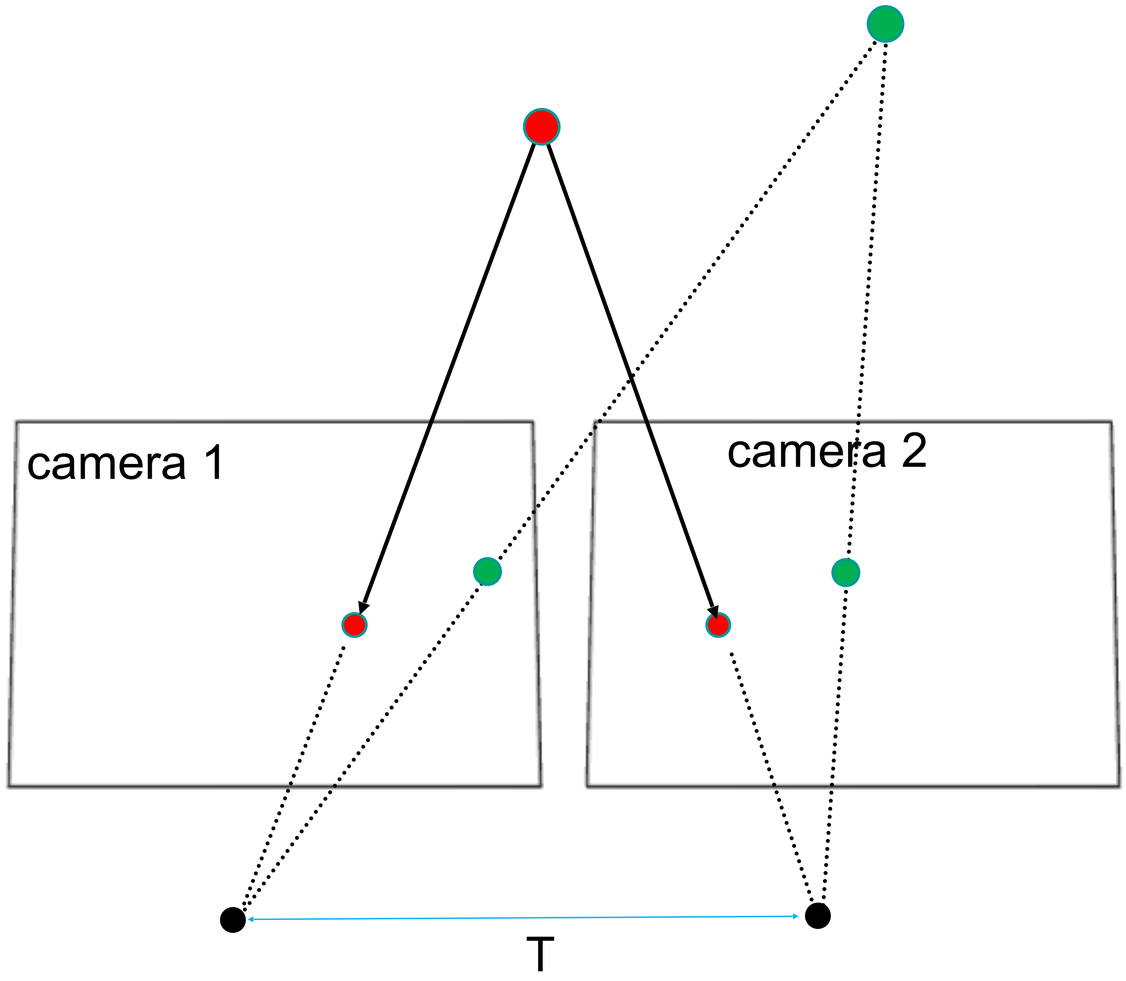
Solving for Z:

$$Z = f\,\frac{T}{X_L - X_R}$$

Disparity

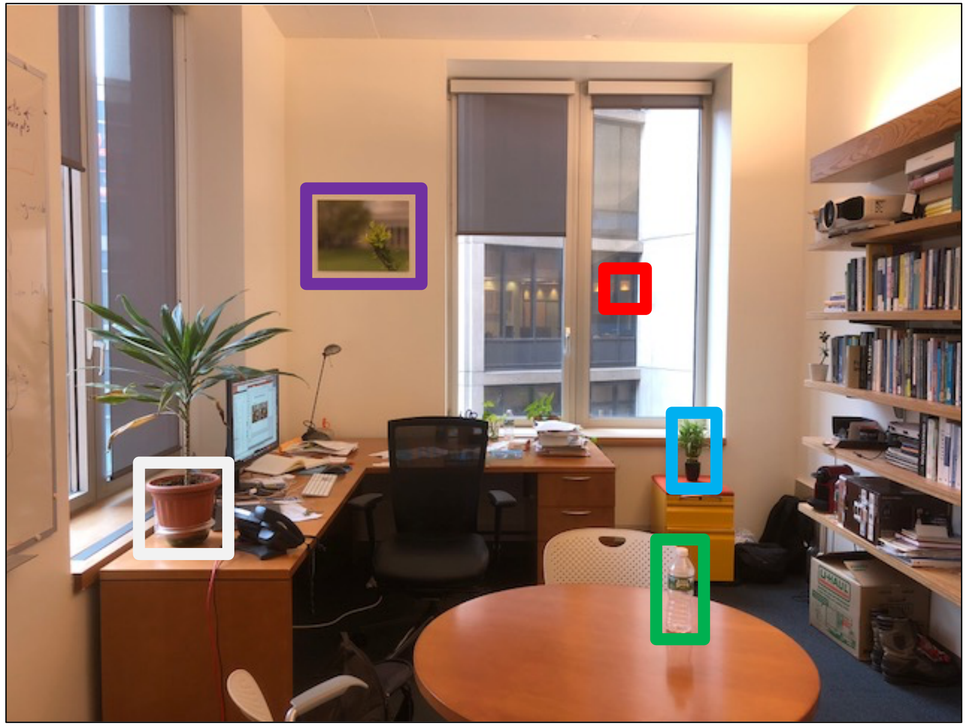
Source: Torralba, Isola, Freeman

In 3D



Source: Torralba, Isola, Freeman

Disparity map



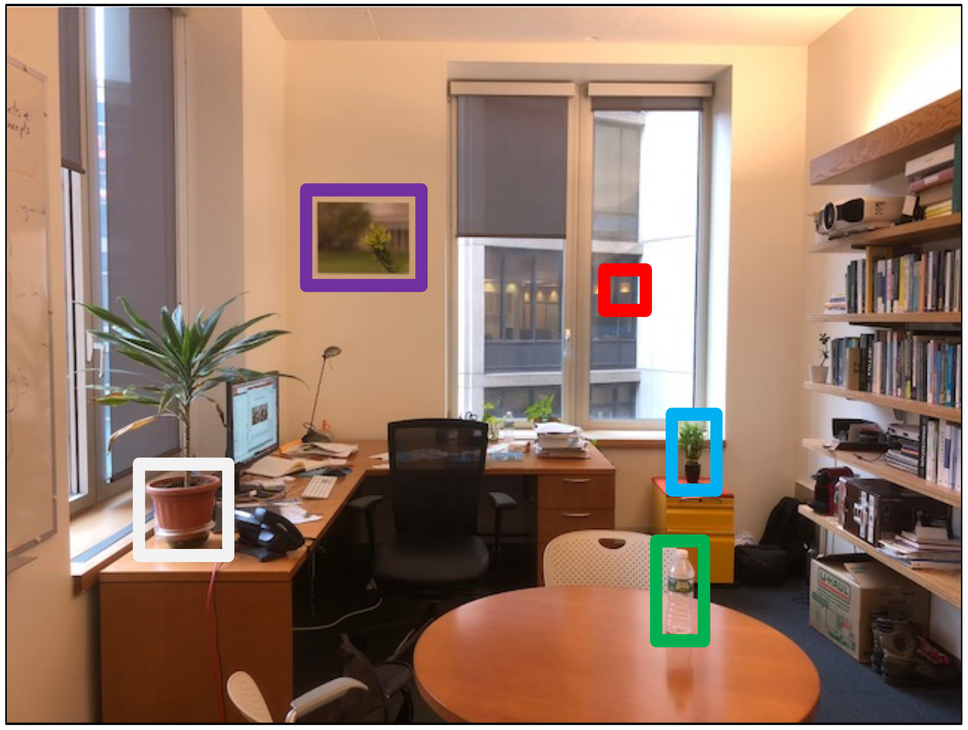
Left image

Right image

Second picture is ~1m to the right

Source: Torralba, Isola, Freeman

Disparity map

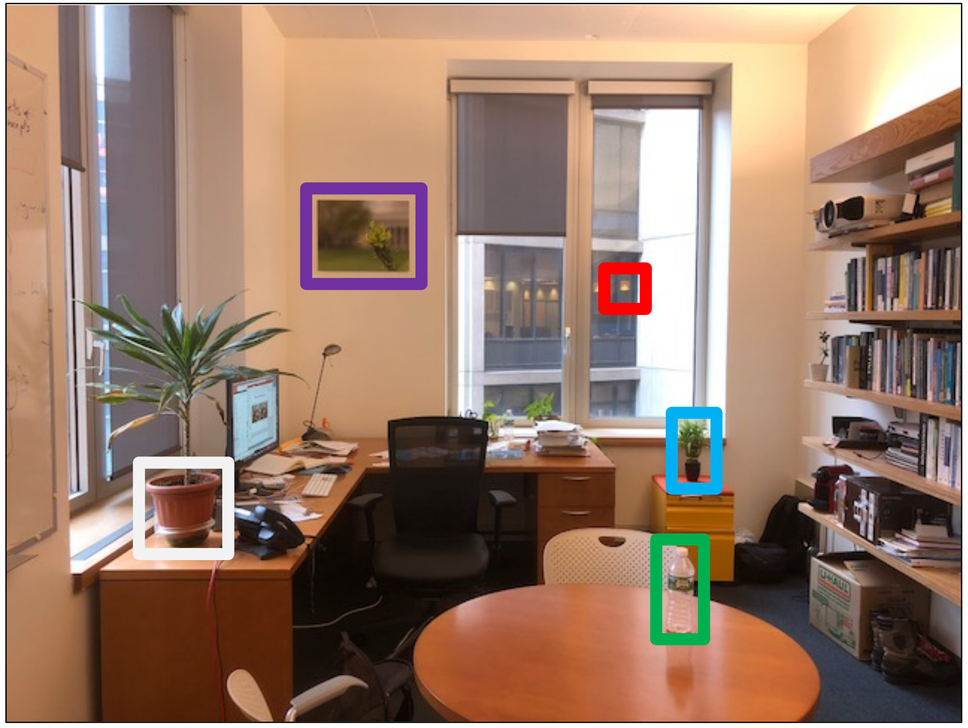


Left image

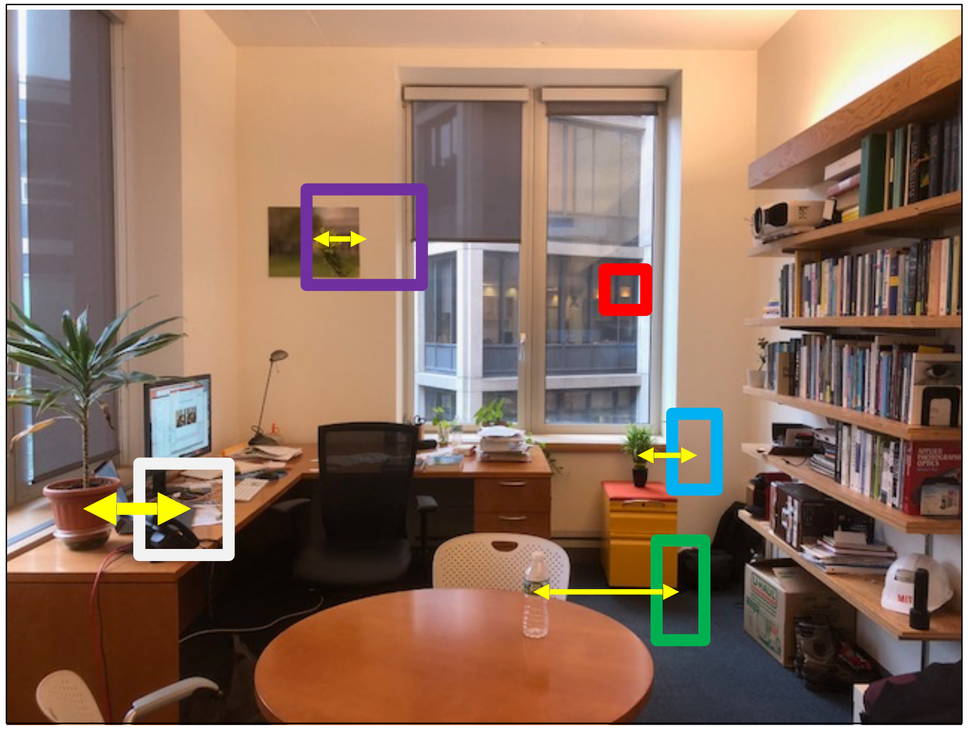
Right image

Source: Torralba, Isola, Freeman

Disparity map



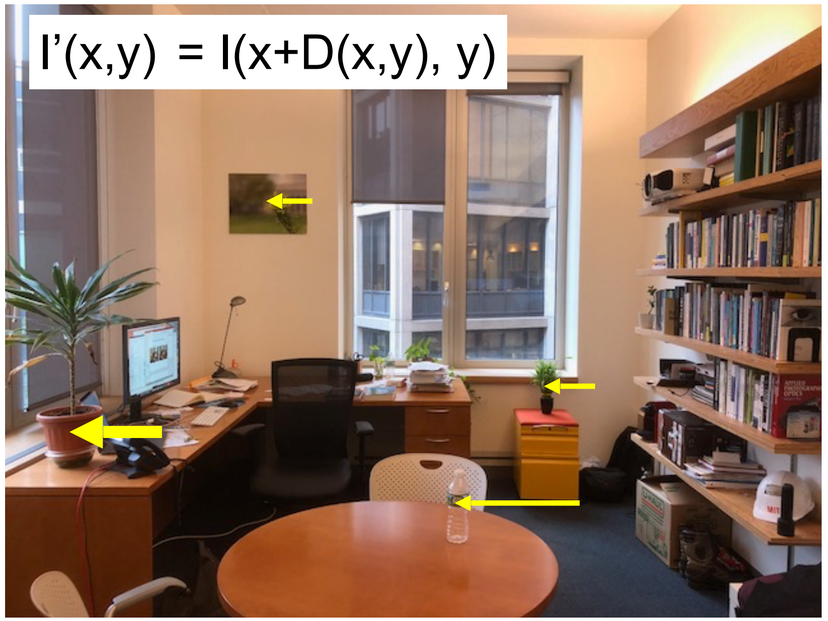
Left image



Right image

Source: Torralba, Isola, Freeman

Disparity map



D(x,y)

$$Z(x,y) = \frac{a}{D(x,y)}$$

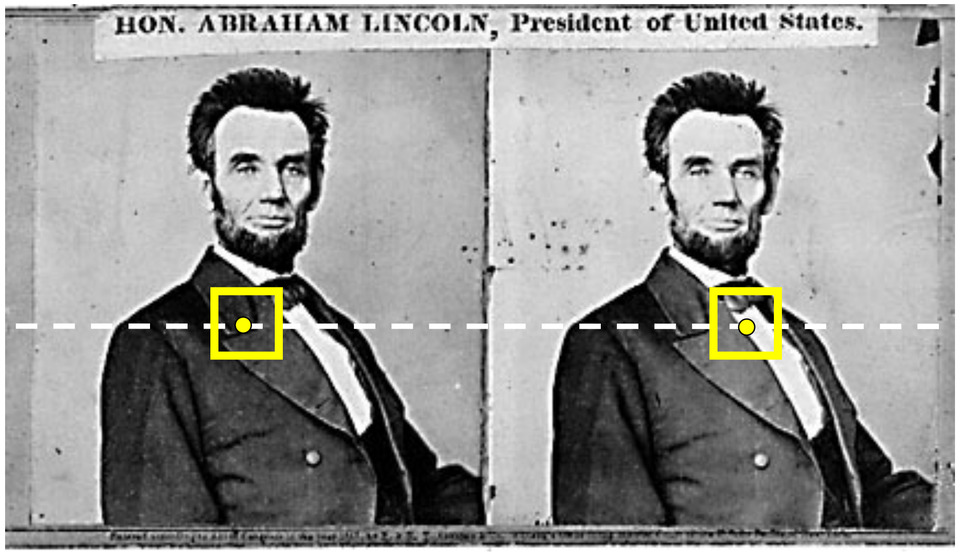
Source: Torralba, Isola, Freeman

Finding correspondences

We only need to search for matches along horizontal lines.

Source: Torralba, Isola, Freeman

Basic stereo algorithm



For each “epipolar line”

For each pixel in the left image

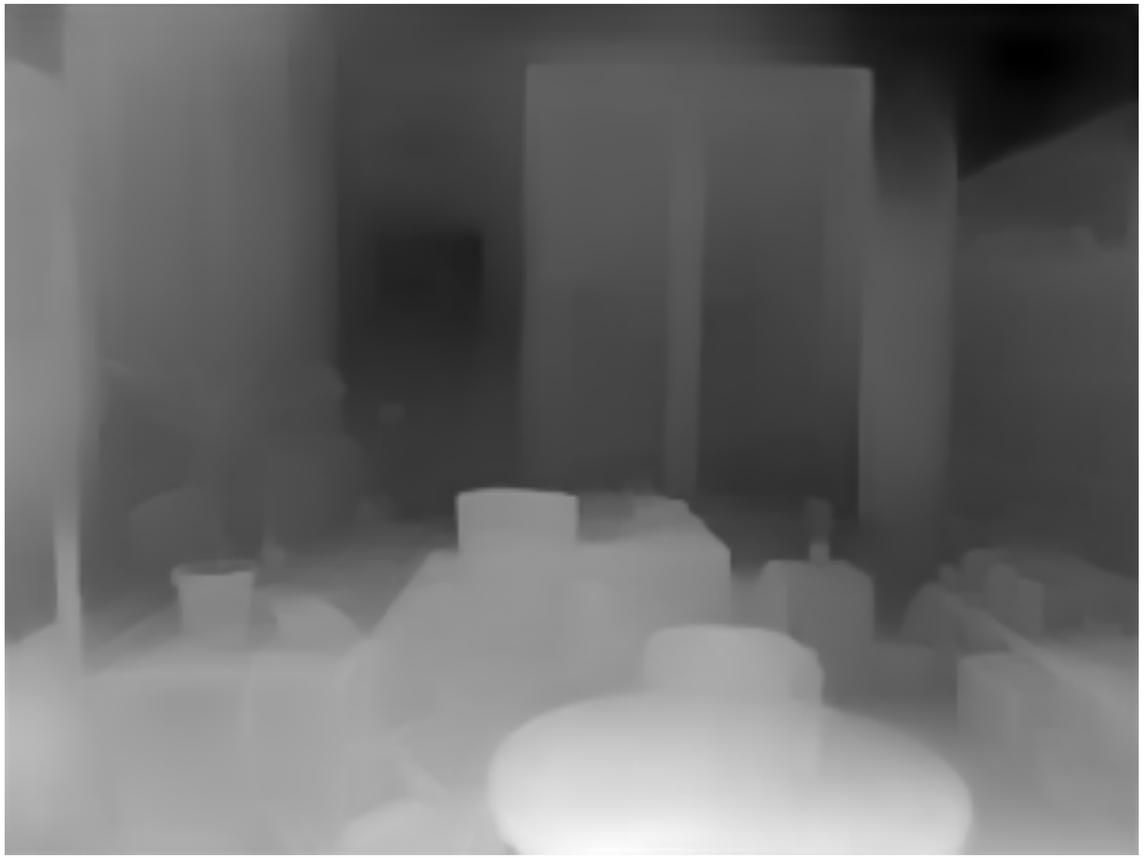
- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

Source: R. Szeliski

Computing disparity

Source: Torralba, Isola, Freeman

But you can learn depth from a single image



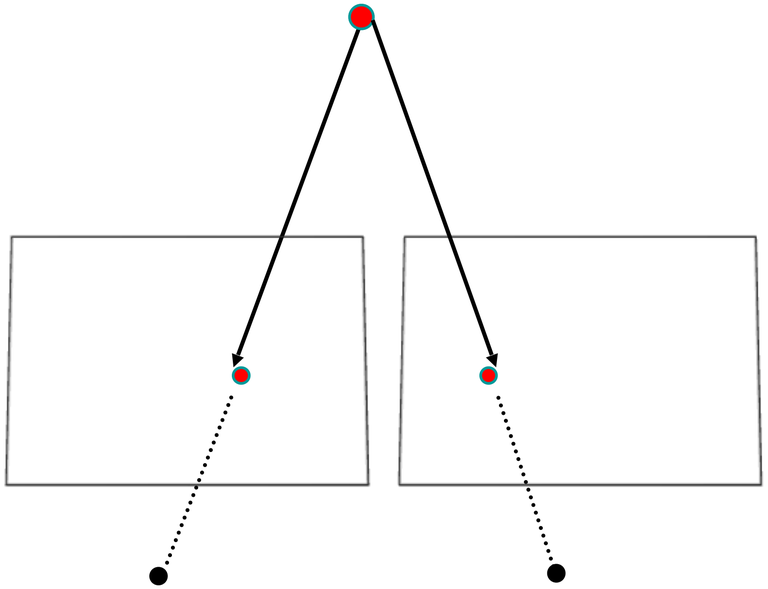
MegaDepth: Learning Single-View Depth Prediction from Internet Photos

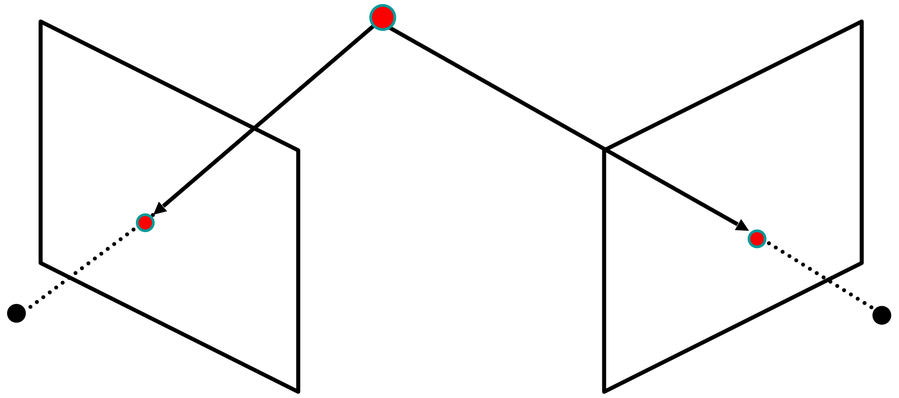
Zhengqi Li Noah Snavely
Department of Computer Science & Cornell Tech, Cornell University

Source: Torralba, Isola, Freeman

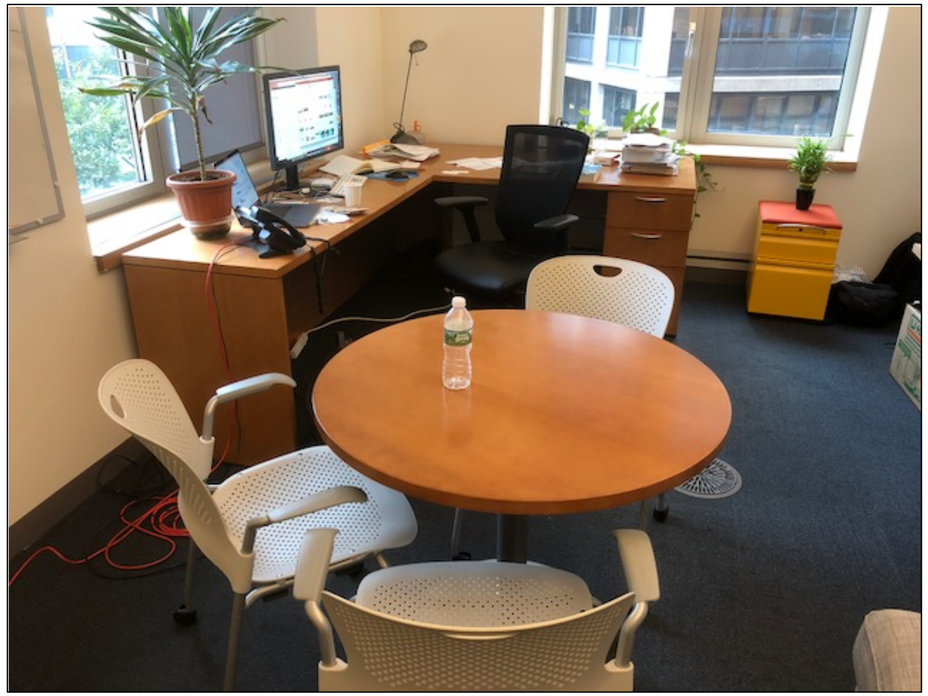
General case

- The two cameras need not have parallel optical axes.

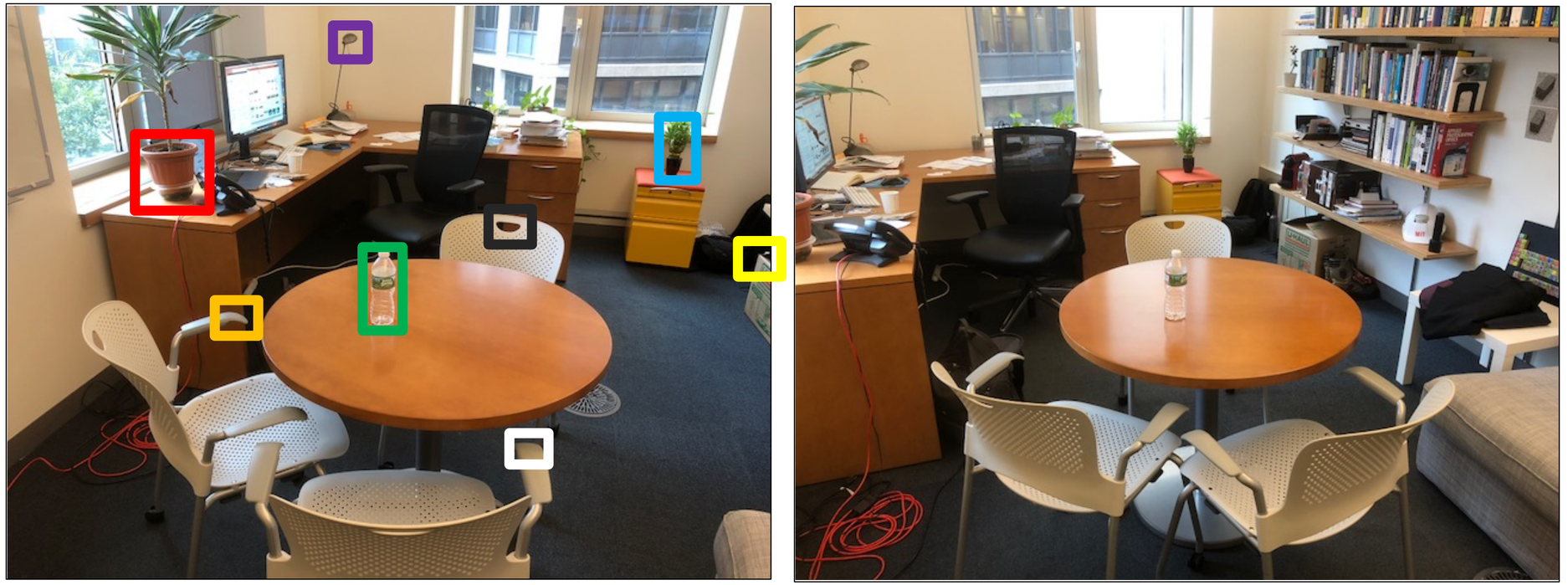




Source: Torralba, Isola, Freeman

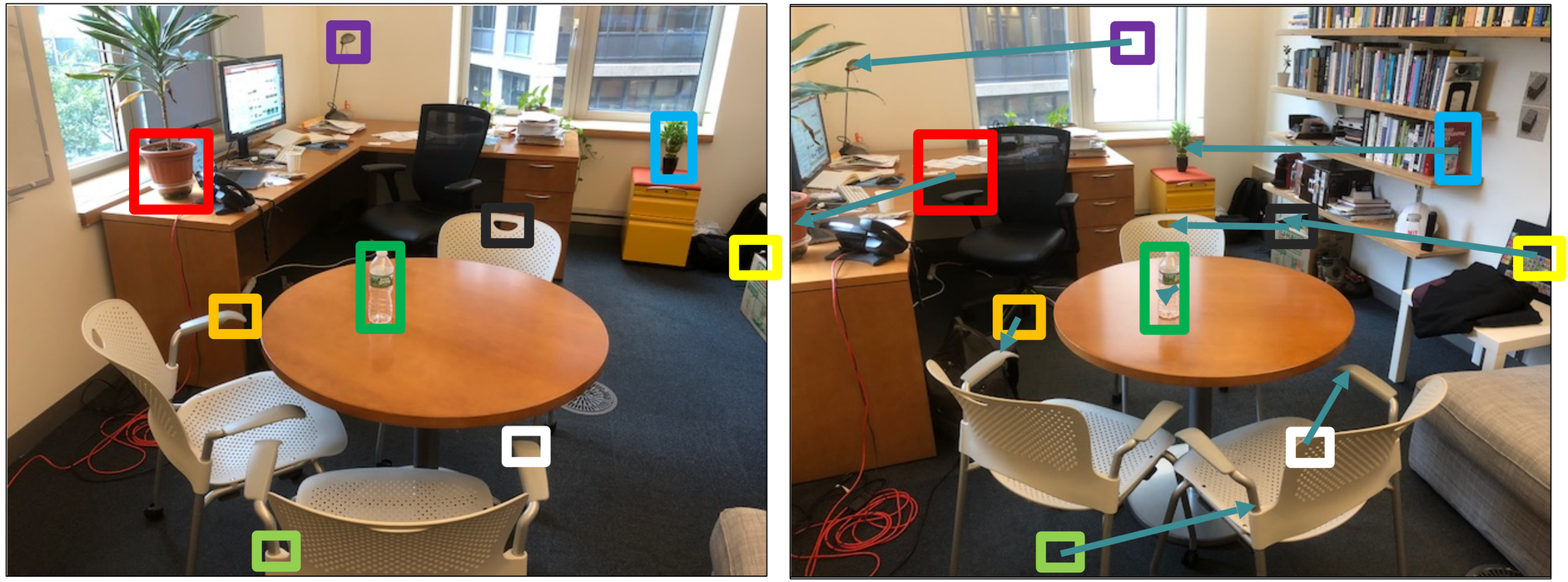


Source: Torralba, Isola, Freeman



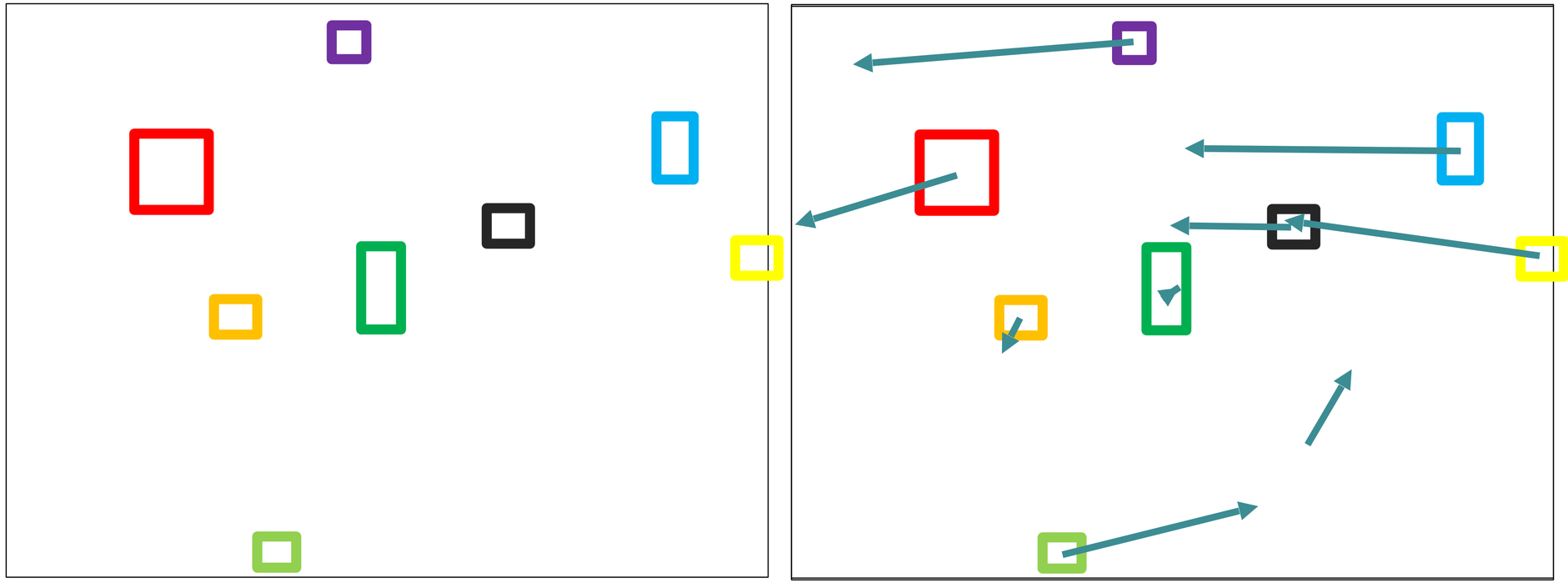
Do we need to search for matches only along horizontal lines?

Source: Torralba, Isola, Freeman



Do we need to search for matches only along horizontal lines?

Source: Torralba, Isola, Freeman

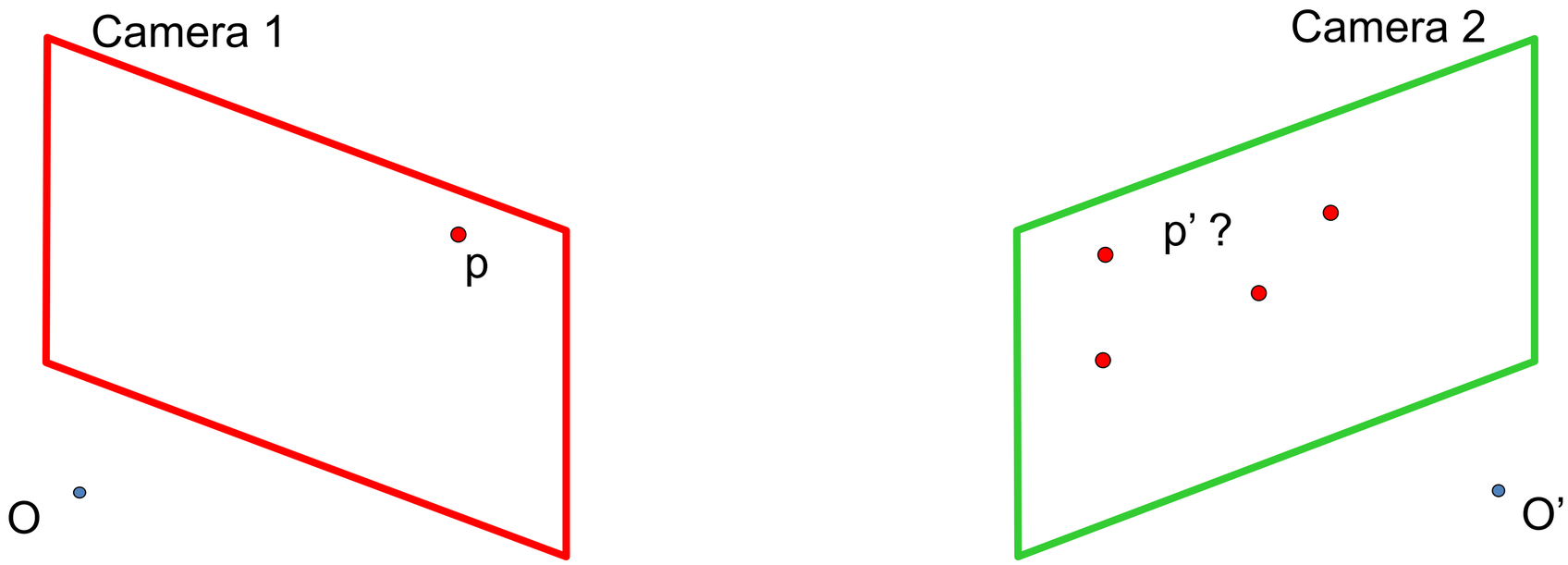


~~Do we need to search for matches only along horizontal lines?~~

It looks like we need to search everywhere... are there any constraints that can guide the search?

Source: Torralba, Isola, Freeman

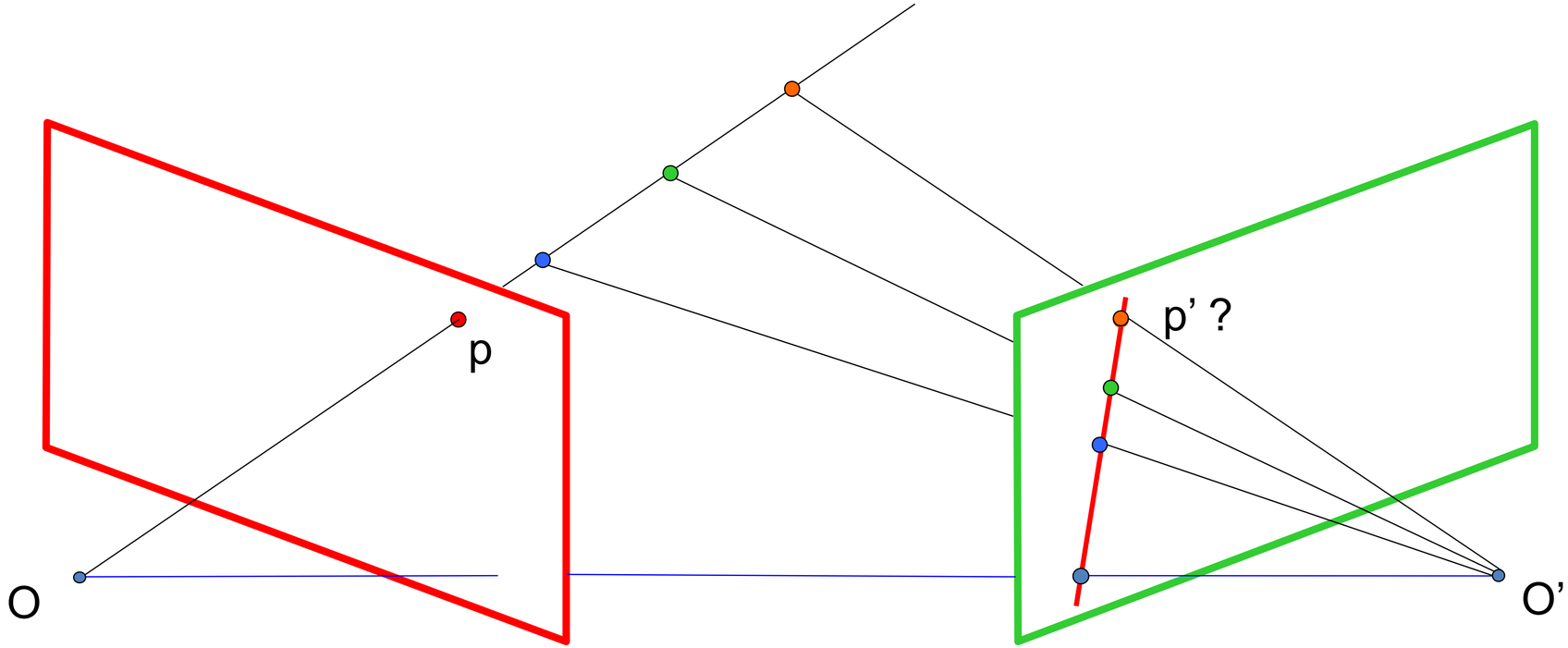
Stereo correspondence constraints



If we see a point in camera 1, are there any constraints on where we will find it on camera 2?

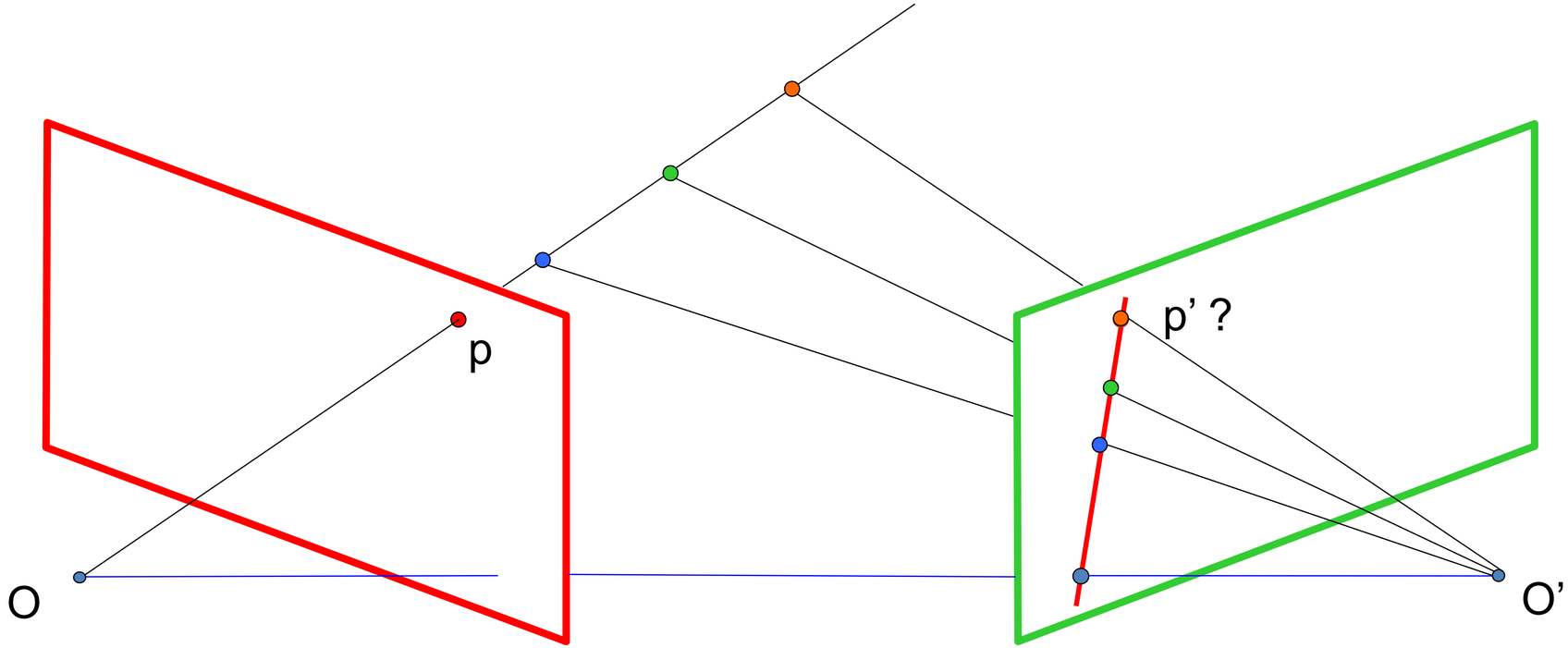
Source: Torralba, Isola, Freeman

Stereo correspondence constraints



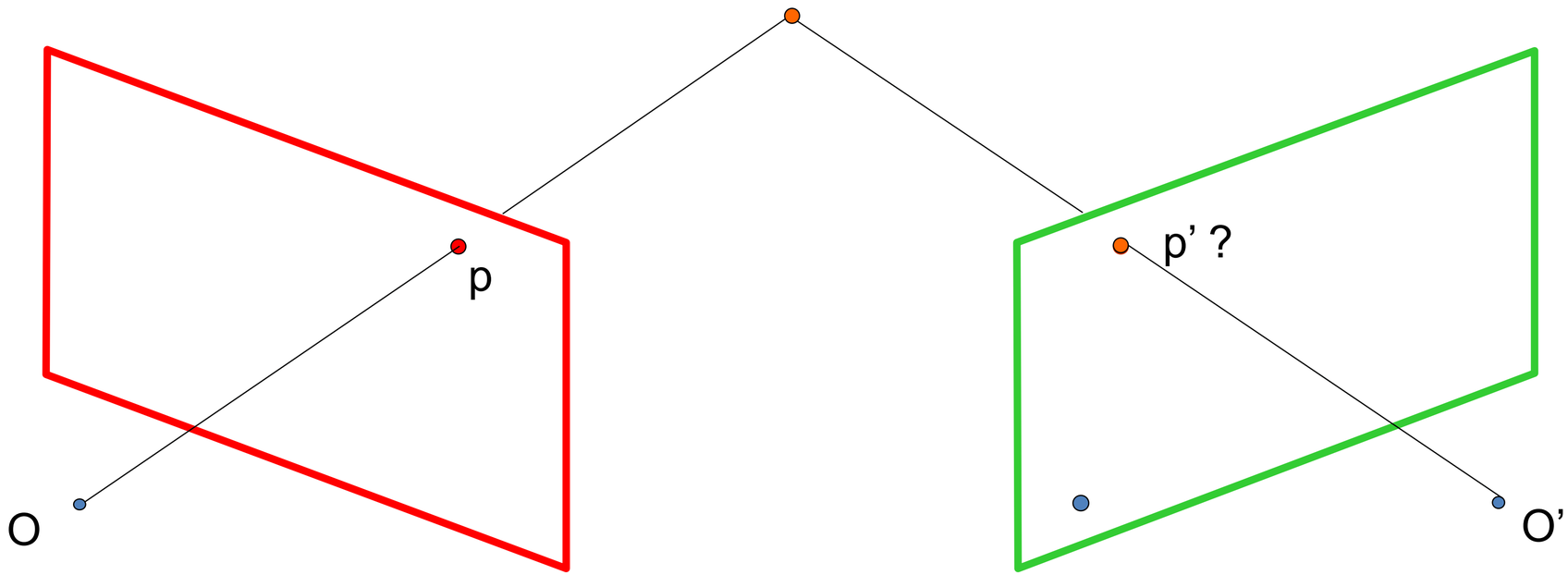
Source: Torralba, Isola, Freeman

Epipolar constraint



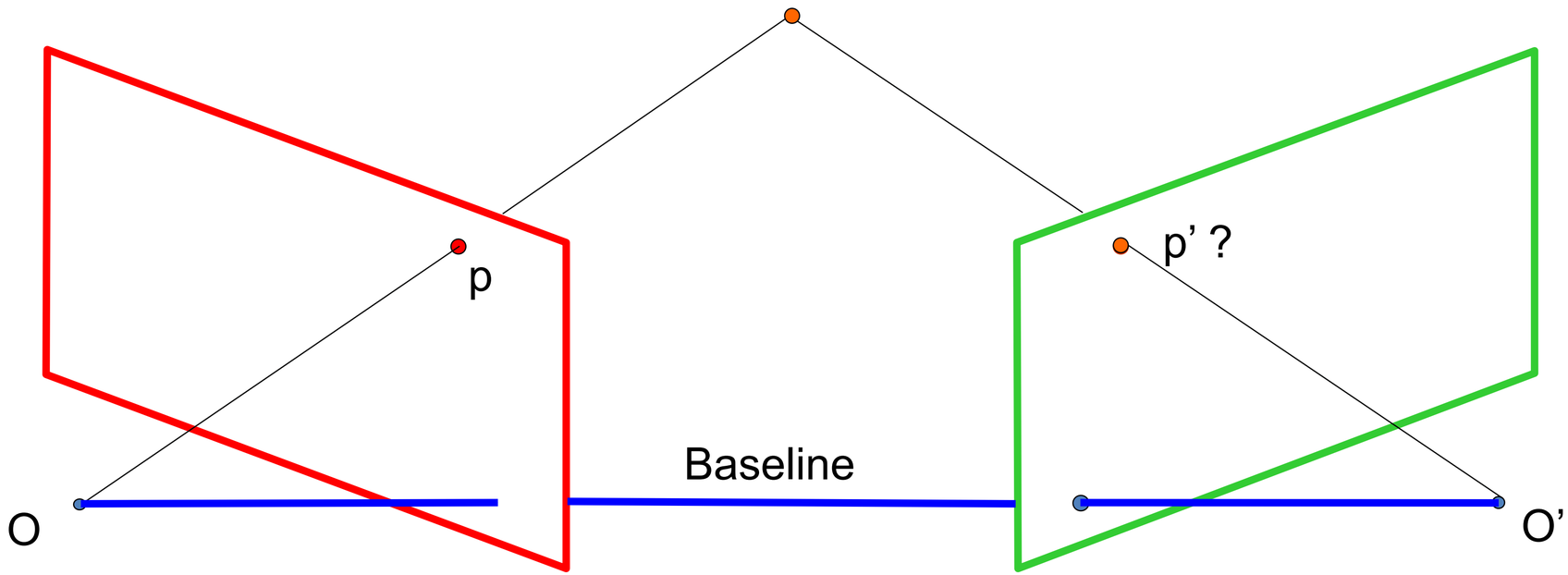
Source: Torralba, Isola, Freeman

Some terminology



Source: Torralba, Isola, Freeman

Some terminology

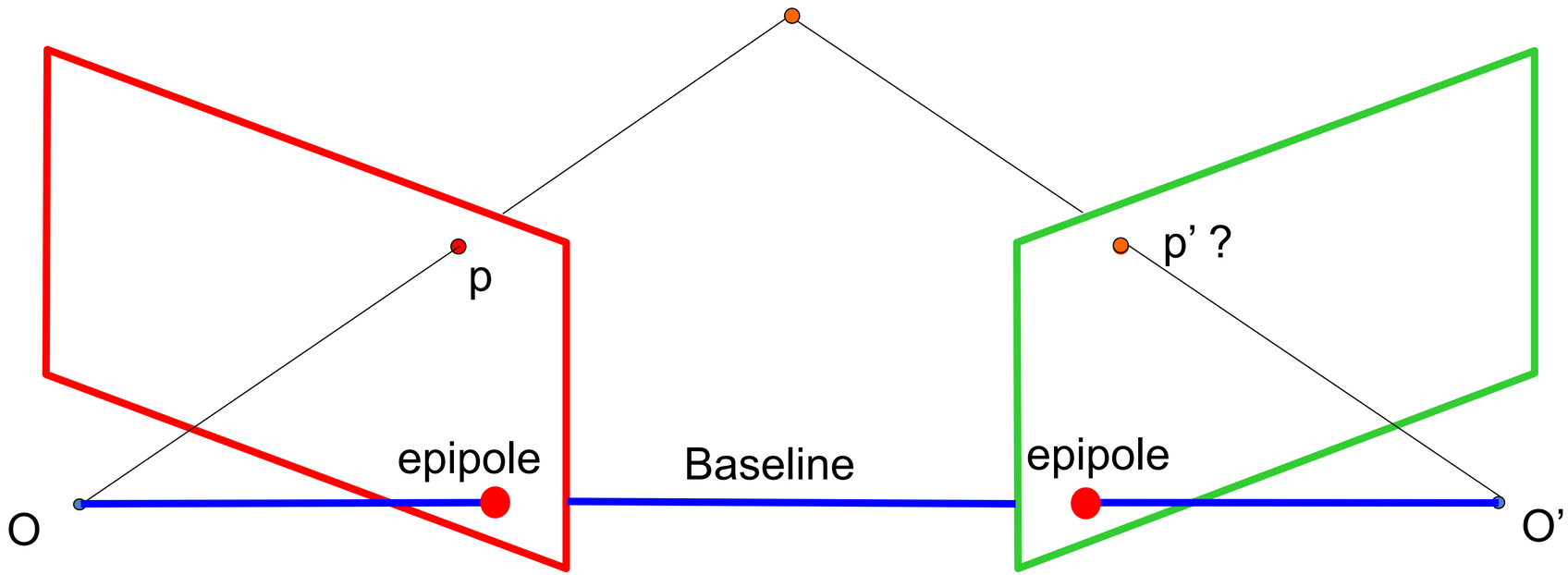


Baseline: the line connecting the two camera centers

Epipole: point of intersection of *baseline* with the image plane

Source: Torralba, Isola, Freeman

Some terminology

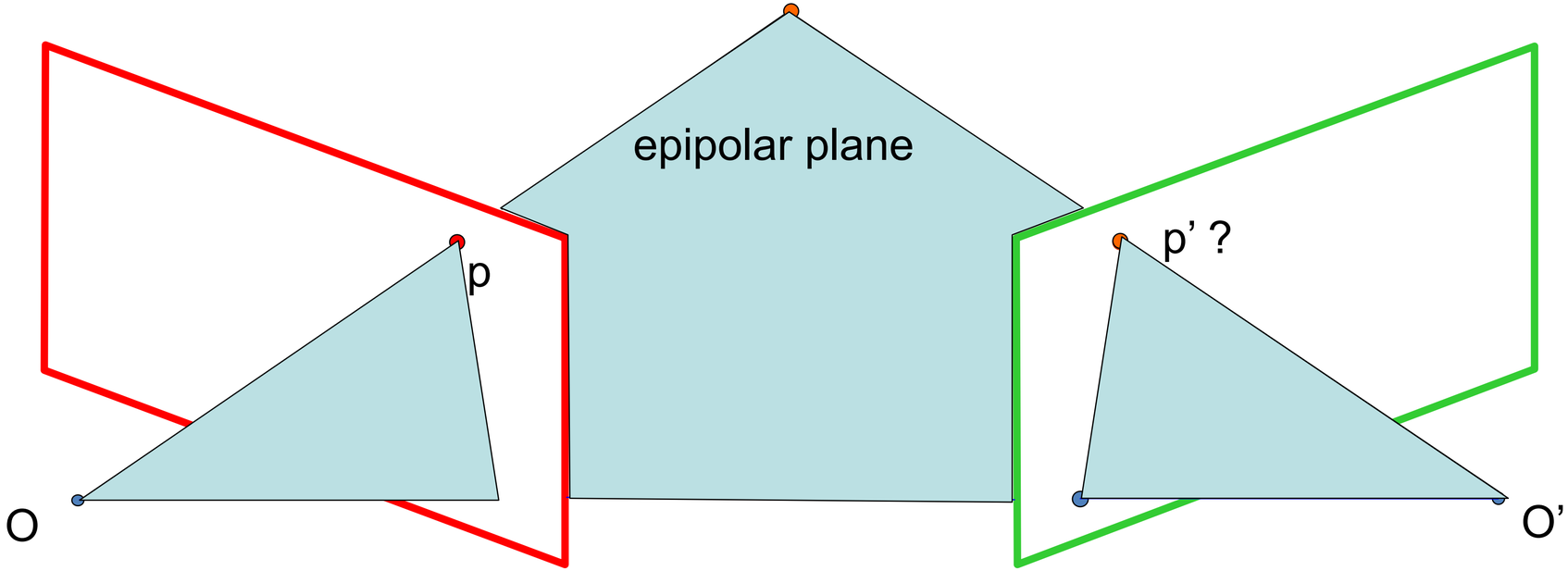


Baseline: the line connecting the two camera centers

Epipole: point of intersection of *baseline* with the image plane

Source: Torralba, Isola, Freeman

Some terminology



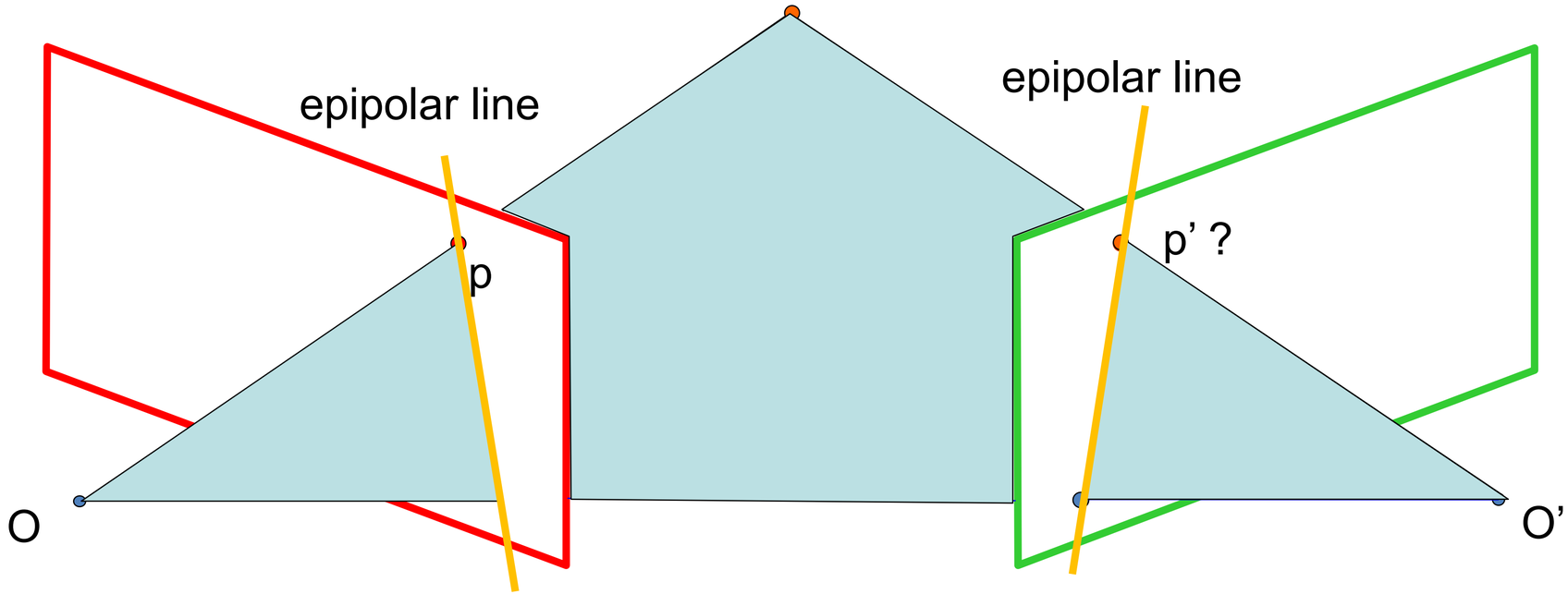
Baseline: the line connecting the two camera centers

Epipole: point of intersection of *baseline* with the image plane

Epipolar plane: the plane that contains the two camera centers and a 3D point in the world

Source: Torralba, Isola, Freeman

Some terminology



Baseline: the line connecting the two camera centers

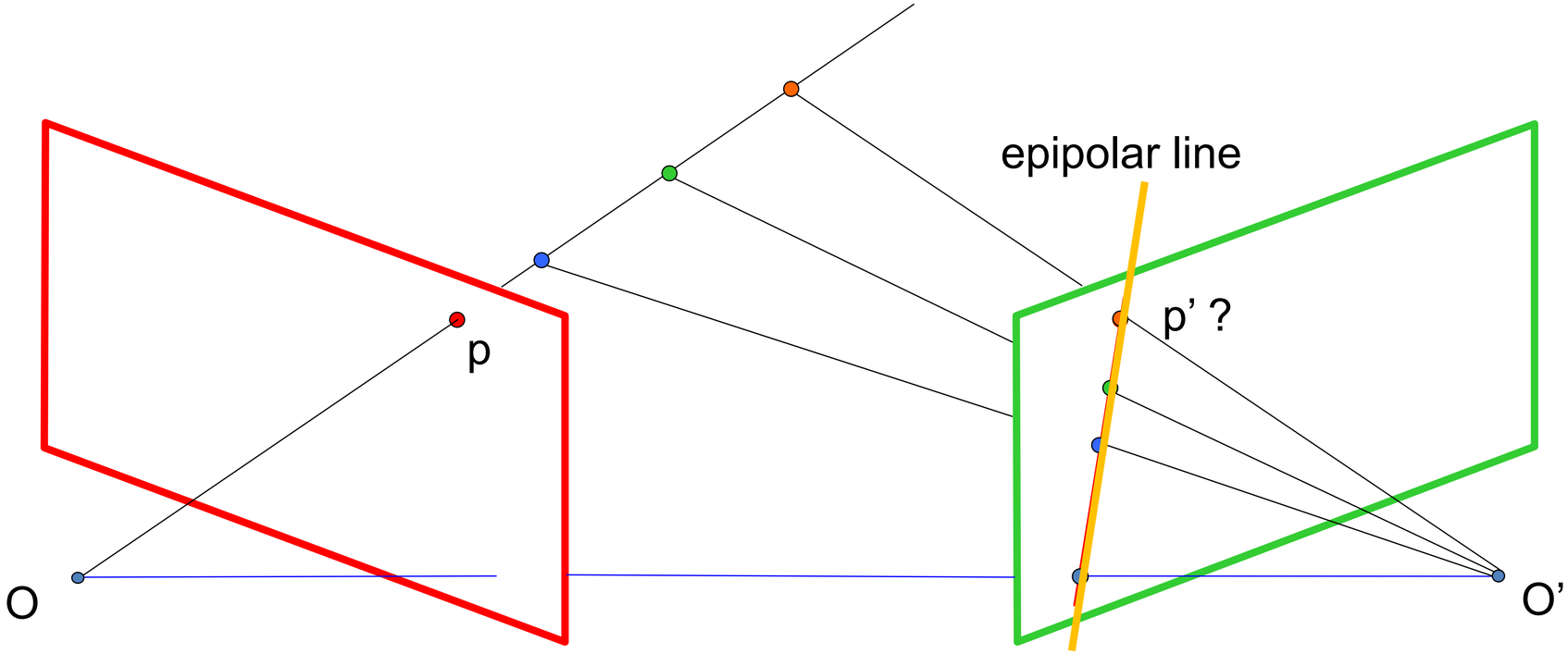
Epipole: point of intersection of *baseline* with the image plane

Epipolar plane: the plane that contains the two camera centers and a 3D point in the world

Epipolar line: intersection of the *epipolar plane* with each image plane

Source: Torralba, Isola, Freeman

Epipolar constraint

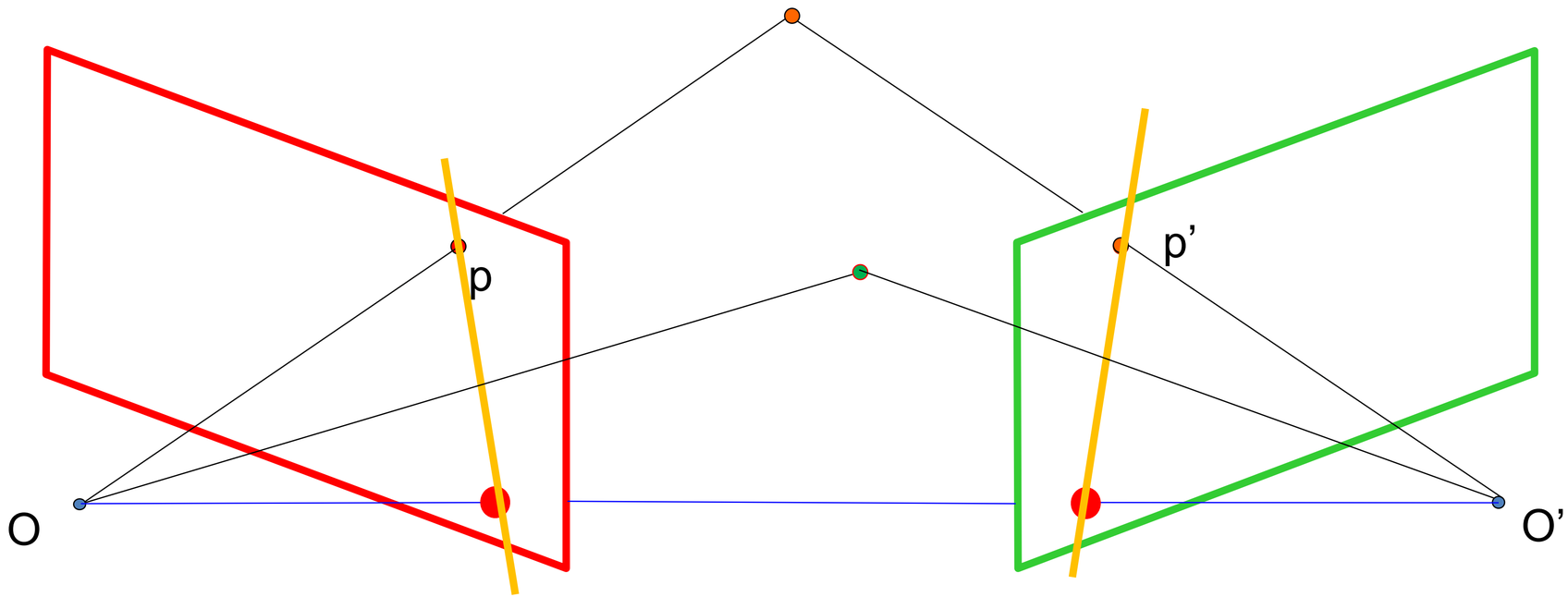


We can search for matches across epipolar lines

All epipolar lines intersect at the epipoles

Source: Torralba, Isola, Freeman

The fundamental matrix



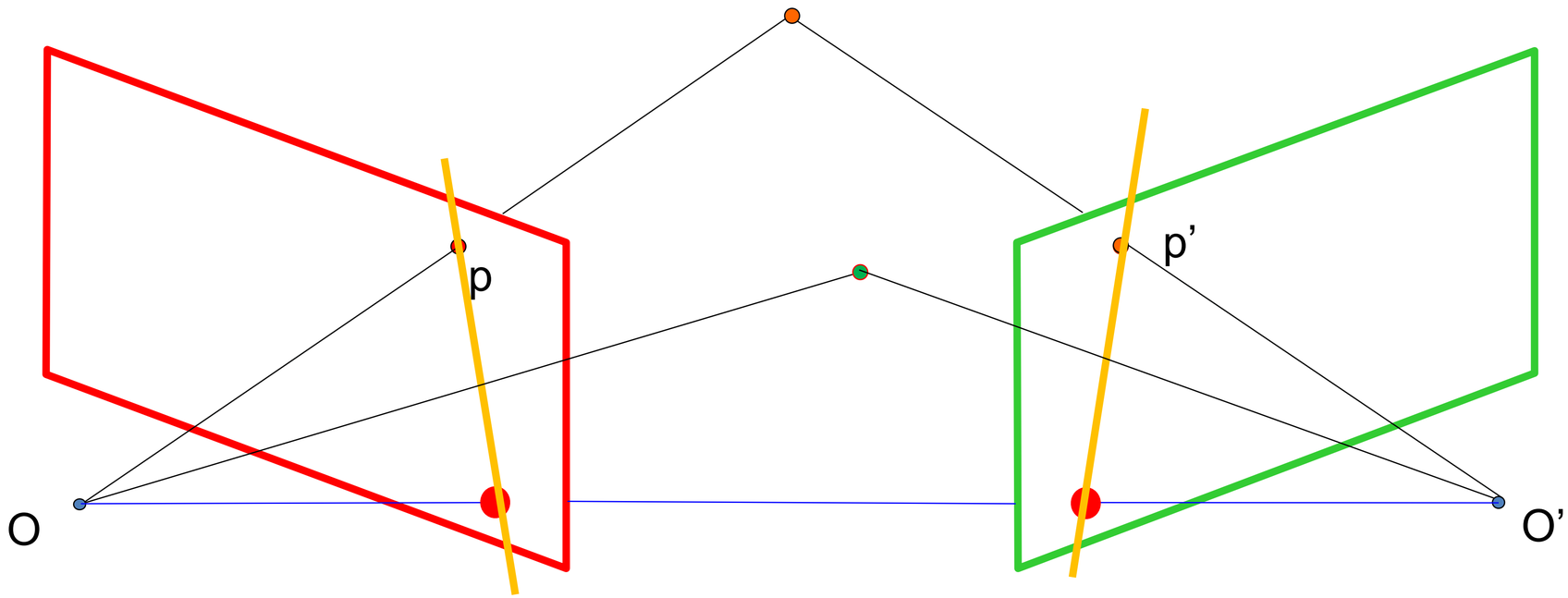
If we observe a point in one image, its position in the other image is constrained to lie on line defined by above.

$$p^T F p' = 0$$

F: fundamental matrix
p, p’: image points in homogeneous coordinates

Source: Torralba, Isola, Freeman

The fundamental matrix



$$(p^T F)\, p' = 0$$

$$u^T p' = 0$$

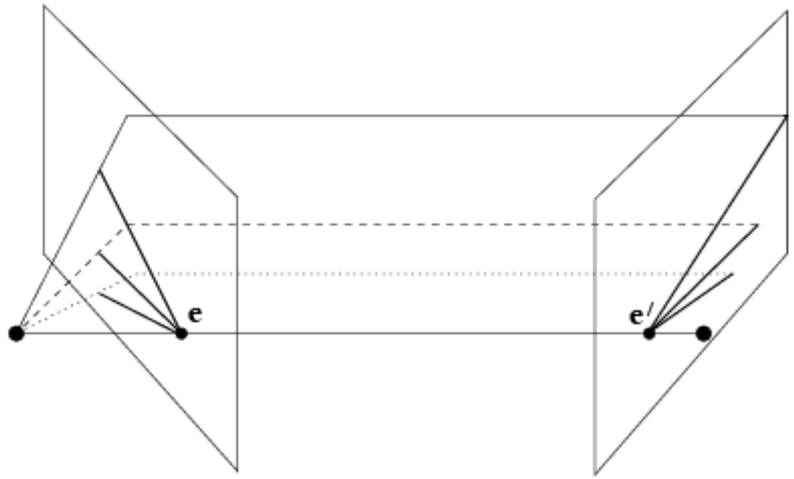
u: a line induced by p
p, p’: image points in homogeneous coordinates

Closely related to projection matrix:

$$\mathbf{F} = \mathbf{K}'^{-\mathsf{T}} [\mathbf{t}]_\times \mathbf{R} \mathbf{K}^{-1}$$

K, K’: intrinsics matrices
R, t: relative pose
See Hartley and Zisserman for derivation

Example: converging cameras



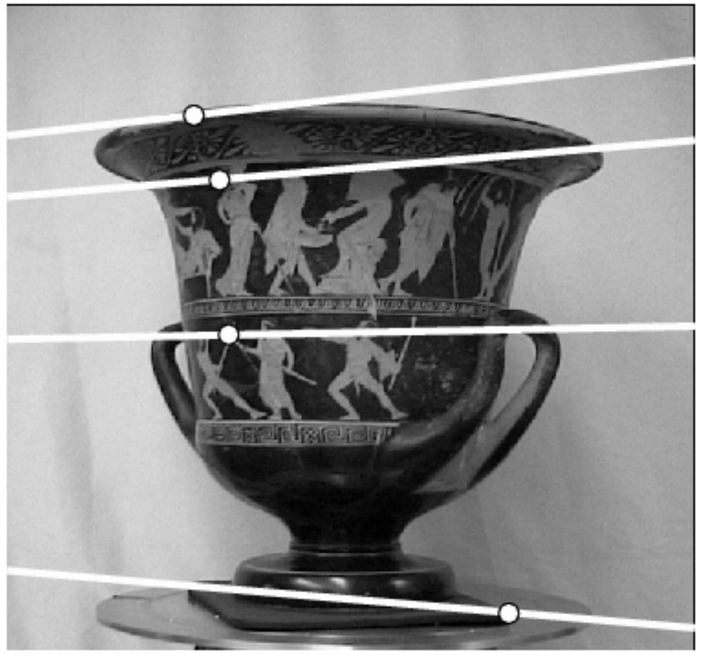
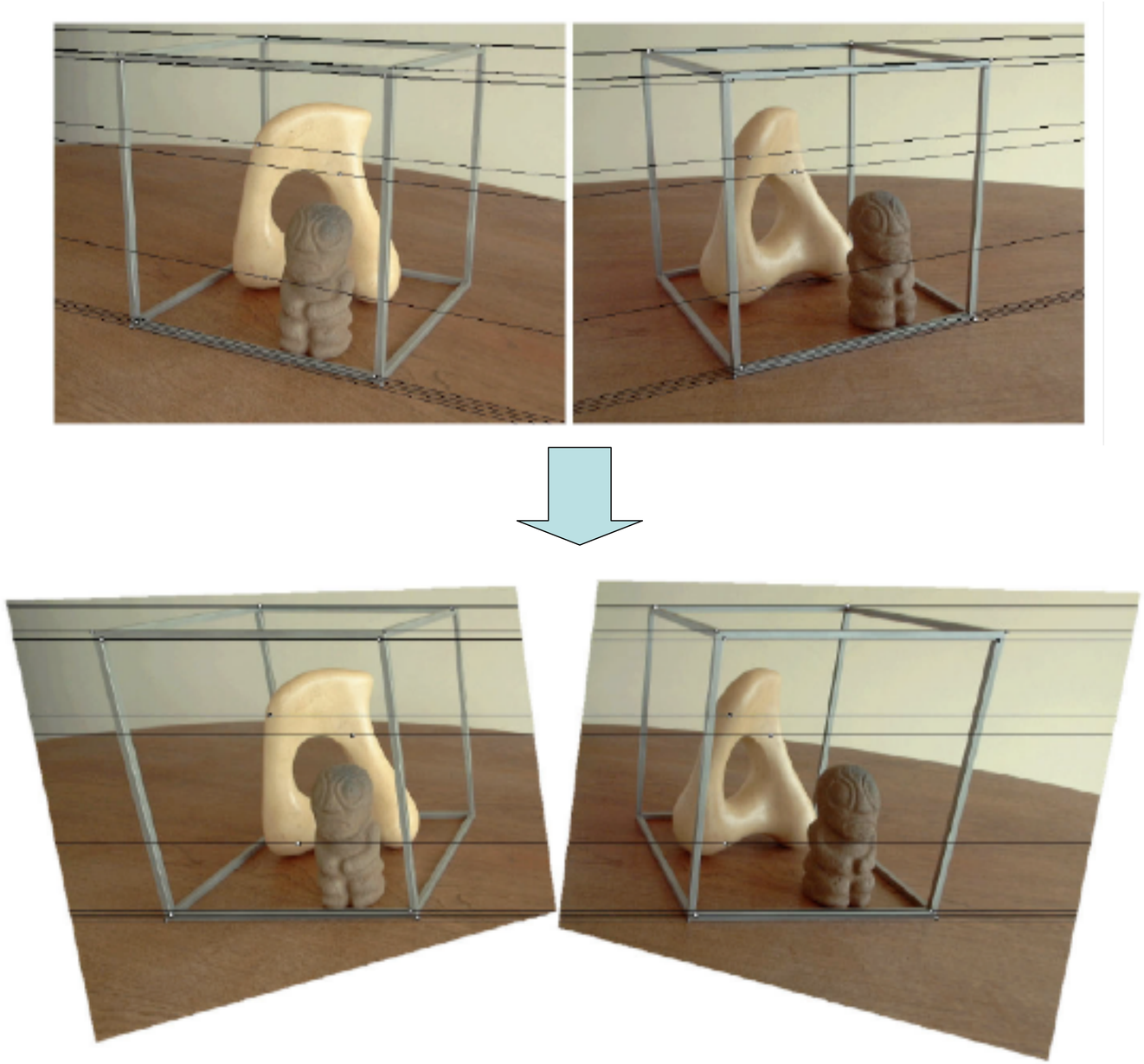


Figure from Hartley & Zisserman

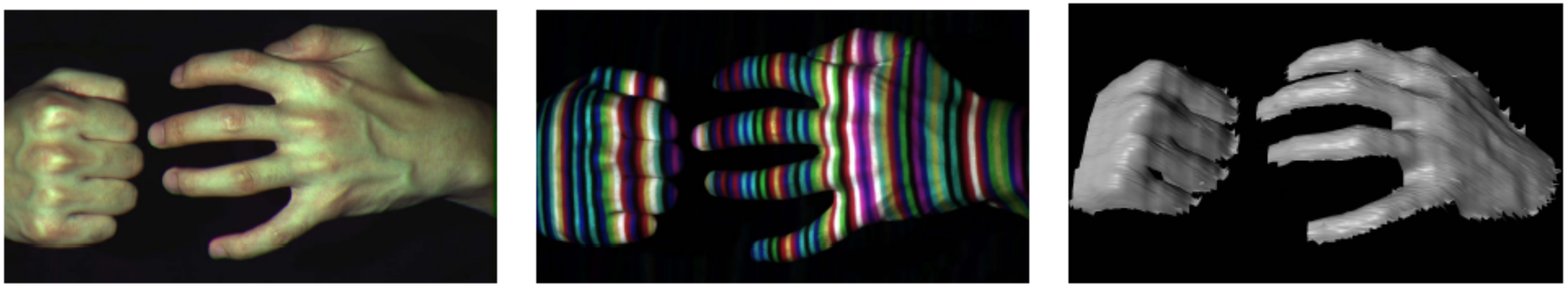
Source: Kristen Grauman

Image rectification

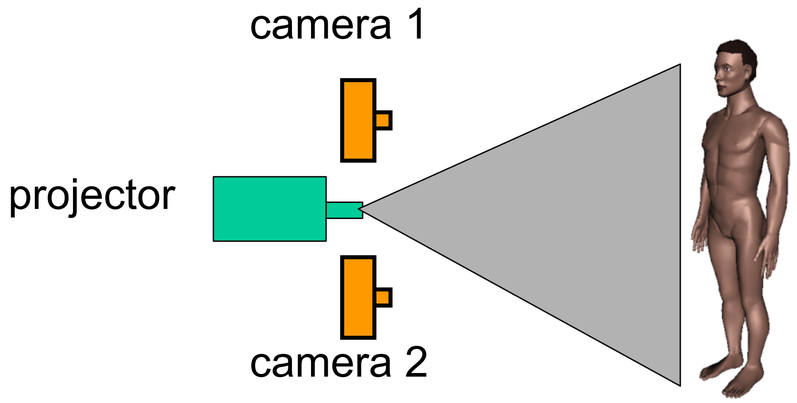


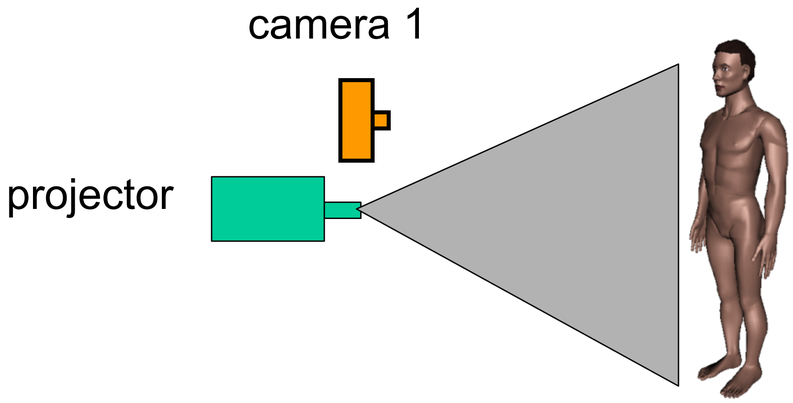
Source: A. Efros

Active stereo with structured light



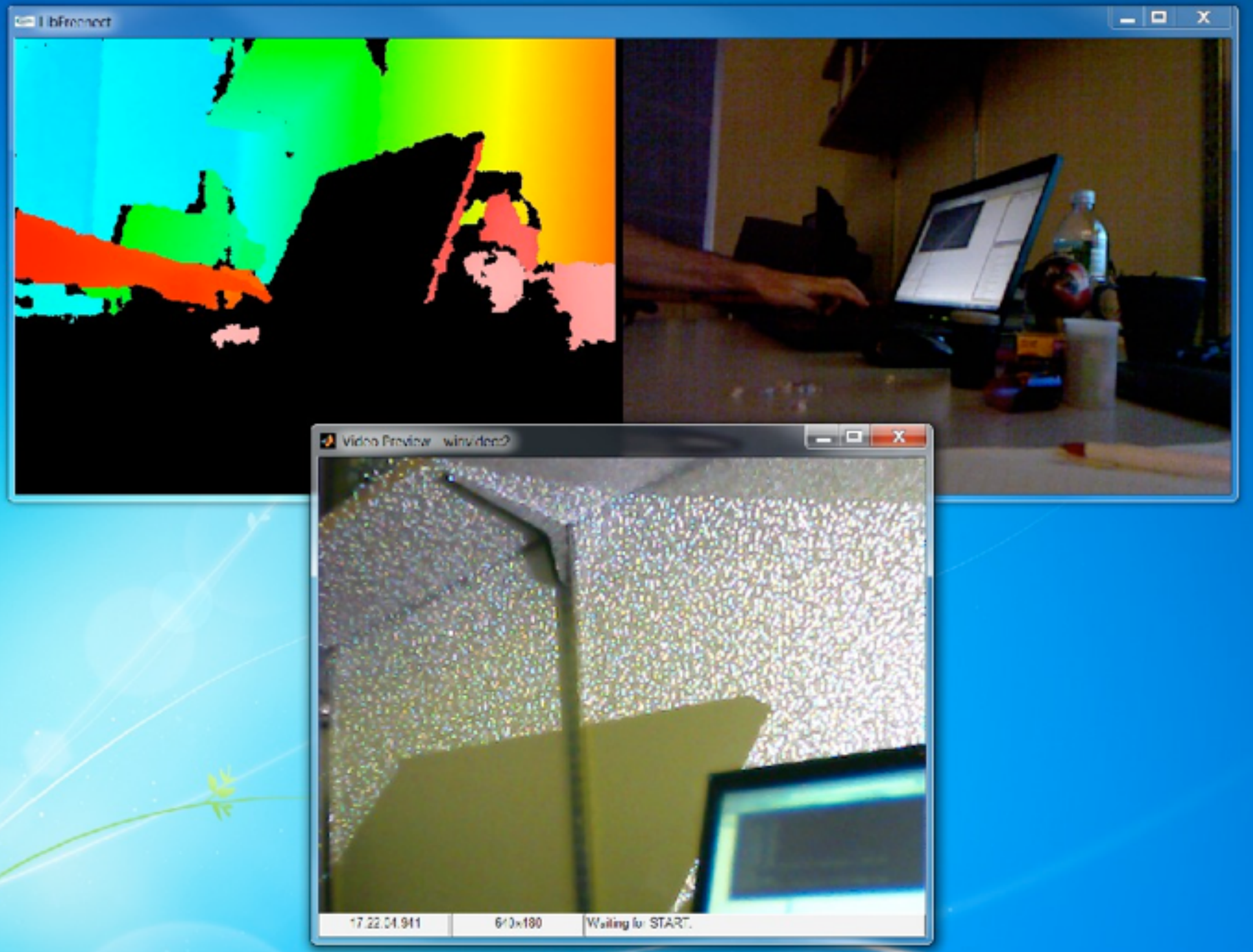
Li Zhang's one-shot stereo

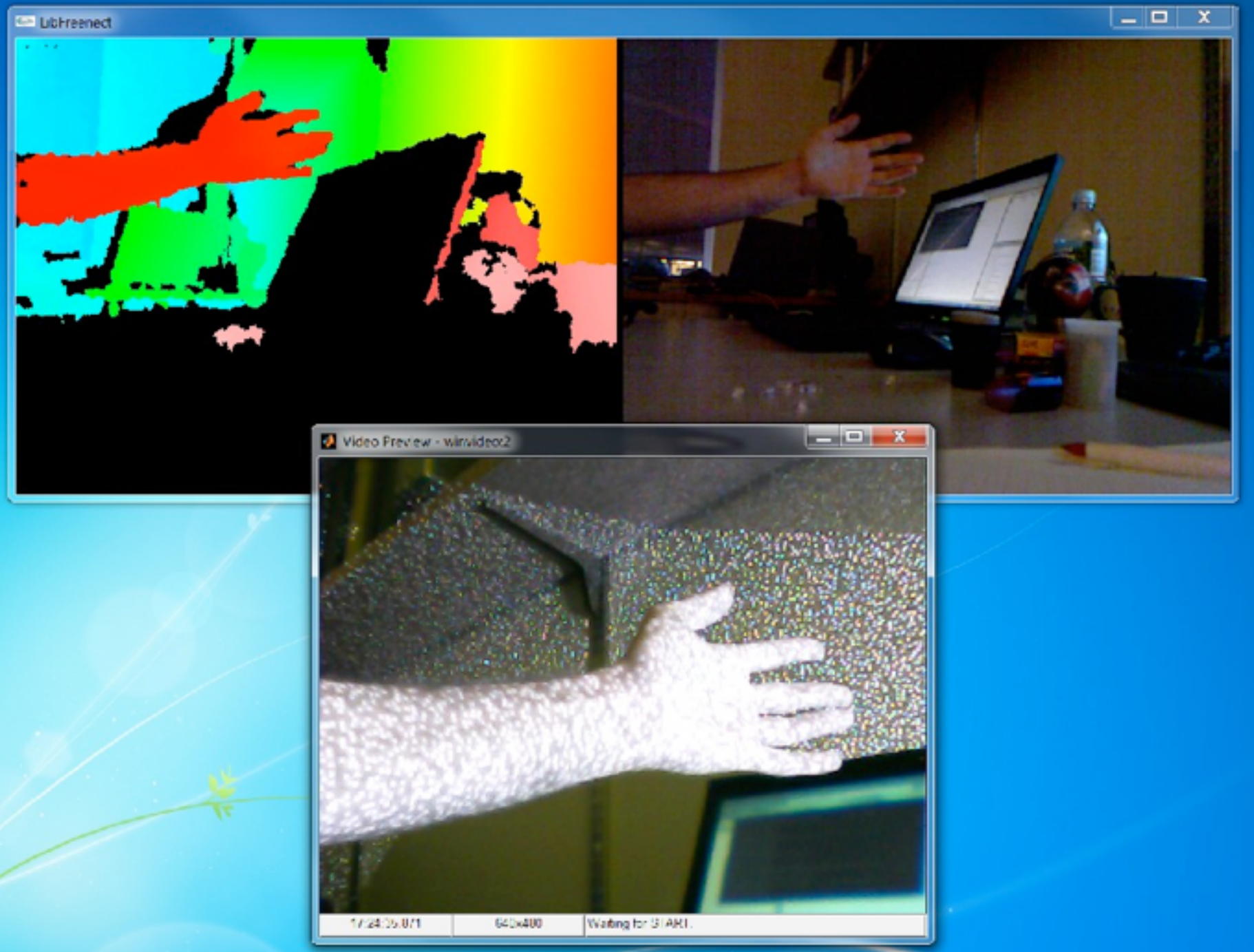




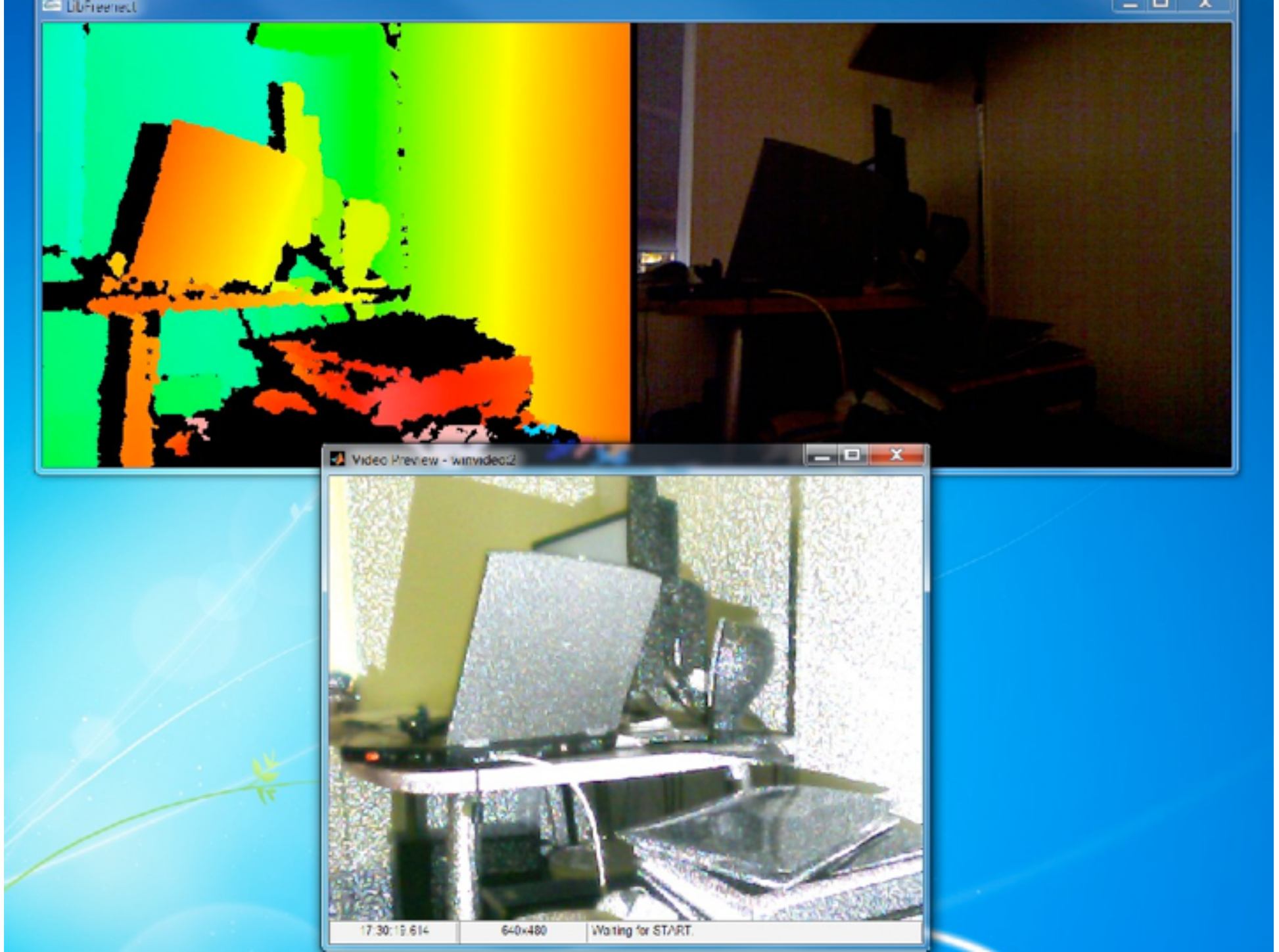
Li Zhang, Brian Curless, and Steven M. Seitz. Rapid Shape Acquisition Using Color Structured Light and Multi-pass Dynamic Programming. In *Proceedings of the 1st International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)*, Padova, Italy, June 19-21, 2002, pp. 24-36.

Source: R. Szeliski

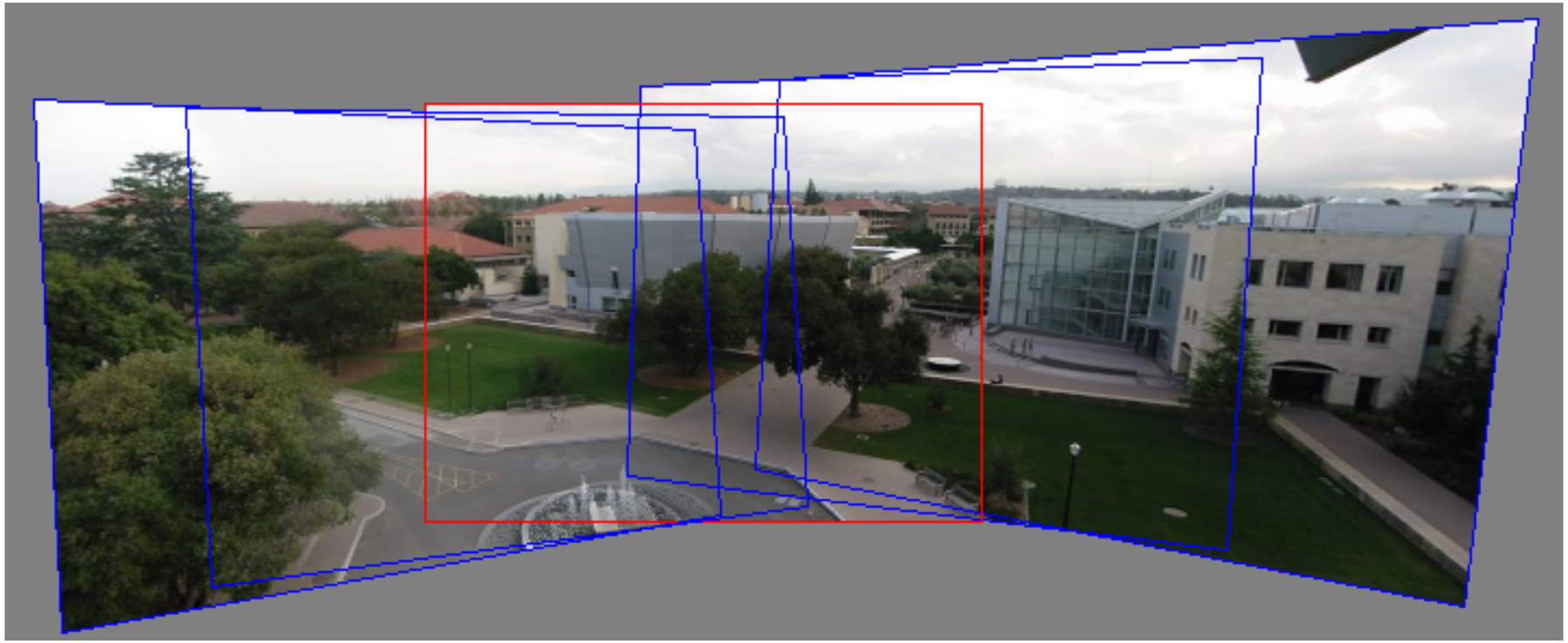

LibFreenect
Video Preview - winvideo2
17.22.04.941
640×480
Waiting for START.


Libfreenect
Video Preview - winvideo:2
640x480


LibFreenect
Video Preview - winvideo:2
17:27:03.917
640x480
Waiting for START


Video Preview - winvideo2
640×480
Waiting for START

Making panoramas



Future problem set!

Source: N. Snavely

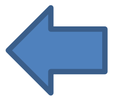
Image alignment

Why don’t these image line up exactly?

Source: N. Snavely

Recall: affine transformations

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix}$$



what happens when we change this row?

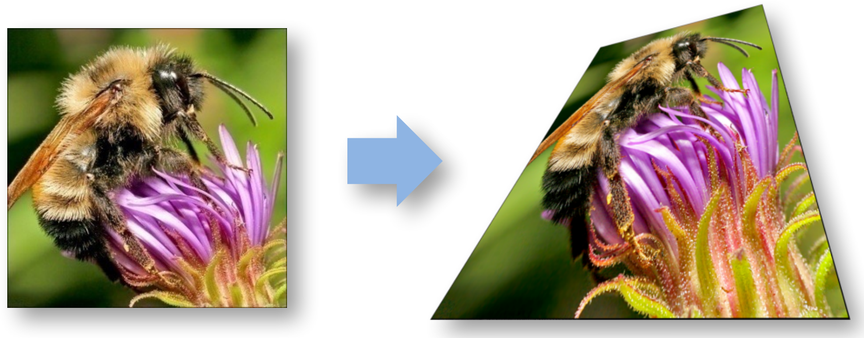
affine transformation

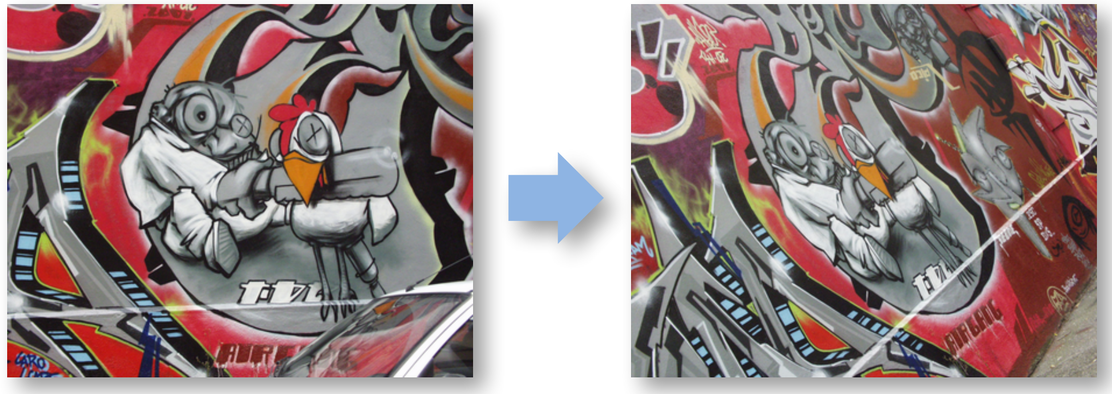
Source: N. Snavely

Projective Transformations aka Homographies aka Planar Perspective Maps

$$\mathbf{H} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix}$$

Called a **homography**
(or planar perspective map)





Source: N. Snavely

Homographies

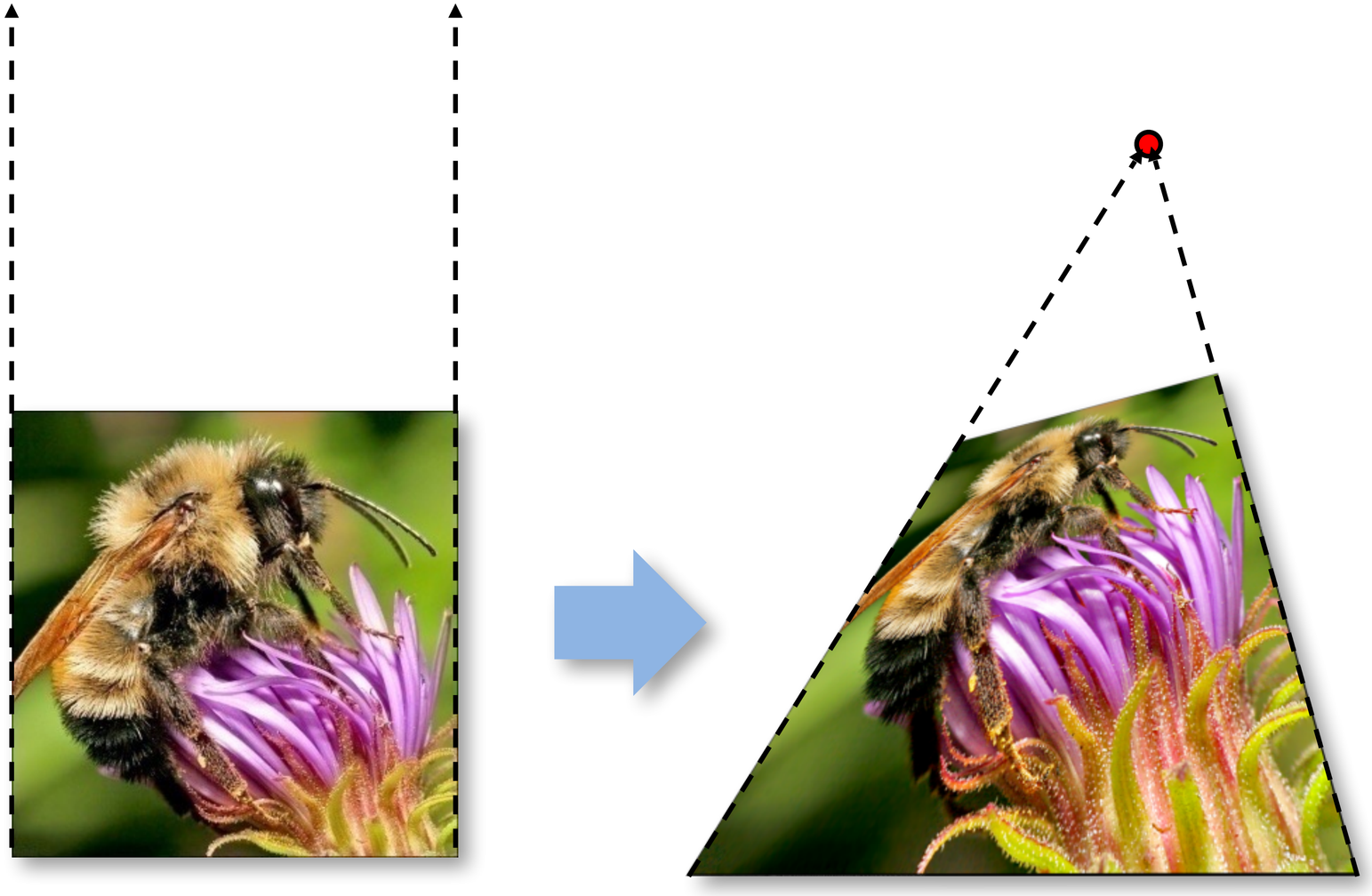
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Note that this can be 0!
A "point at infinity"

$$\sim \begin{bmatrix} \frac{ax+by+c}{gx+hy+1} \\ \frac{dx+ey+f}{gx+hy+1} \\ 1 \end{bmatrix}$$

Source: N. Snavely

Points at infinity



Source: N. Snavely

Homography

Example: two pictures taken by rotating the camera:

If we try to build a panorama by overlapping them:

Source: Torralba, Isola, Freeman

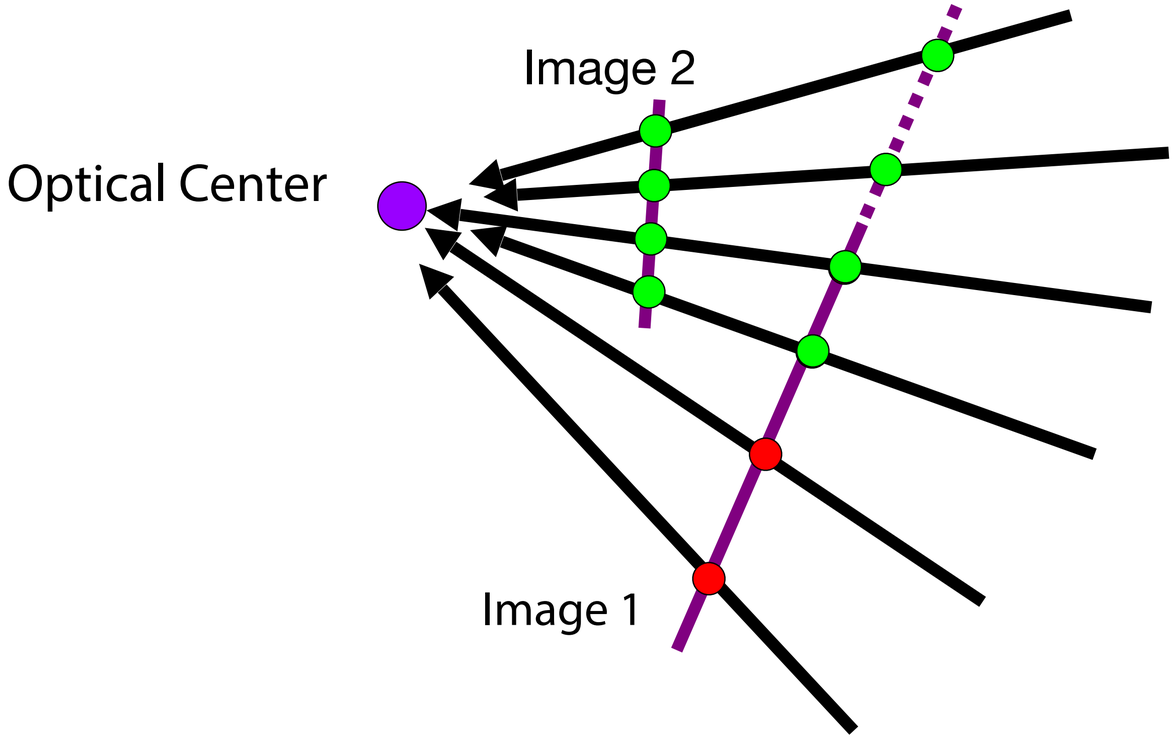
Homography

Example: two pictures taken by rotating the camera:

With a homography you can map both images into a single camera:

Source: Torralba, Isola, Freeman

Why does this work?



How do we map points in image 2 into image 1?

	image 1	image 2
intrinsics	$\mathbf{K}_1$	$\mathbf{K}_2$
extrinsics (rotation only)	$\mathbf{R}_1 = \mathbf{I}_{3\times 3}$	$\mathbf{R}_2$

Step 1: Convert pixels in image 2 to rays in camera 2's coordinate system.

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \mathbf{K}_2^{-1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

Step 2: Convert rays in camera 2's coordinates to rays in camera 1's coordinates.

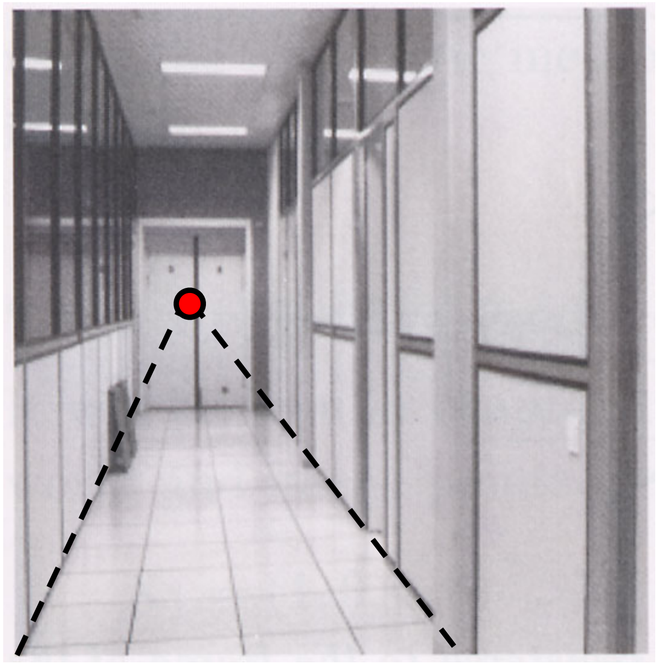
$$\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} = \mathbf{R}_2^T \mathbf{K}_2^{-1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

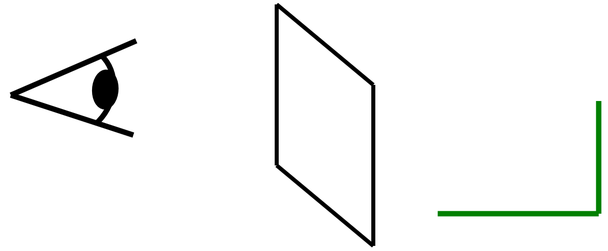
Step 3: Convert rays in camera 1's coordinates to pixels in image 1's coordinates.

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \sim \underbrace{\mathbf{K}_1 \mathbf{R}_2^T \mathbf{K}_2^{-1}}_{\text{3x3 homography}} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

Source: N. Snavely

Plane-to-plane homography





$\mathbf{H}_1$

$\mathbf{H}_2$

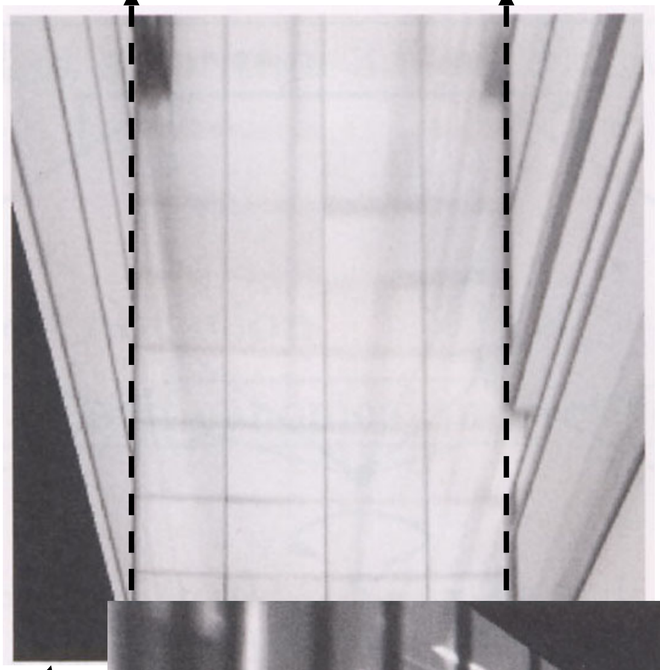


image plane in front

black area
where no pixel
maps to

Source: N. Snavely

Homographies

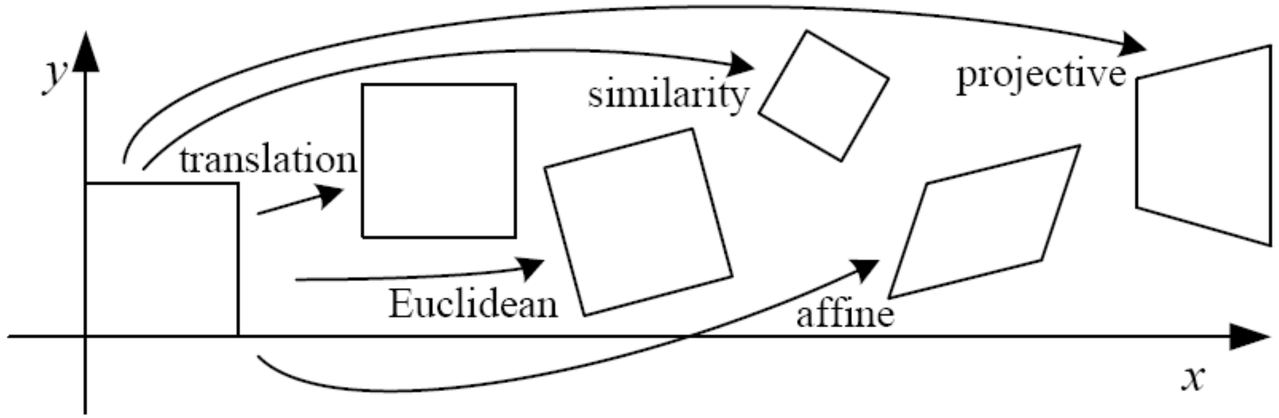
- Homographies …
  - Affine transformations, and
  - Projective warps

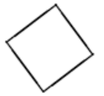
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- Properties of projective transformations:
  - Origin does not necessarily map to origin
  - Lines map to lines
  - Parallel lines do not necessarily remain parallel
  - Ratios are not preserved
  - Closed under composition

Source: N. Snavely

2D image transformations

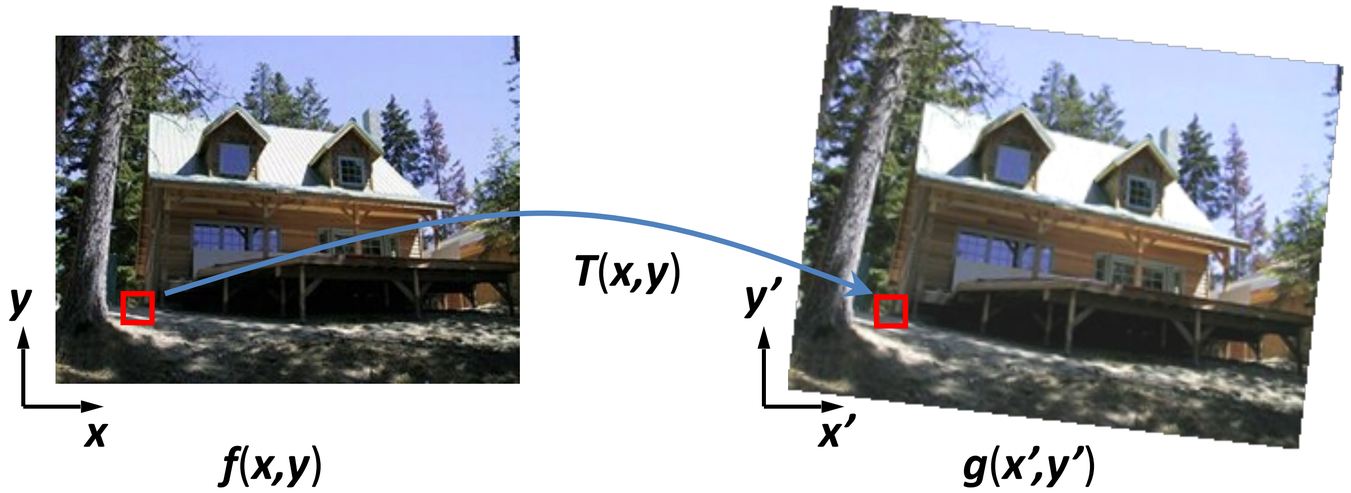


Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	2	orientation $+\cdots$	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	3	lengths $+\cdots$	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	4	angles $+\cdots$	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times3}$	6	parallelism $+\cdots$	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times3}$	8	straight lines	

Source: N. Snavely

Image warping

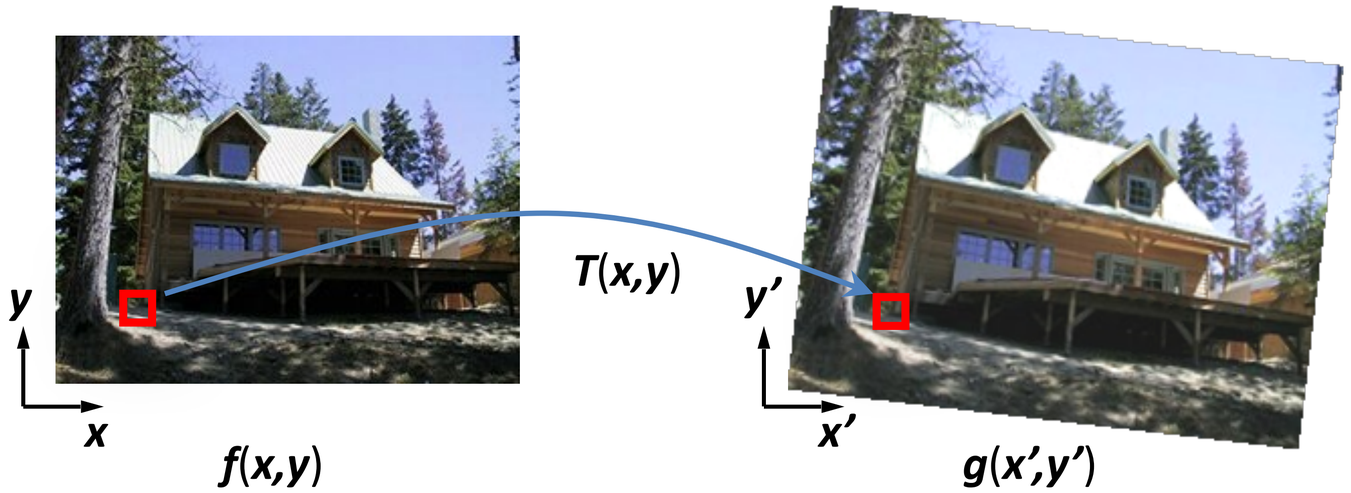
Given a coordinate transformation $(x',y') = T(x,y)$ and a source image $f(x,y)$, how do we compute a transformed image $g(x',y') = f(T(x,y))$?



Source: N. Snavely

Forward warping

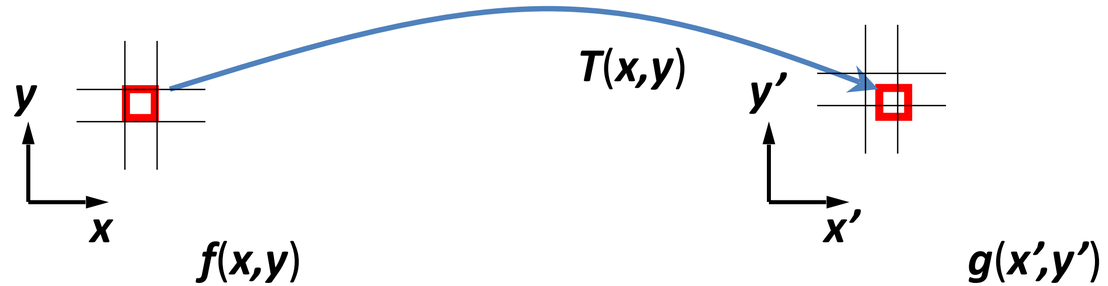
- Send each pixel $f(x)$ to its corresponding location $(x',y') = T(x,y)$ in $g(x',y')$
- What if a pixel lands "between" two pixels?



Source: N. Snavely

Forward warping

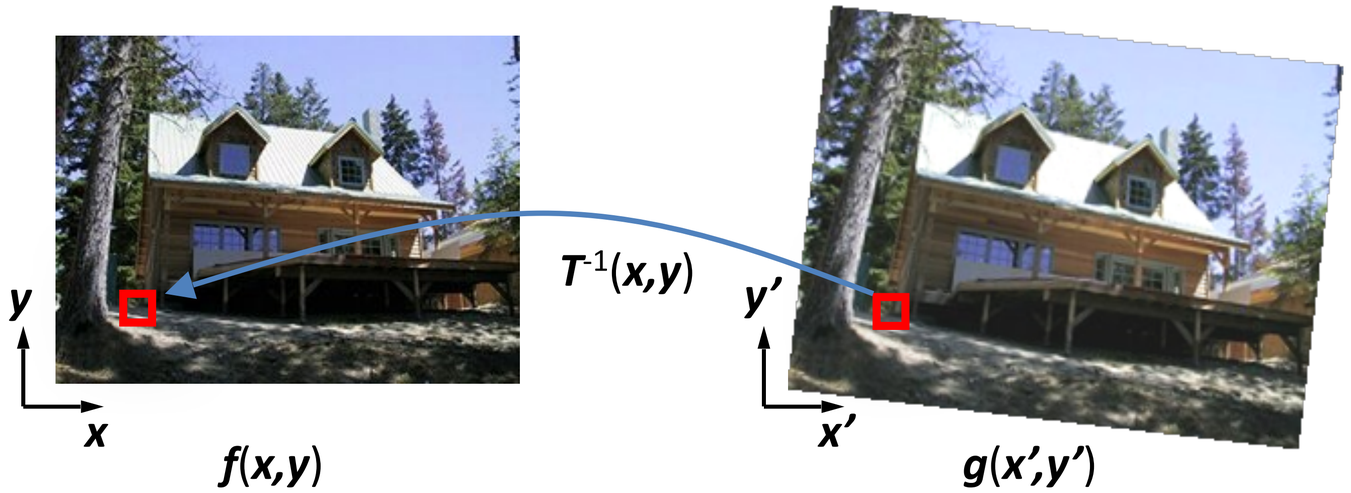
- Send each pixel $f(x)$ to its corresponding location $(x',y') = T(x,y)$ in $g(x',y')$
- What if a pixel lands "between" two pixels?
  - Answer: add "contribution" to several pixels, normalize later (*splatting*)
  - Can still result in holes



Source: N. Snavely

Inverse warping

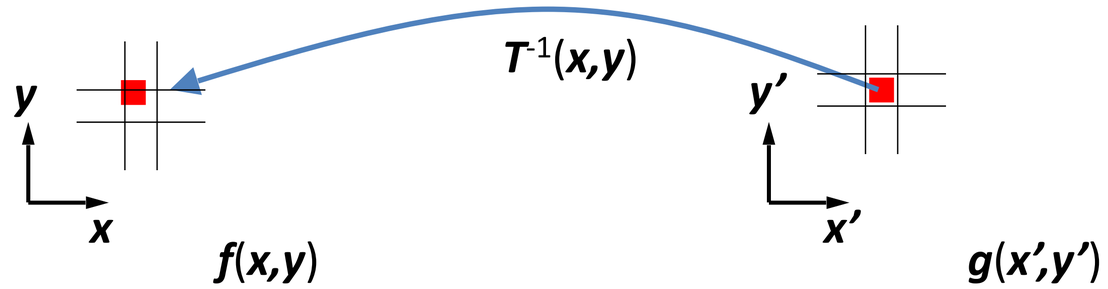
- Get each pixel $g(x',y')$ from its corresponding location $(x,y) = T^{-1}(x,y)$ in $f(x,y)$
  - Requires taking the inverse of the transform
  - What if pixel comes from "between" two pixels?



Source: N. Snavely

Inverse warping

- Get each pixel $g(x')$ from its corresponding location $x' = h(x)$ in $f(x)$
  - What if pixel comes from "between" two pixels?
  - Answer: *resample* color value from *interpolated* (*prefiltered*) source image



Source: N. Snavely

Interpolation

- Possible interpolation filters:
  - nearest neighbor
  - bilinear
  - bicubic (interpolating)
  - sinc
- Needed to prevent “jaggies” and aliasing artifacts

  (with prefiltering)

Source: N. Snavely

Next lecture: estimating geometry from images