Stereo

Computer Vision
Fall 2018
Columbia University

Homework

- Homework 2 grades are back
  - Median 37/40, std 7.2
- Homework 3 due now
- Homework 4 out today

My Office Hours

- Now Mondays 5pm-6pm

Course Evaluations

- 60% response rate so far
- Please respond by tomorrow
- We read all feedback!

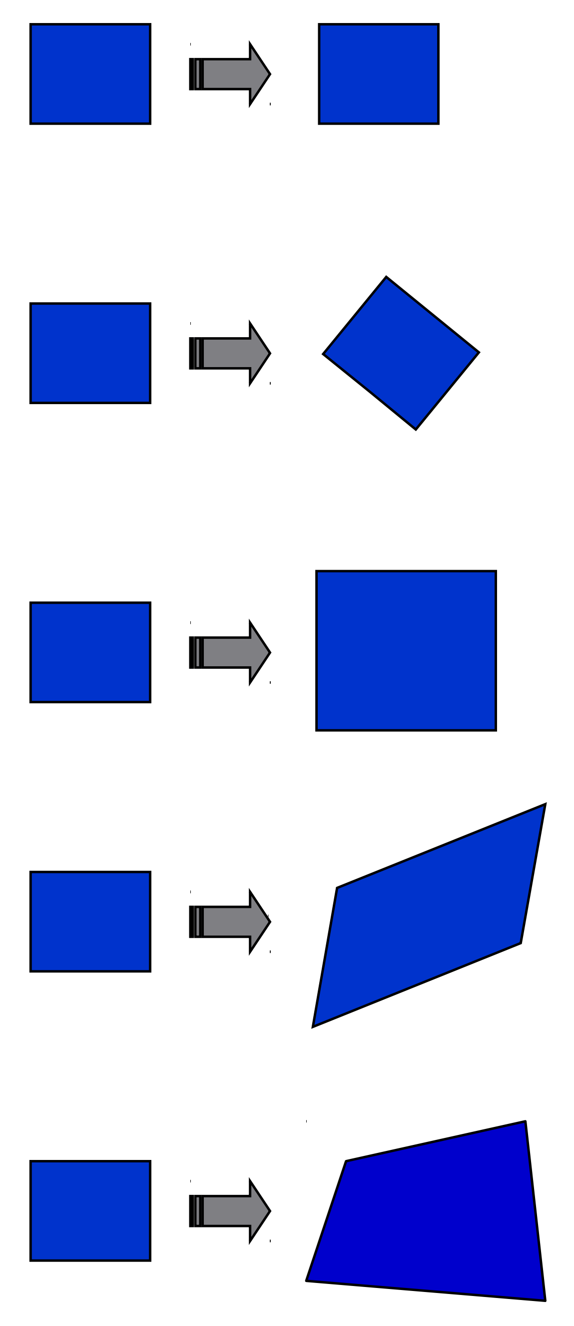
Image Stitching

Image alignment

Why don’t these image line up exactly?

Transformation Models

- Translation only
- Rigid body (translate+rotate)
- Similarity (translate+rotate+scale)
- Affine
- Homography (projective)



Camera Projection

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_x \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Camera Intrinsics

Camera Extrinsics

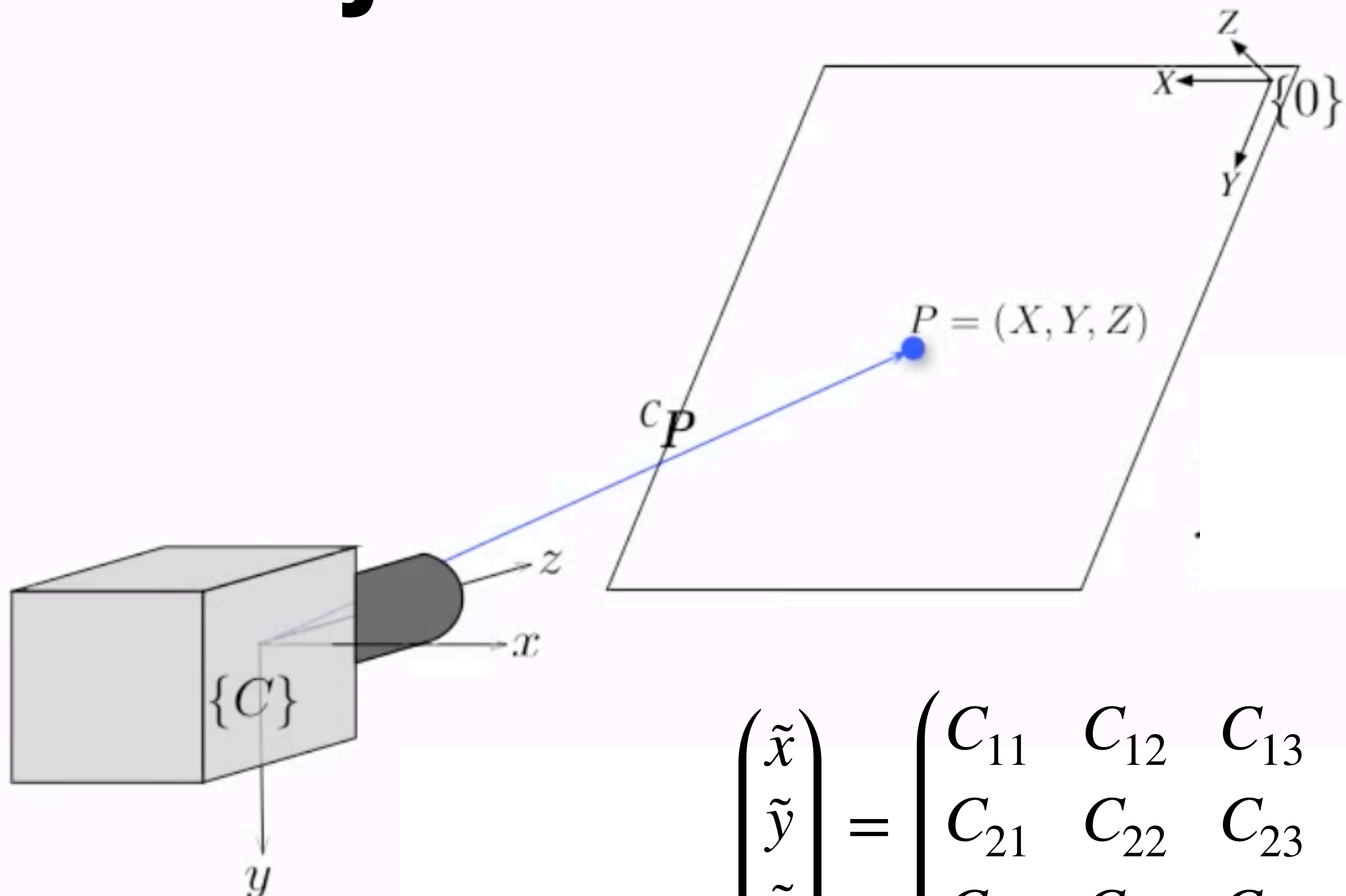
World Coordinates

Camera Matrix

Mapping points from the world to image coordinates is matrix multiplication in homogenous coordinates

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & C_{34} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Projection of 3D Plane

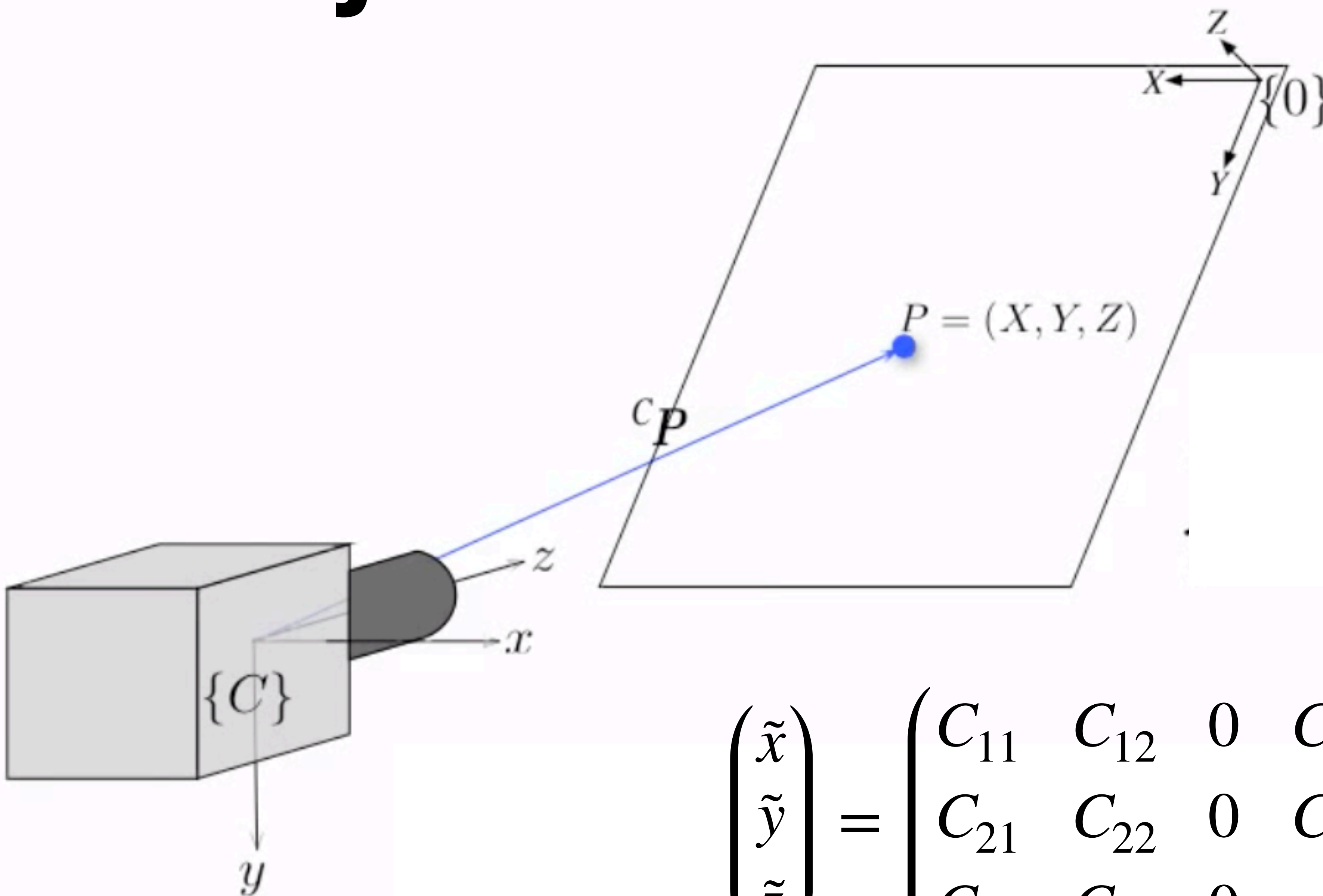


All points on the plane have Z = 0

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix}$$

Slide credit: Peter Corke

Projection of 3D Plane

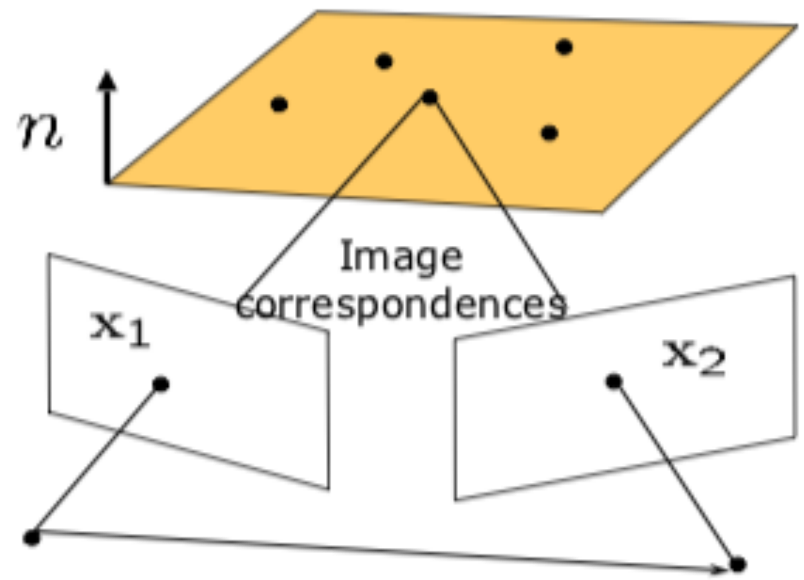


All points on the plane have Z = 0

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} & 0 & C_{14} \\ C_{21} & C_{22} & 0 & C_{24} \\ C_{31} & C_{32} & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix}$$

Slide credit: Peter Corke

Two-views of Plane



$$\begin{pmatrix} \tilde{x}_1 \\ \tilde{y}_1 \\ \tilde{z}_1 \end{pmatrix} = H_1 \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \qquad \begin{pmatrix} \tilde{x}_2 \\ \tilde{y}_2 \\ \tilde{z}_2 \end{pmatrix} = H_2 \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} \tilde{x}_2 \\ \tilde{y}_2 \\ \tilde{z}_2 \end{pmatrix} = H_2 H_1^{-1} \begin{pmatrix} \tilde{x}_1 \\ \tilde{y}_1 \\ \tilde{z}_1 \end{pmatrix}$$

Slide credit: Deva Ramanan

Image Alignment Algorithm

Given images A and B

1. Compute image features for A and B
2. Match features between A and B
3. Compute homography between A and B using least squares on set of matches

What could go wrong?

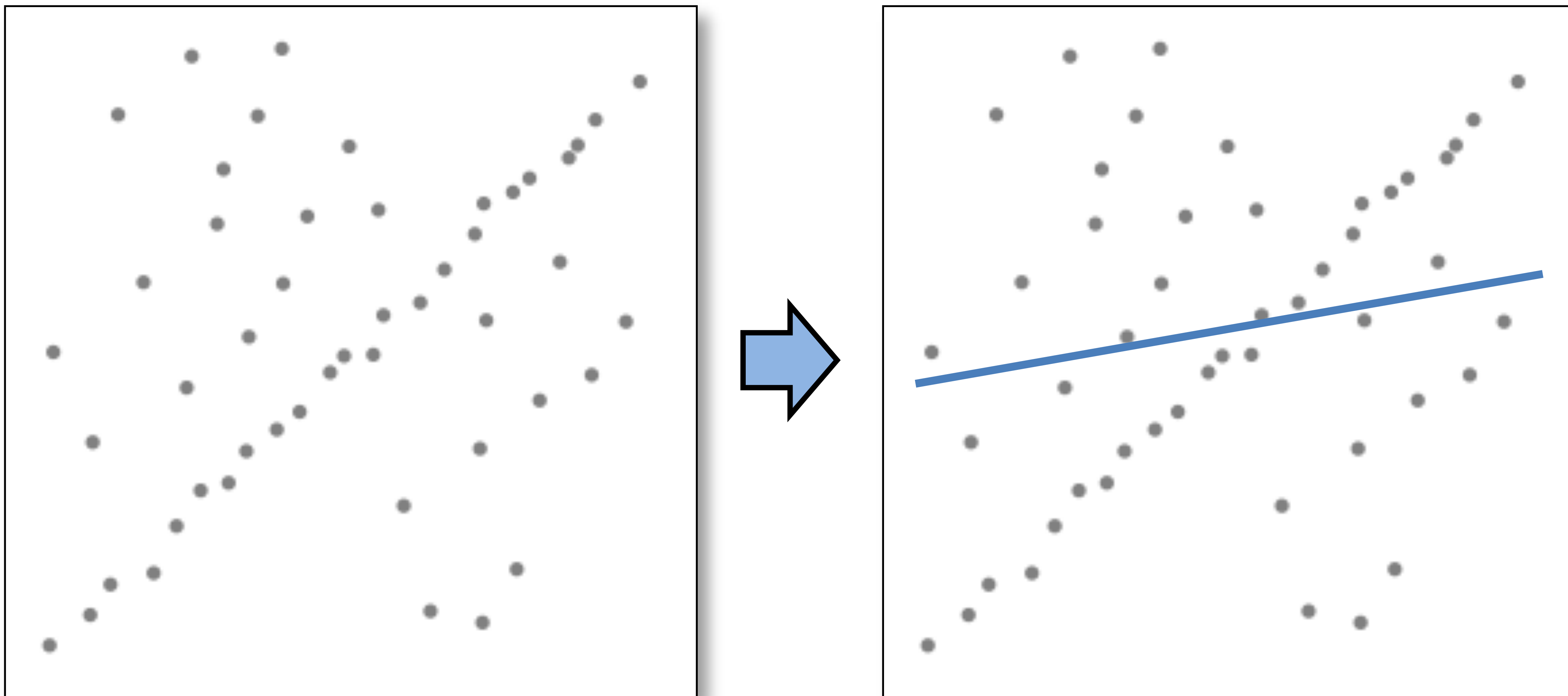
Slide credit: Noah Snavely

Outliers

Slide credit: Noah Snavely

Robustness

- Let's consider a simpler example… linear regression



Problem: Fit a line to these datapoints

Least squares fit

- How can we fix this?

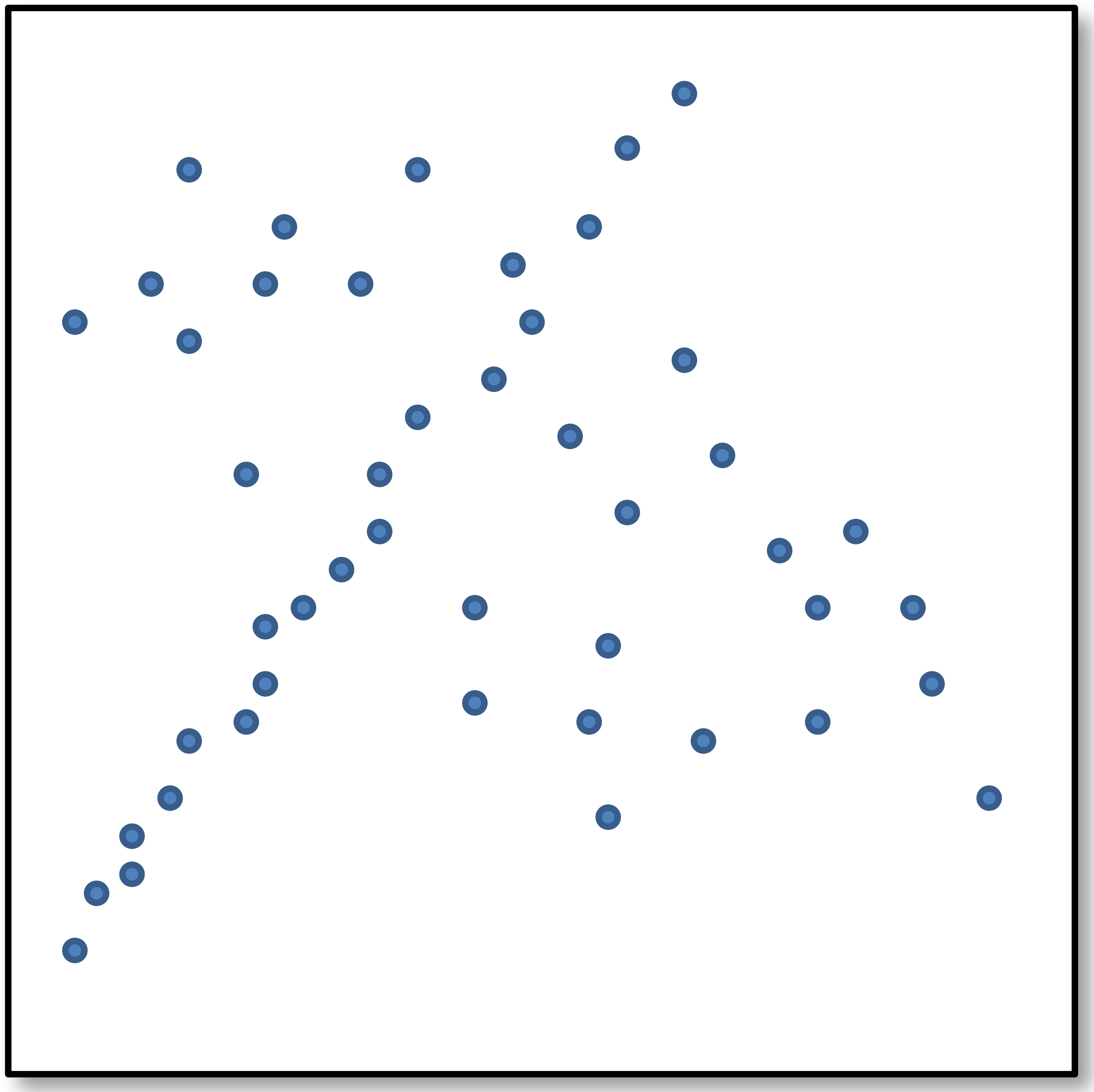
Slide credit: Noah Snavely

We need a better cost function…

- Suggestions?

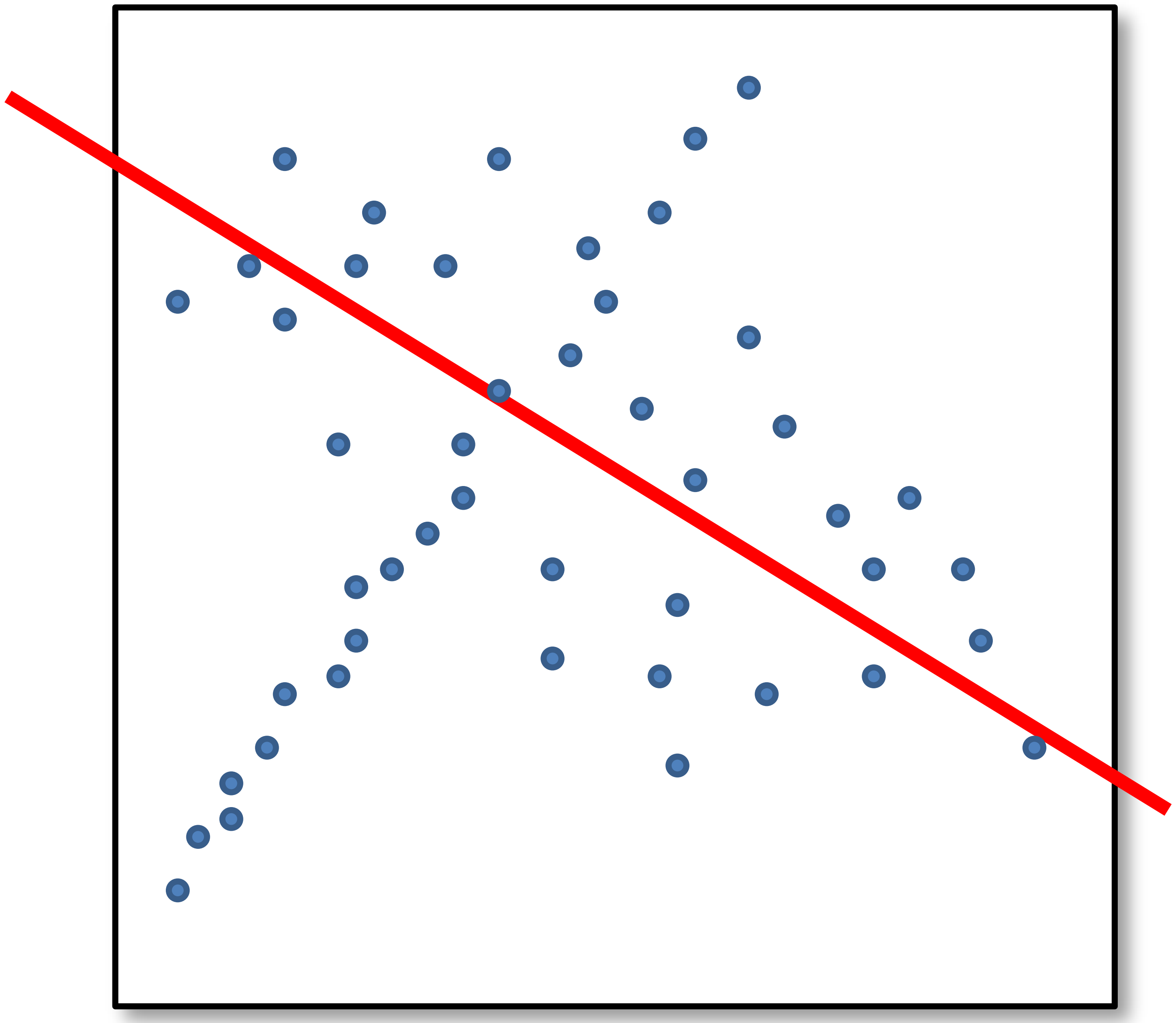
Slide credit: Noah Snavely

Counting inliers



Slide credit: Noah Snavely

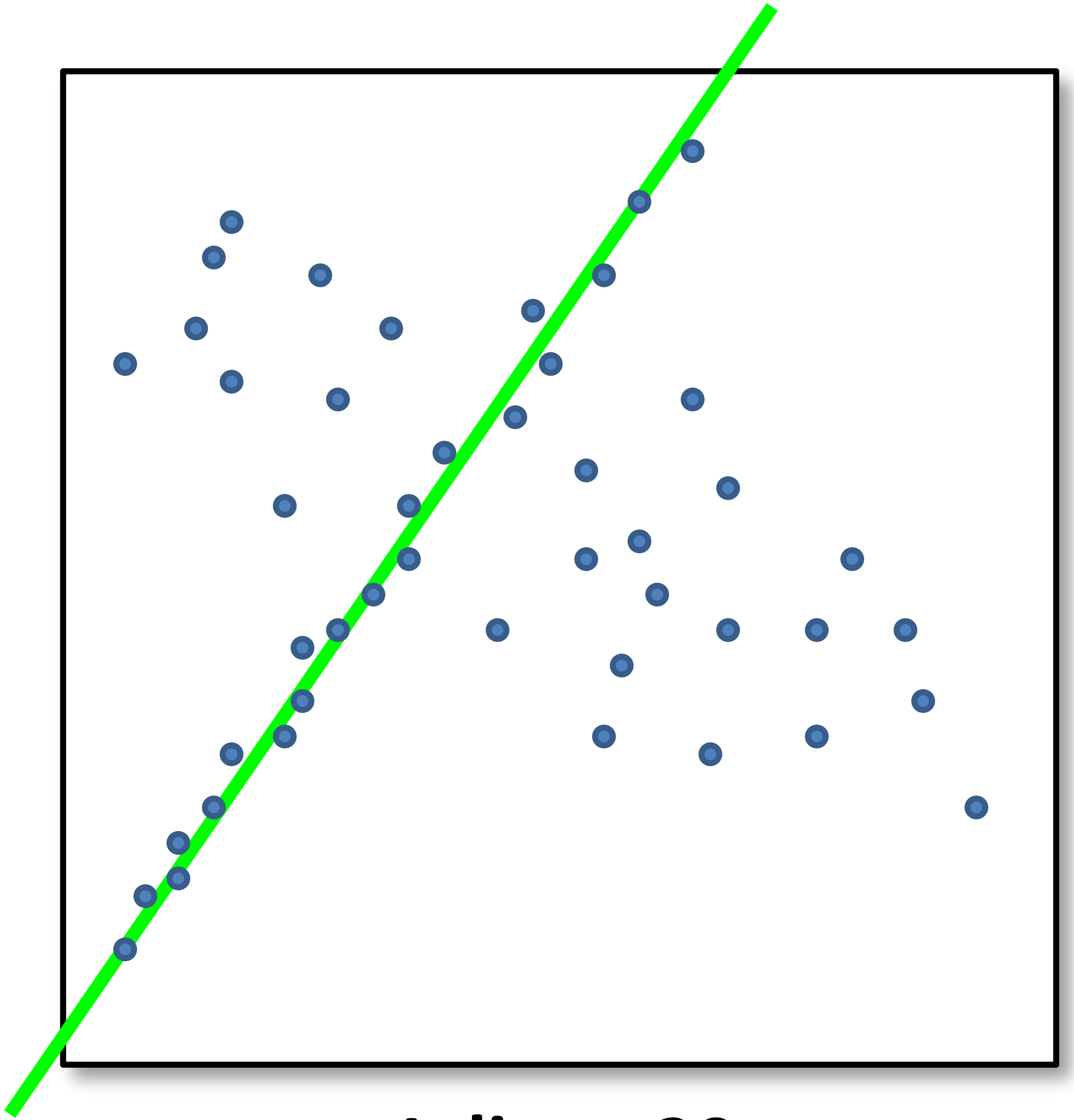
Counting inliers



Inliers: 3

Slide credit: Noah Snavely

Counting inliers



Inliers: 20

Slide credit: Noah Snavely

Idea

- Given a hypothesized line
- Count the number of points that “agree” with the line
  - “Agree” = within a small distance of the line
  - I.e., the **inliers** to that line

- For all possible lines, select the one with the largest number of inliers

How do we find the best line?

- Unlike least-squares, no simple closed-form solution

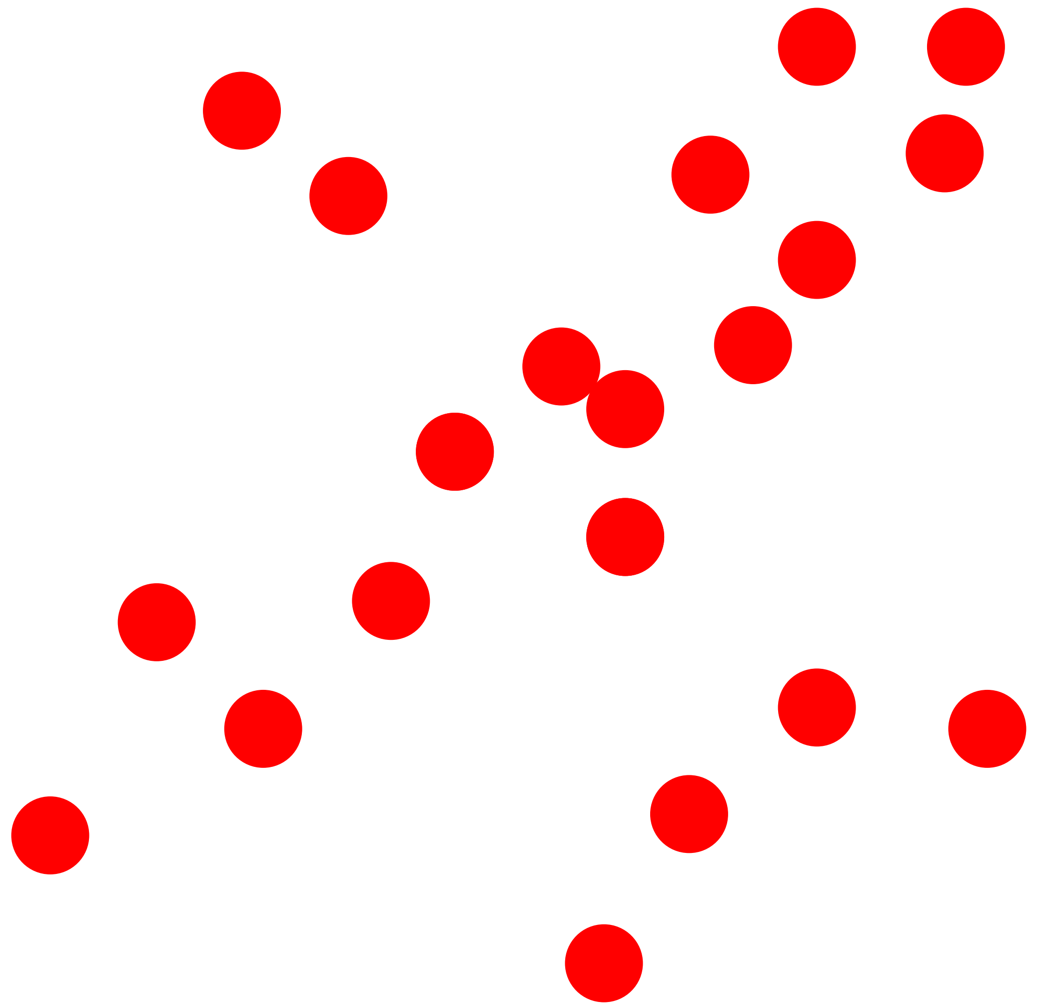
- Hypothesize-and-test
  - Try out many lines, keep the best one
  - Which lines?

Slide credit: Noah Snavely

RANSAC

(RANdom SAmple Consensus) :

Fischler & Bolles in ‘81.



Algorithm:

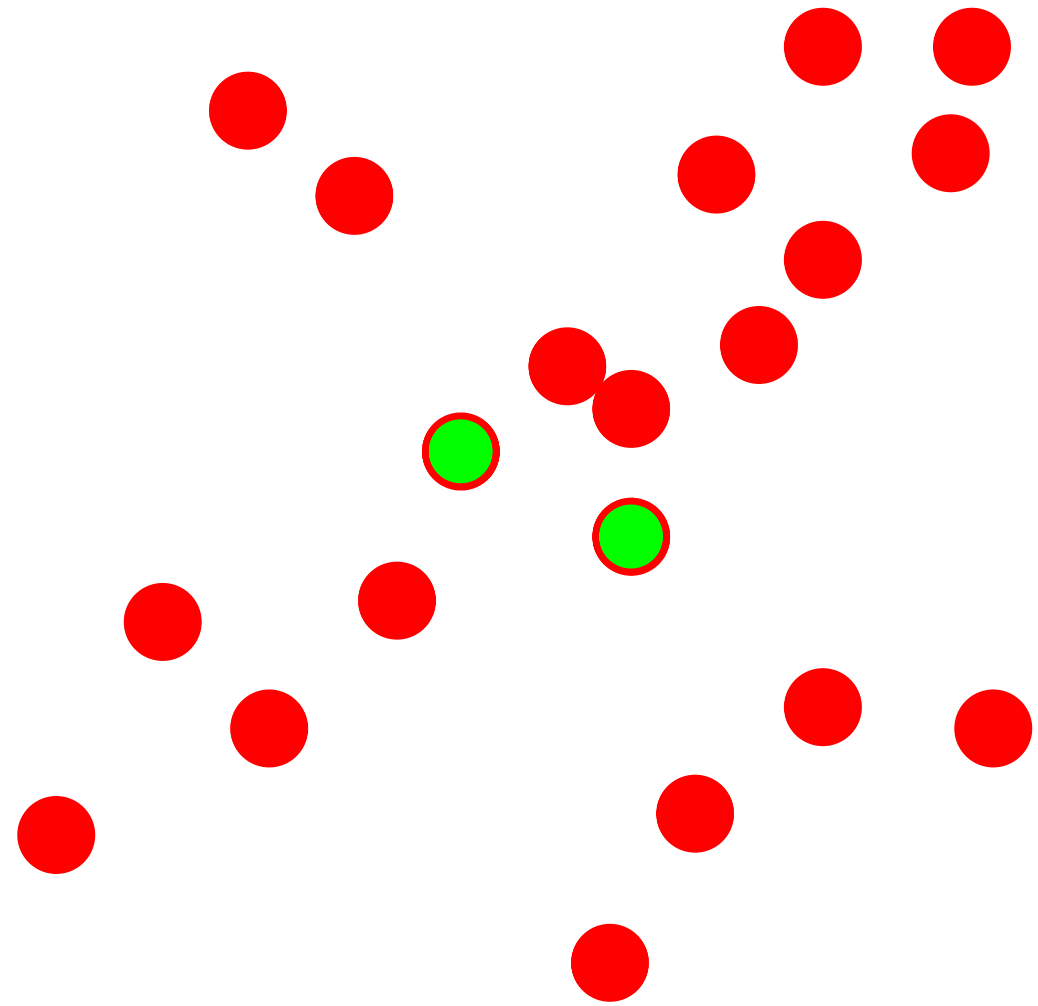
1. **Sample** (randomly) the number of points *s* required to fit the model
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Slide credit: James Hays

RANSAC

Line fitting example



Algorithm:

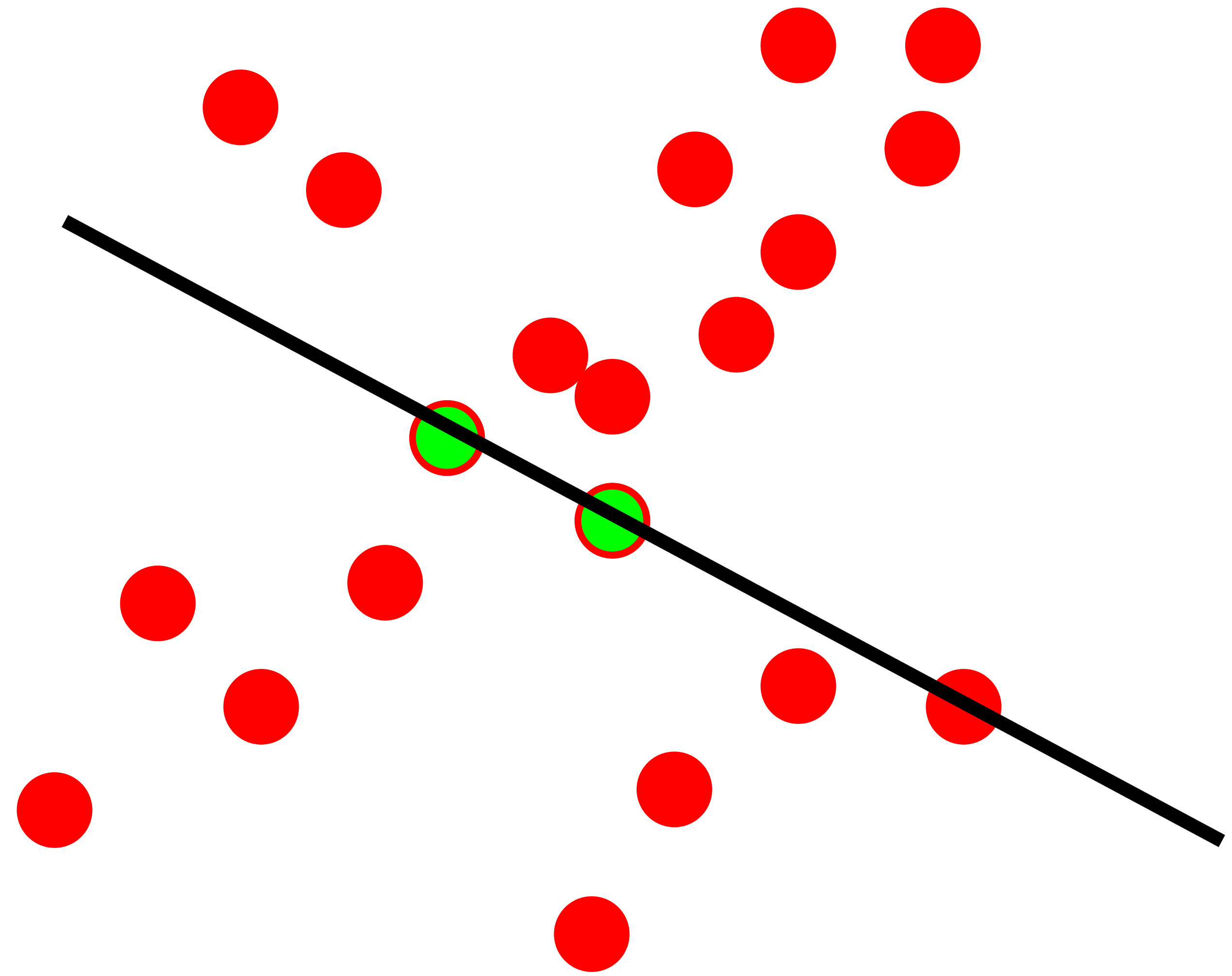
1. **Sample** (randomly) the number of points required to fit the model (s=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Illustration by Savarese

RANSAC

Line fitting example



Algorithm:

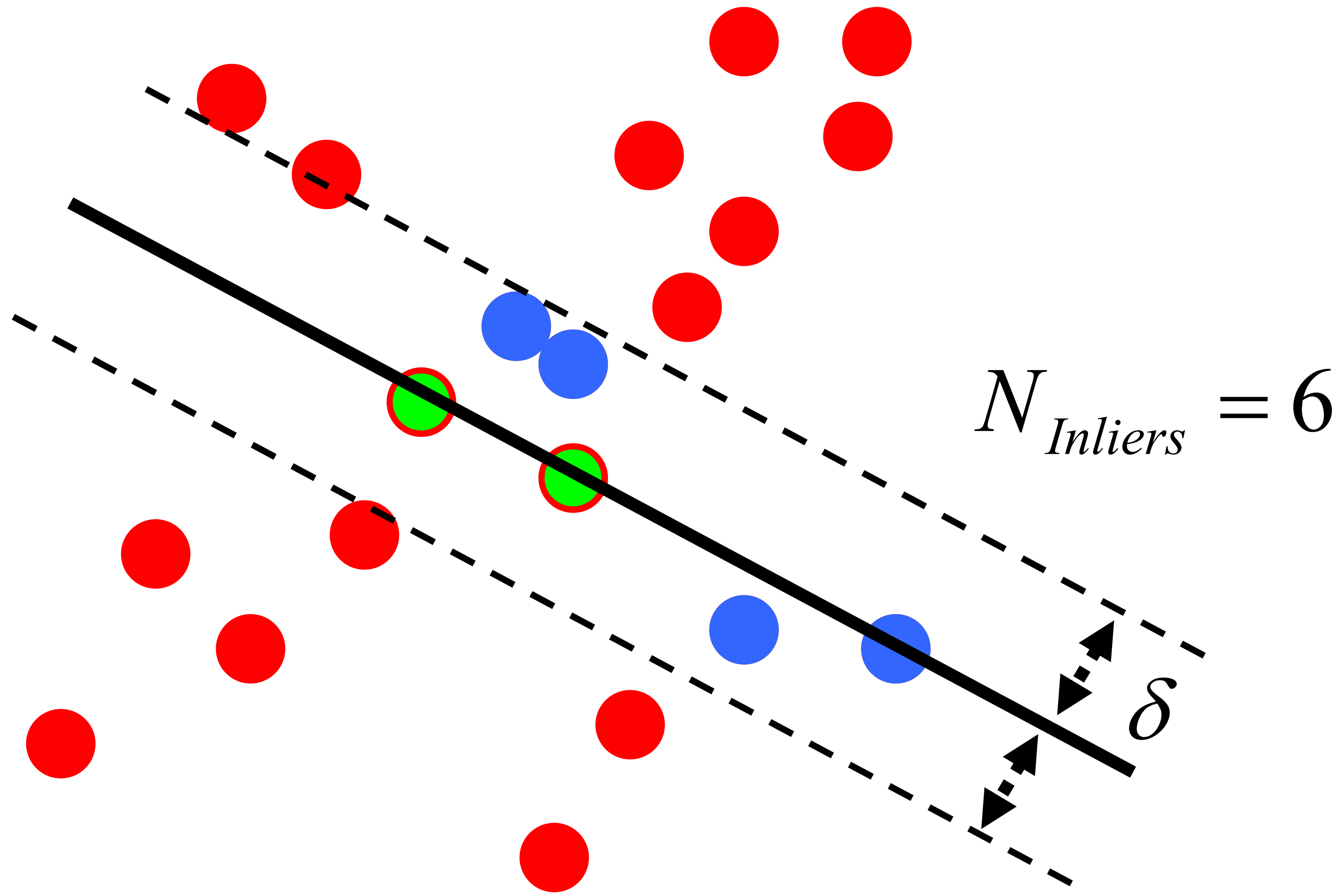
1. **Sample** (randomly) the number of points required to fit the model (*s*=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Slide credit: James Hays

RANSAC

Line fitting example



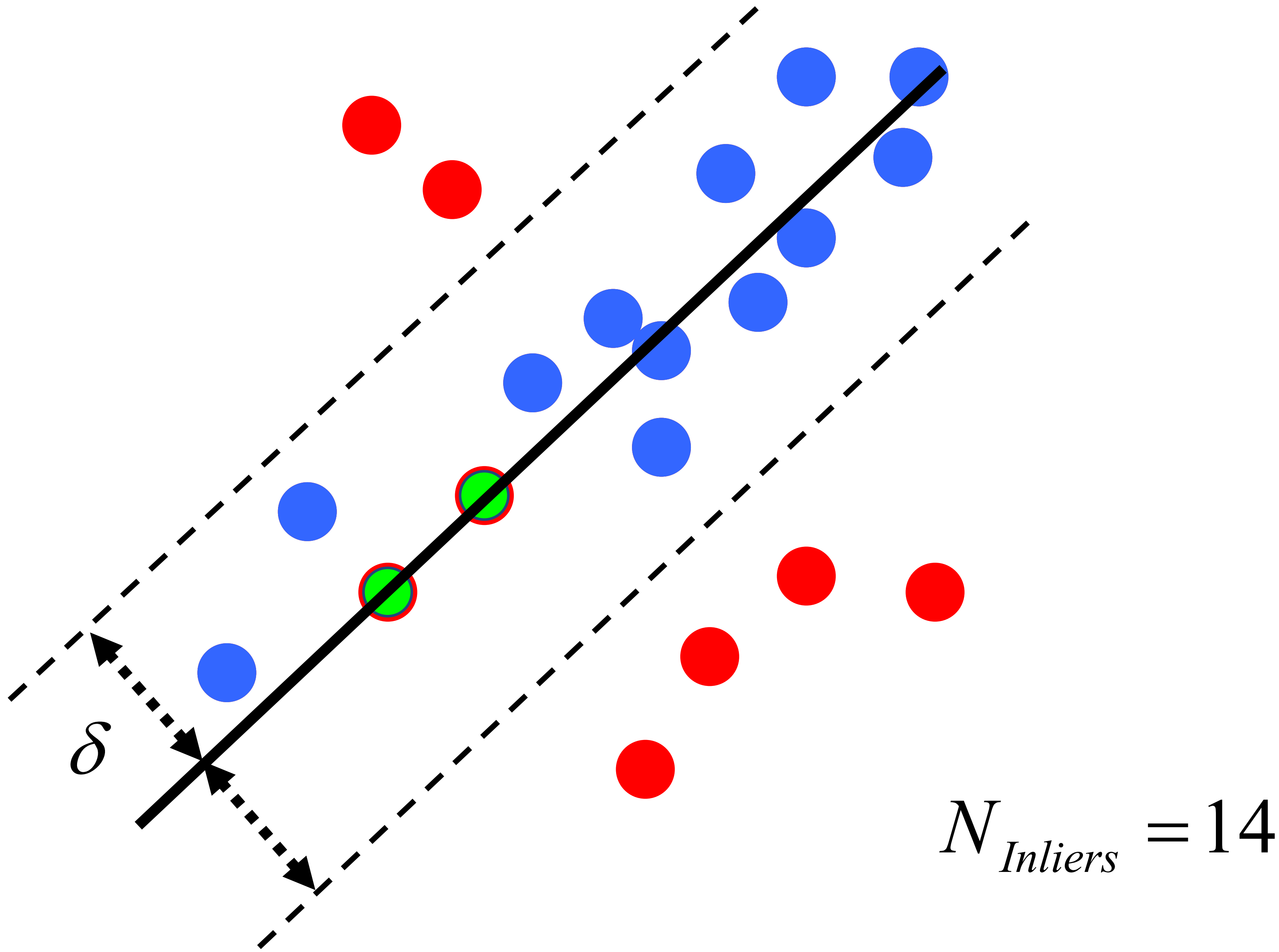
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (*s*=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Slide credit: James Hays

RANSAC



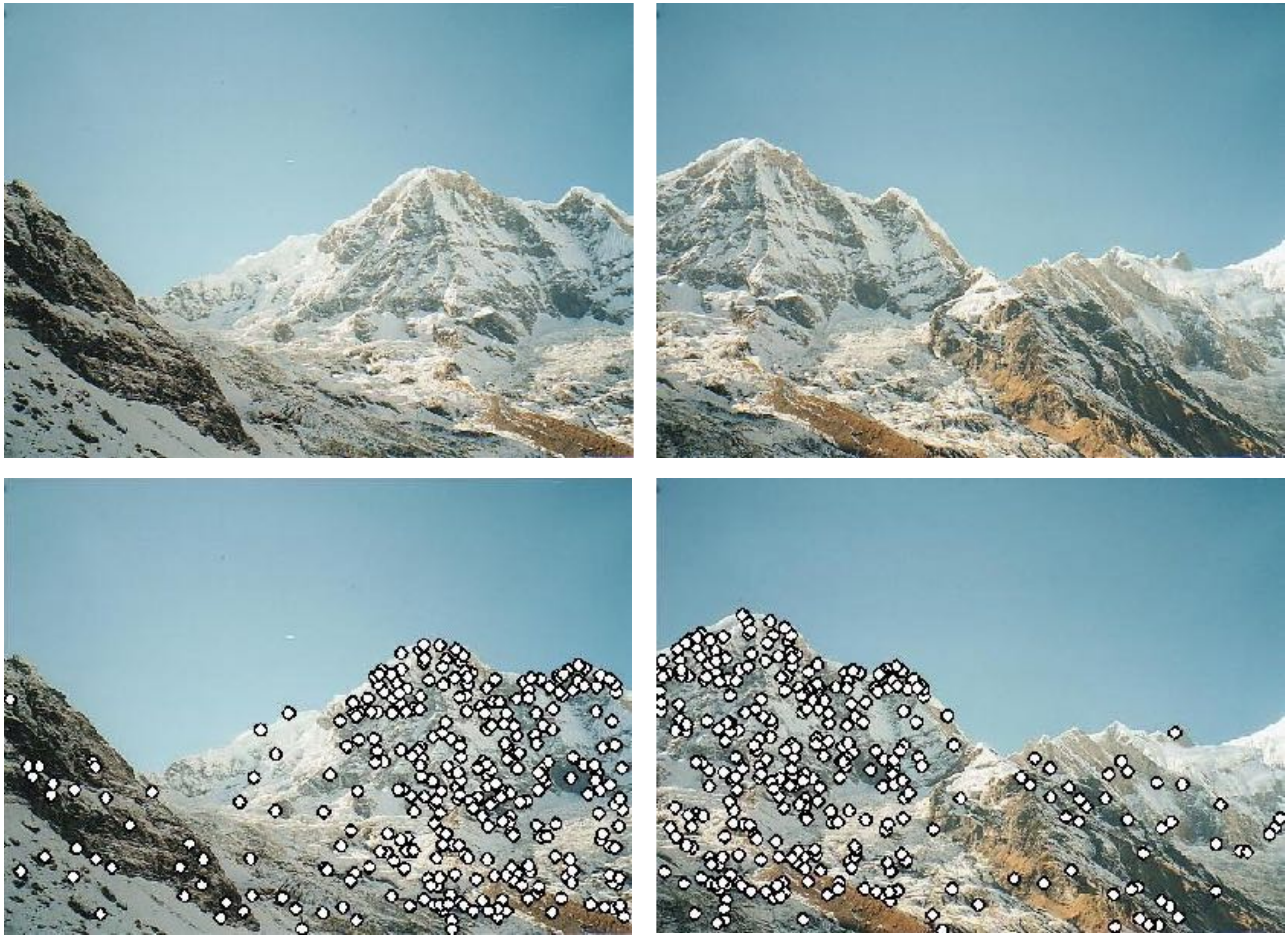
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (s=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

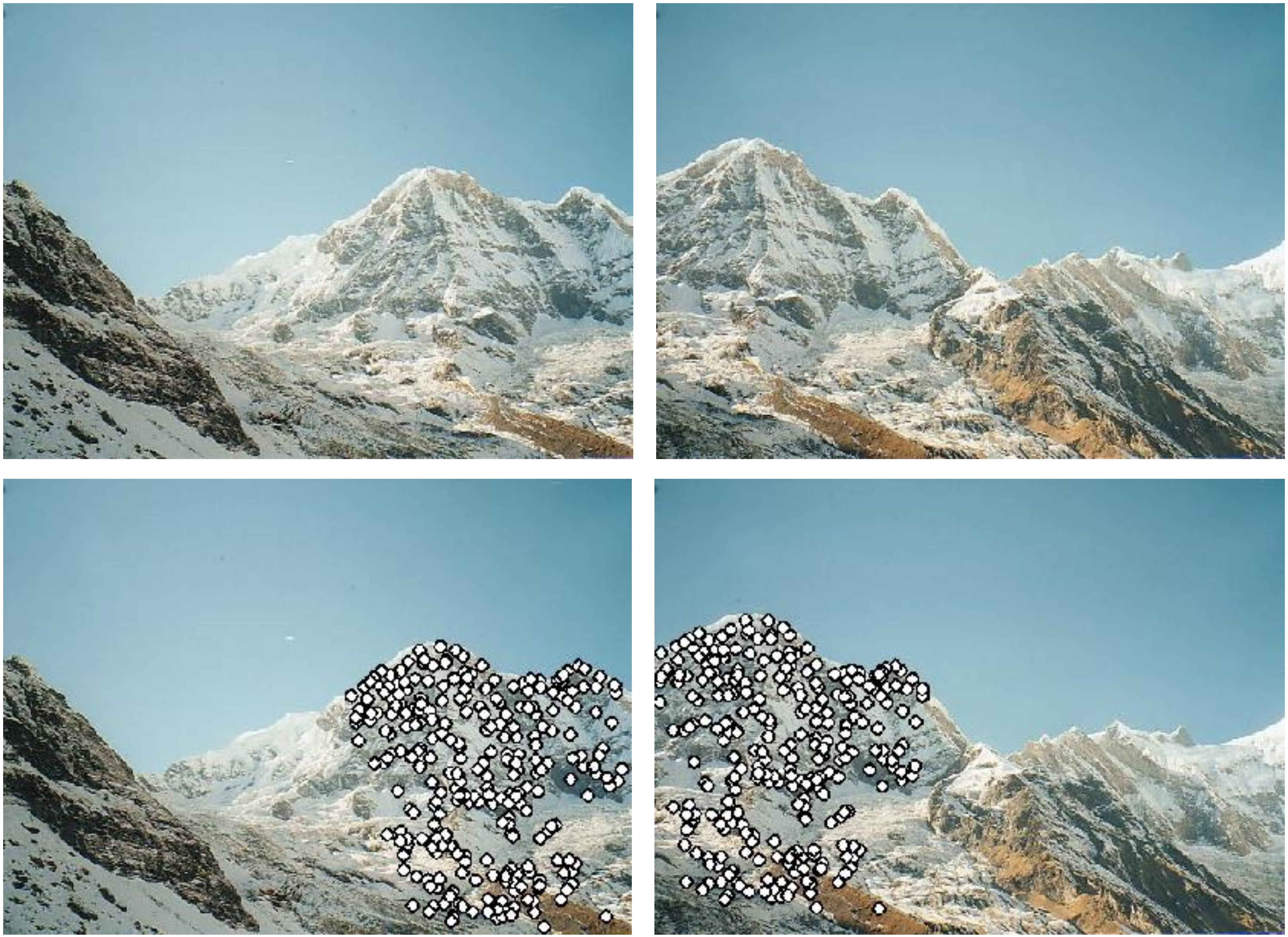
Slide credit: James Hays

RANSAC for alignment



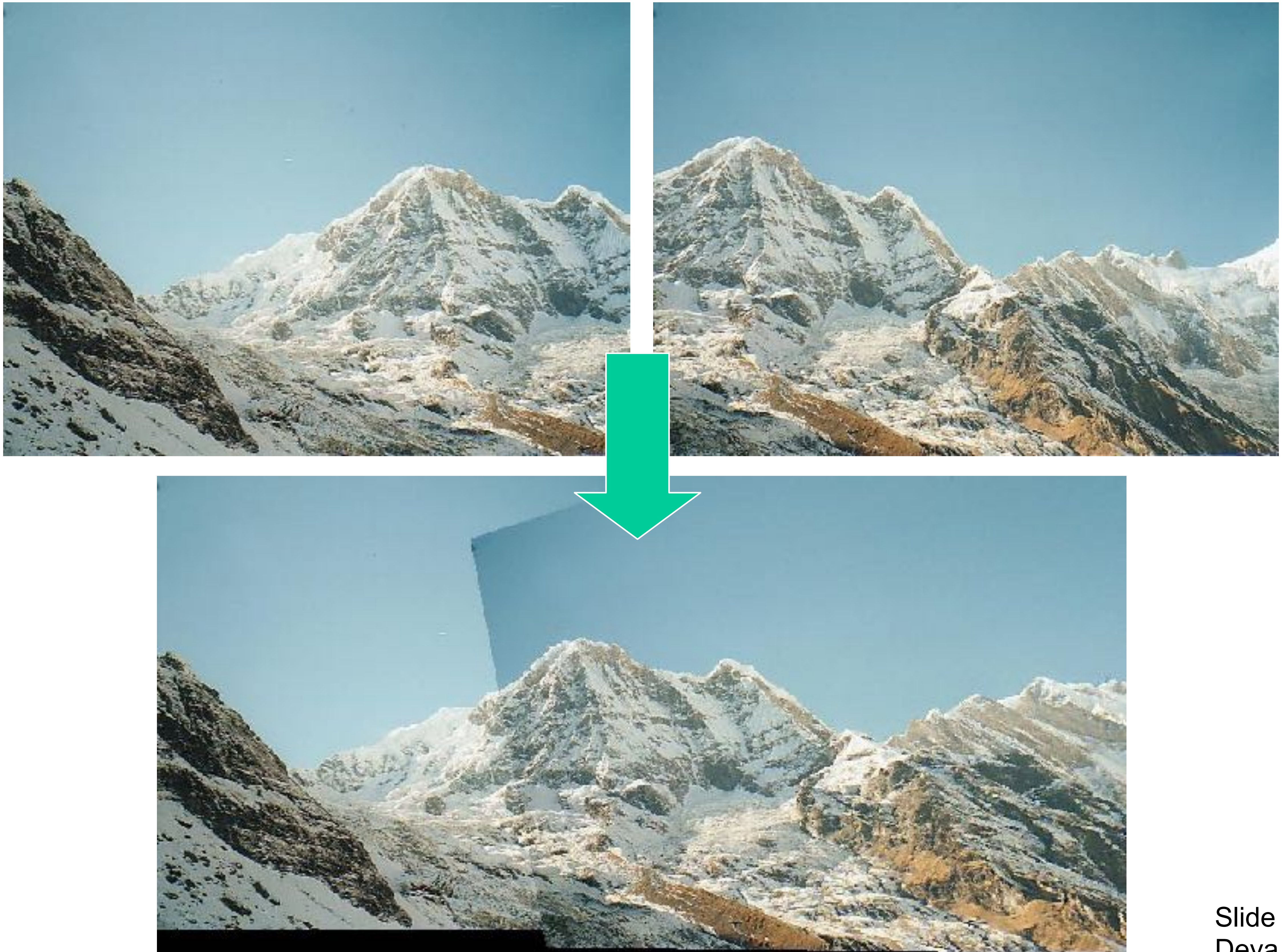
Slide credit: Deva Ramanan

RANSAC for alignment



Slide credit: Deva Ramanan

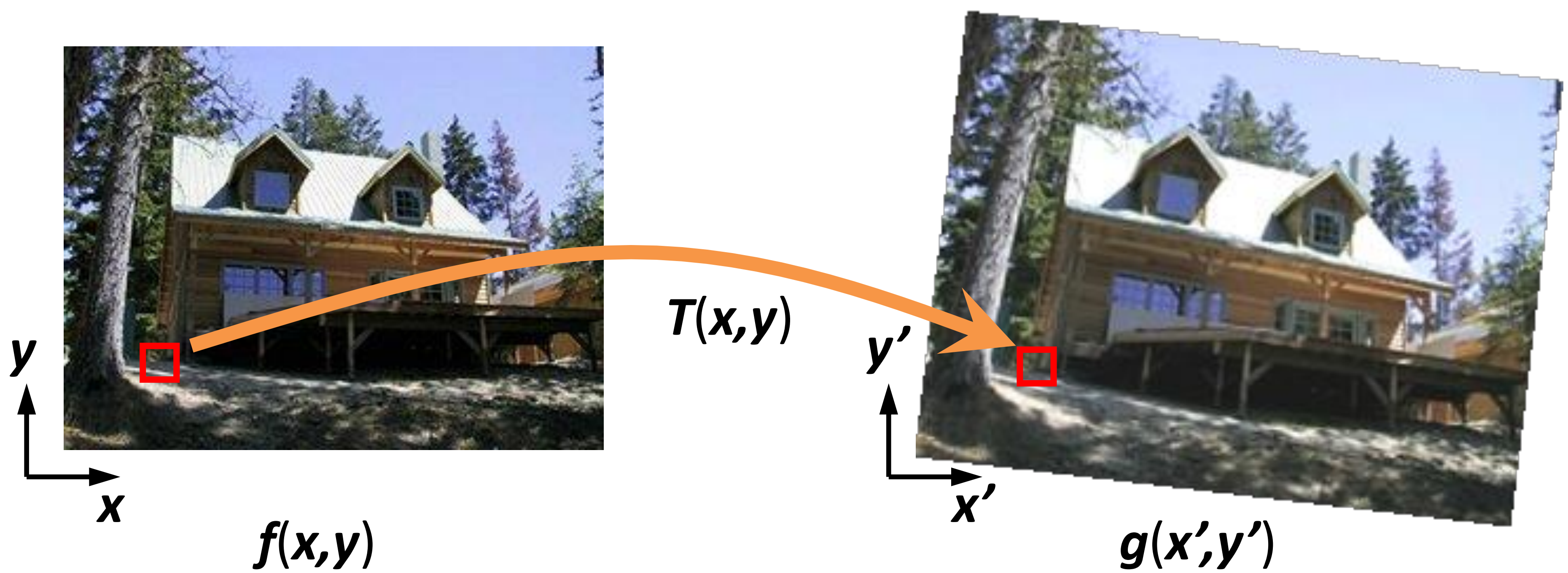
RANSAC for alignment



Slide credit:
Deva Ramanan

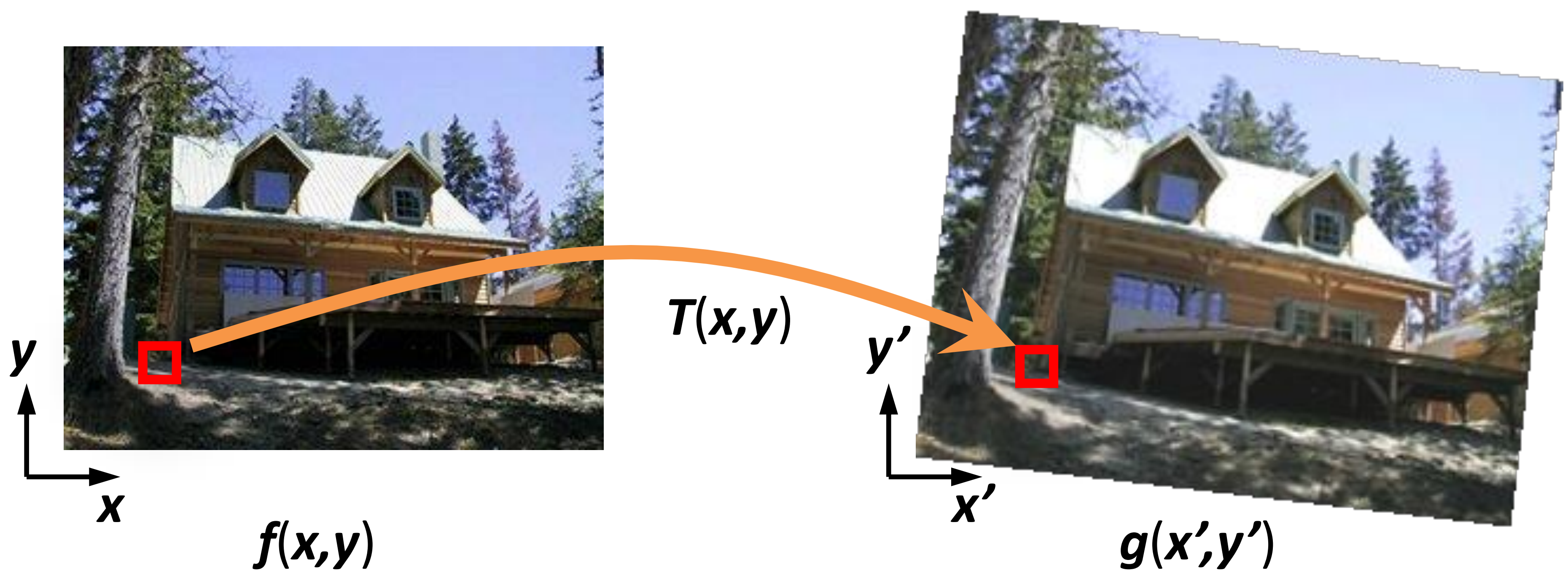
Implementing image warping

- Given a coordinate xform $(\boldsymbol{x'},\boldsymbol{y'}) = \boldsymbol{T}(\boldsymbol{x},\boldsymbol{y})$ and a source image $\boldsymbol{f}(\boldsymbol{x},\boldsymbol{y})$, how do we compute an xformed image $\boldsymbol{g}(\boldsymbol{x'},\boldsymbol{y'}) = \boldsymbol{f}(\boldsymbol{T}(\boldsymbol{x},\boldsymbol{y}))$?



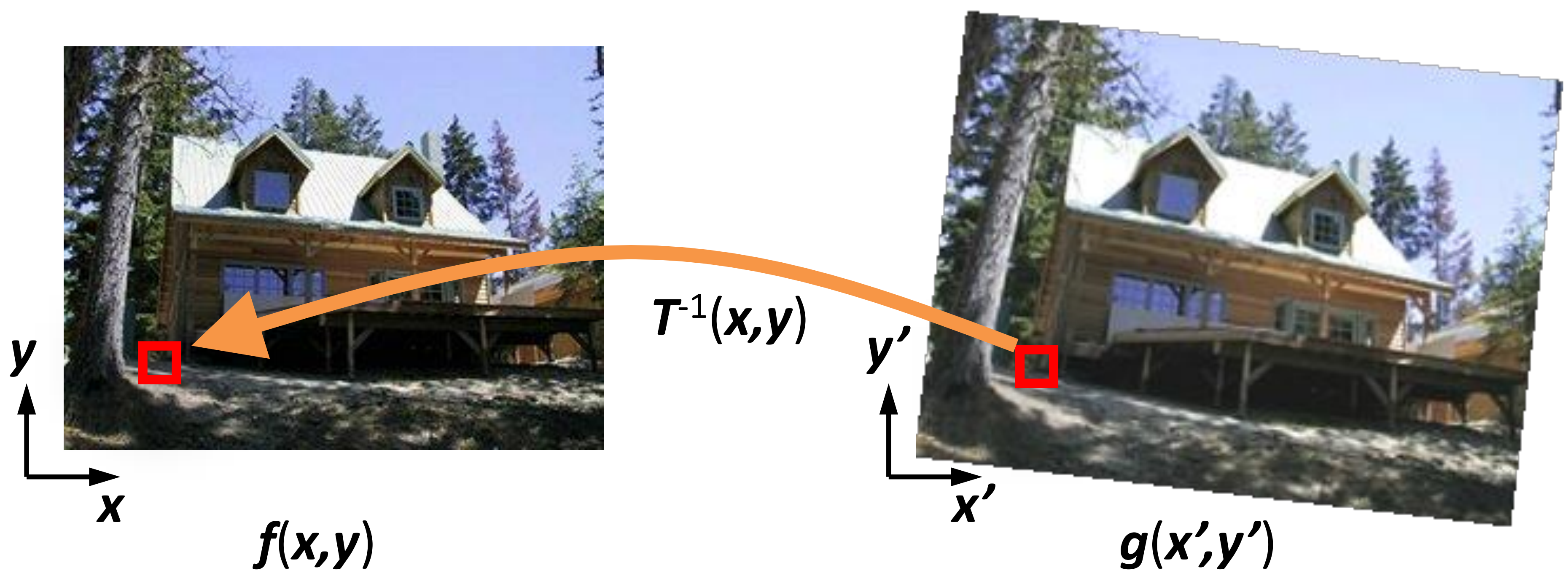
Forward Warping

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



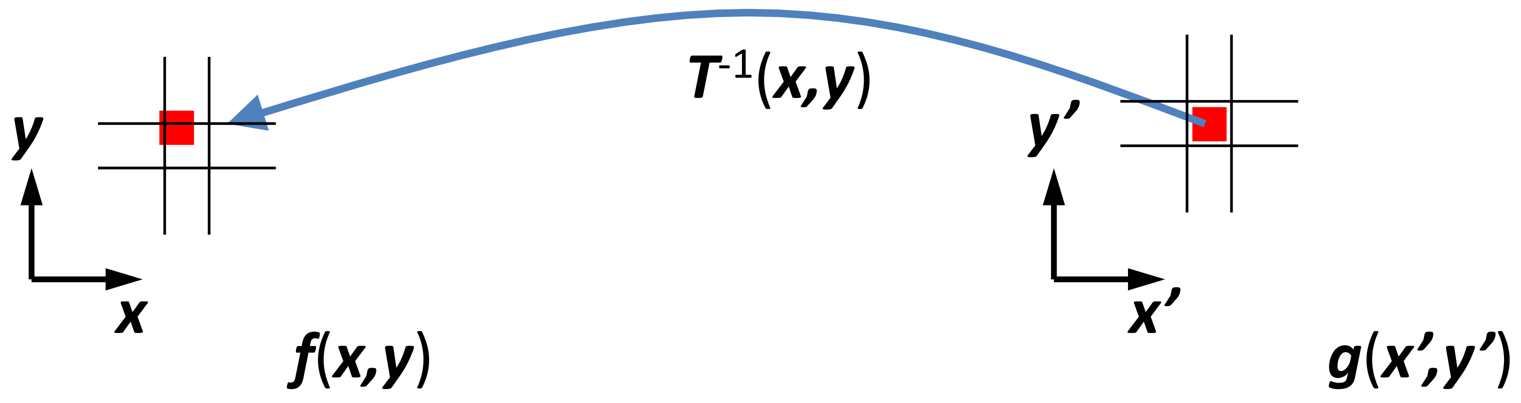
Inverse Warping

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



Inverse Warping

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



Blending

- We’ve aligned the images – now what?

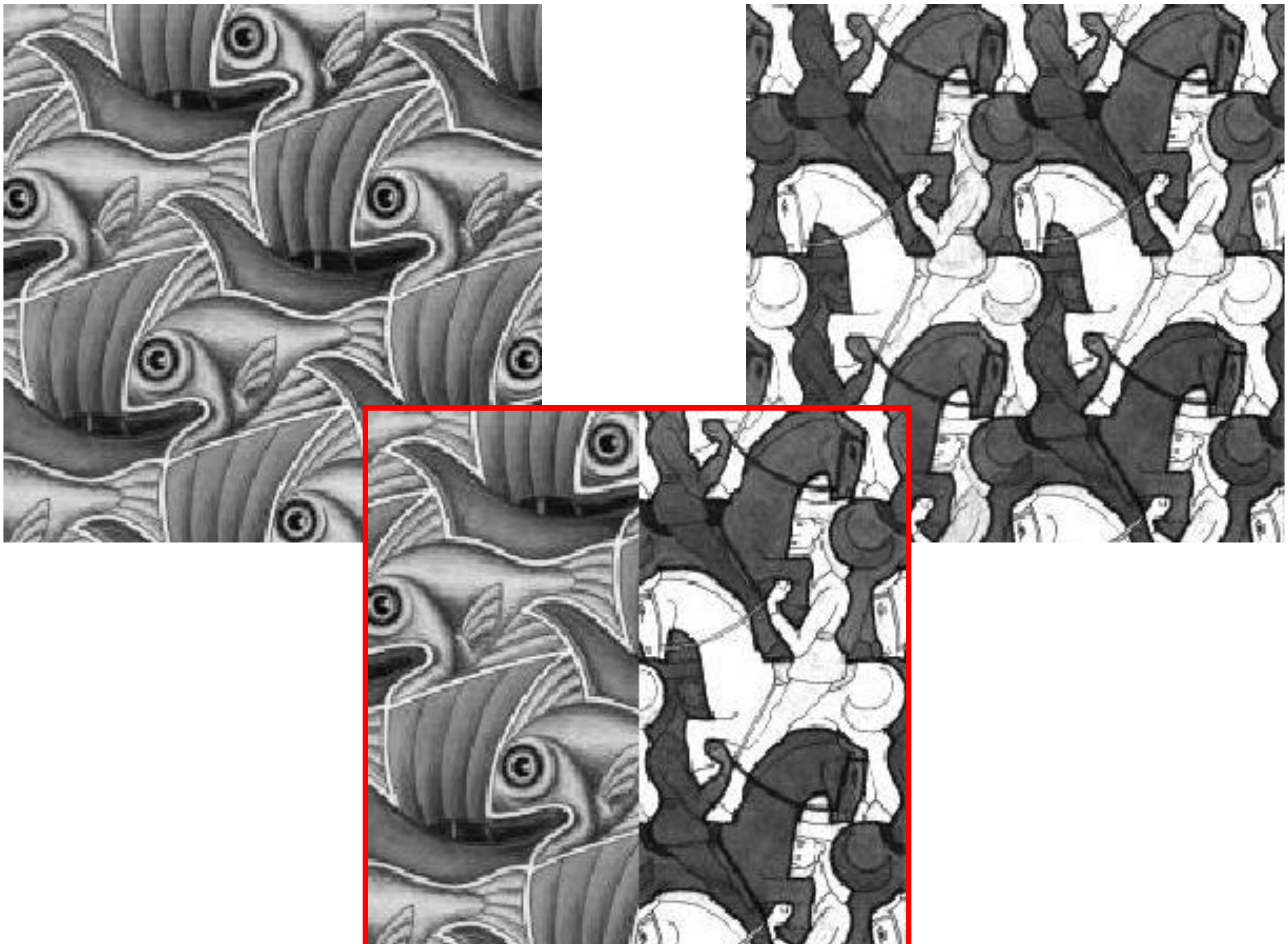
Slide credit: Noah Snavely

Blending

- Want to seamlessly blend them together

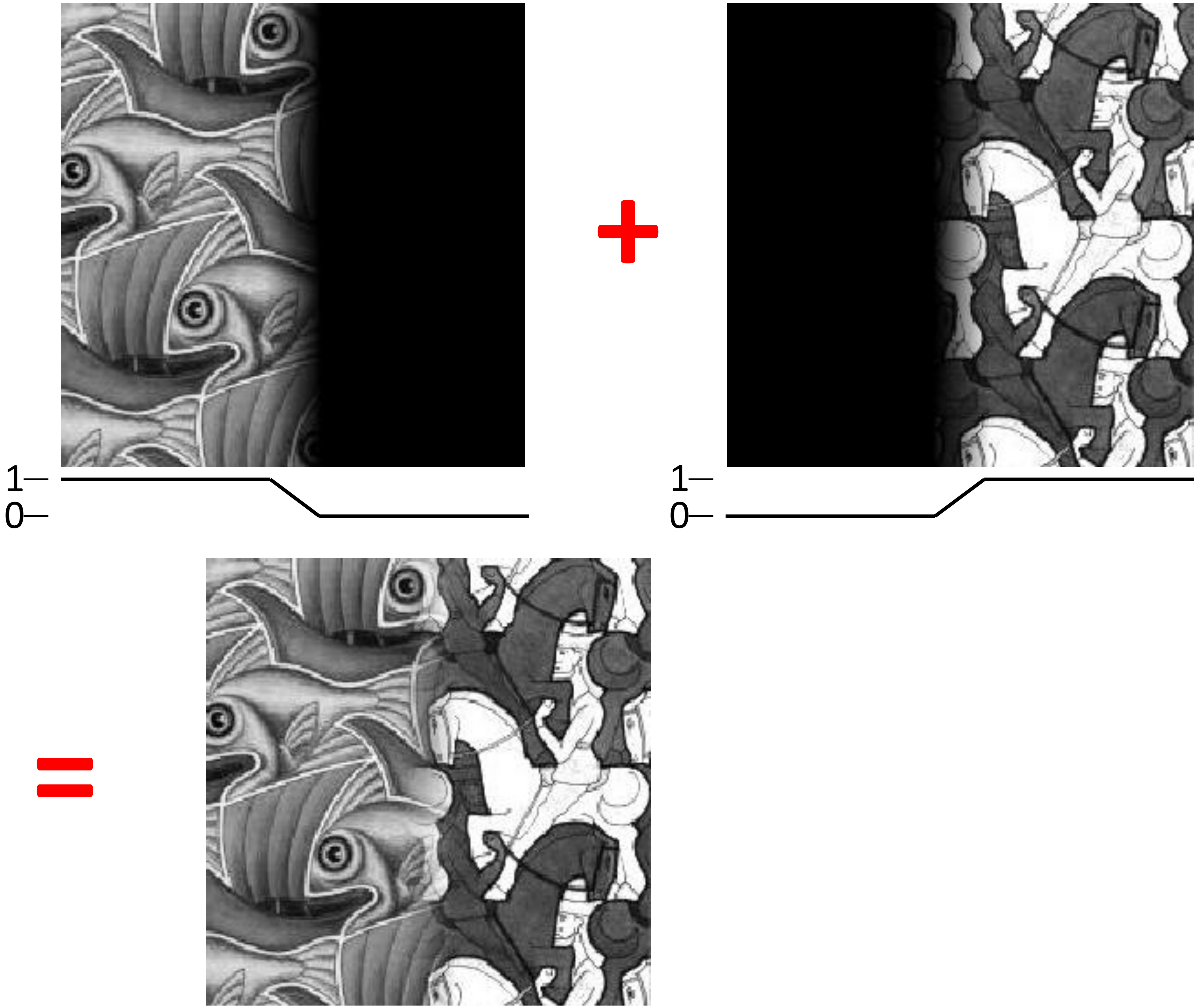
Slide credit: Noah Snavely

Image Blending



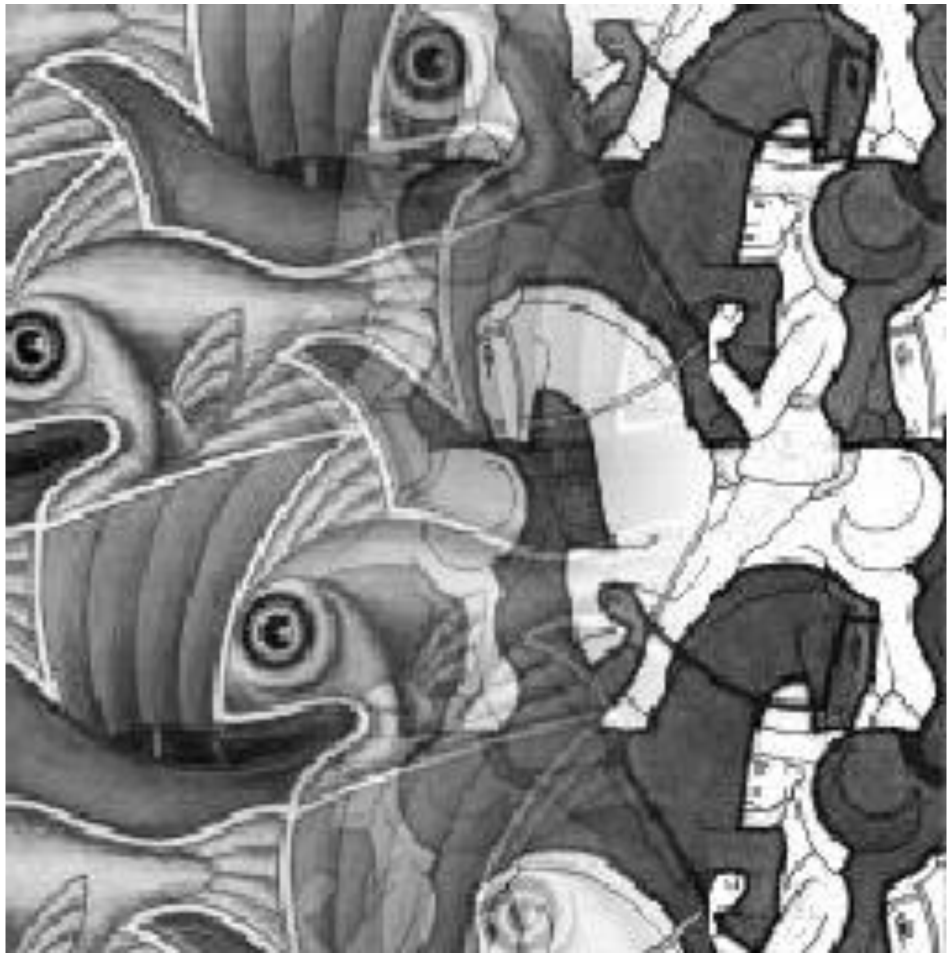
Slide credit: Noah Snavely

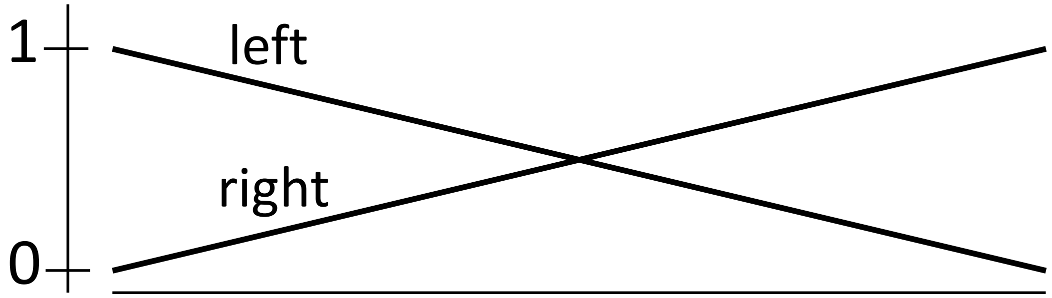
Feathering

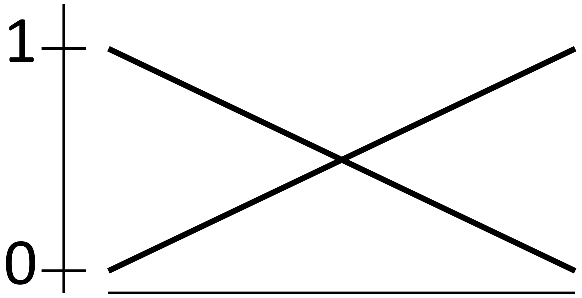


Slide credit: Noah Snavely

Effect of window size

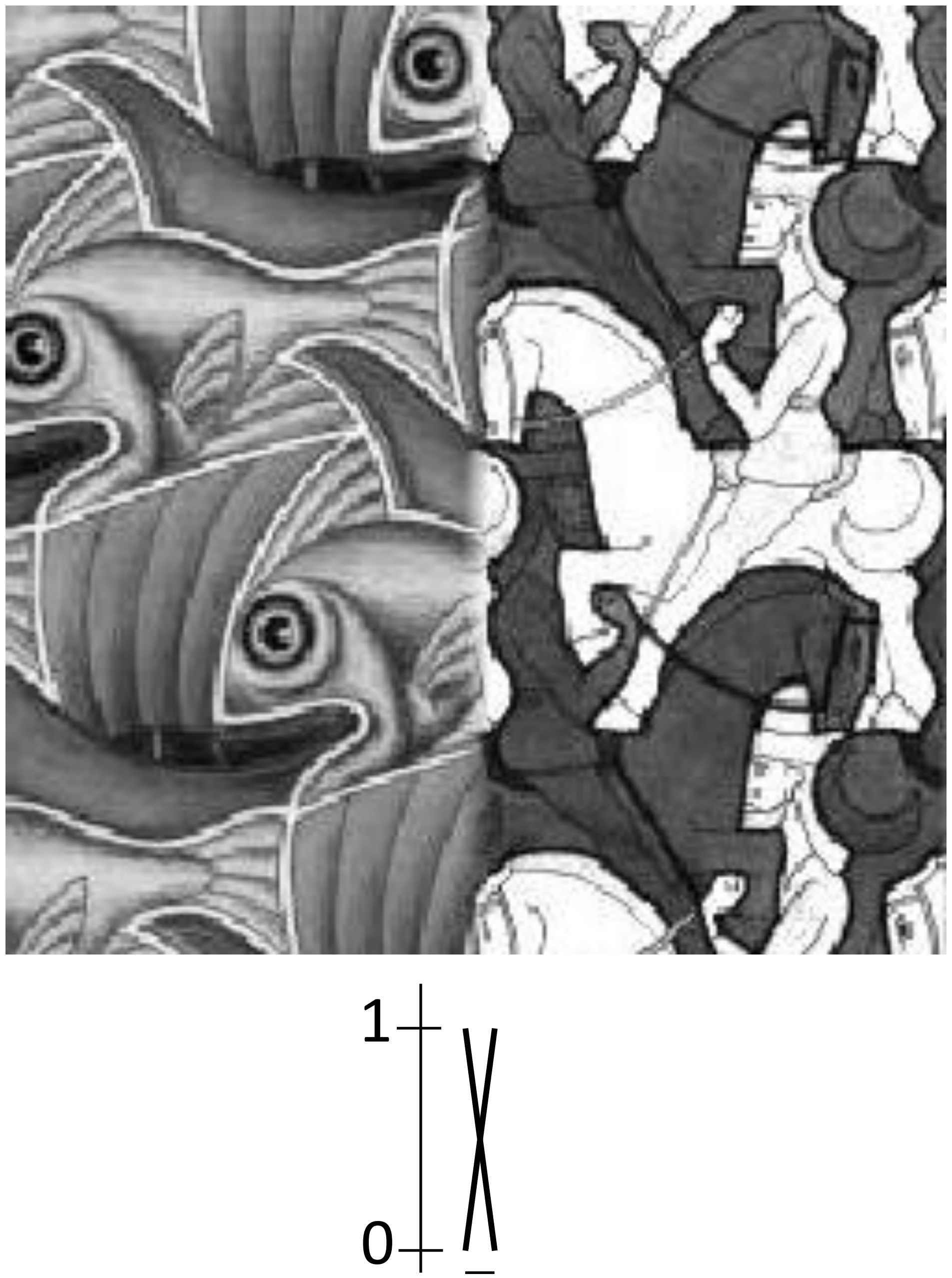


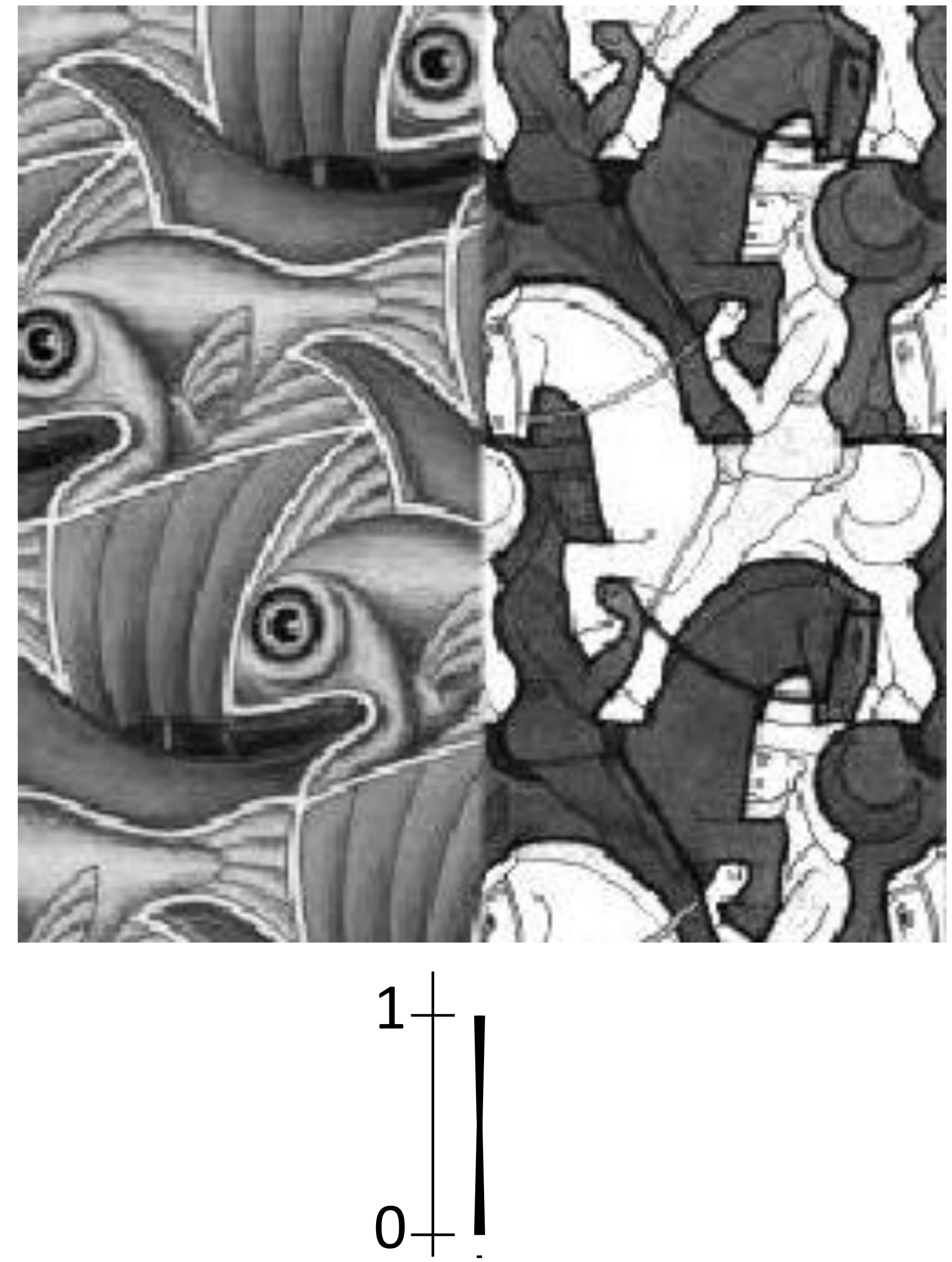




Slide credit: Noah Snavely

Effect of window size





Slide credit: Noah Snavely

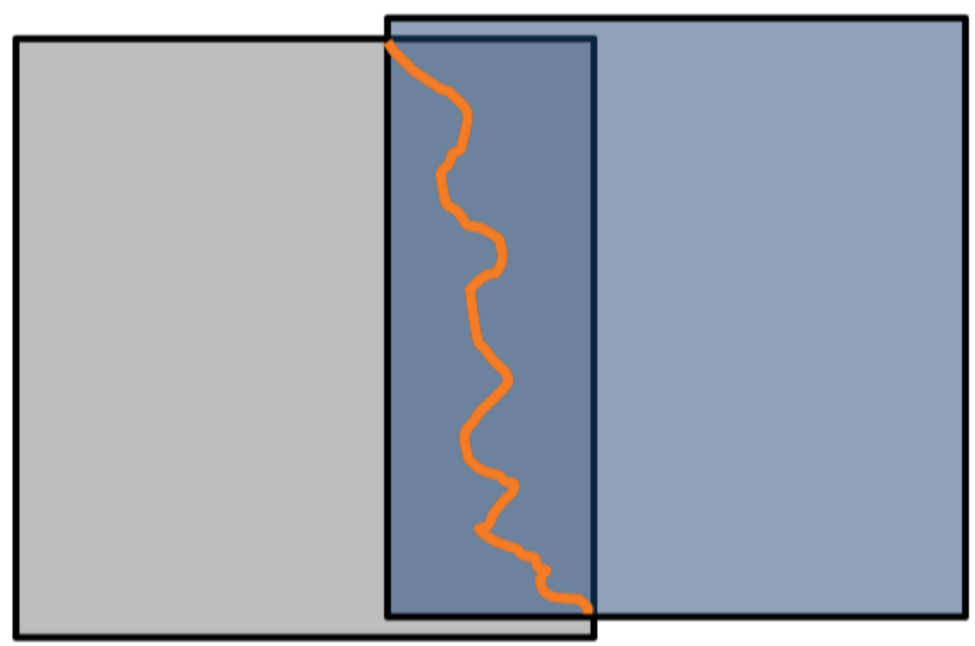
Blending

Moving object, simple blending => blur

Slide credit: Davis '98

Blending

Instead of blending high frequencies along a straight line, blend along line of minimum differences in image intensities



Slide credit: Olga Russakovsky

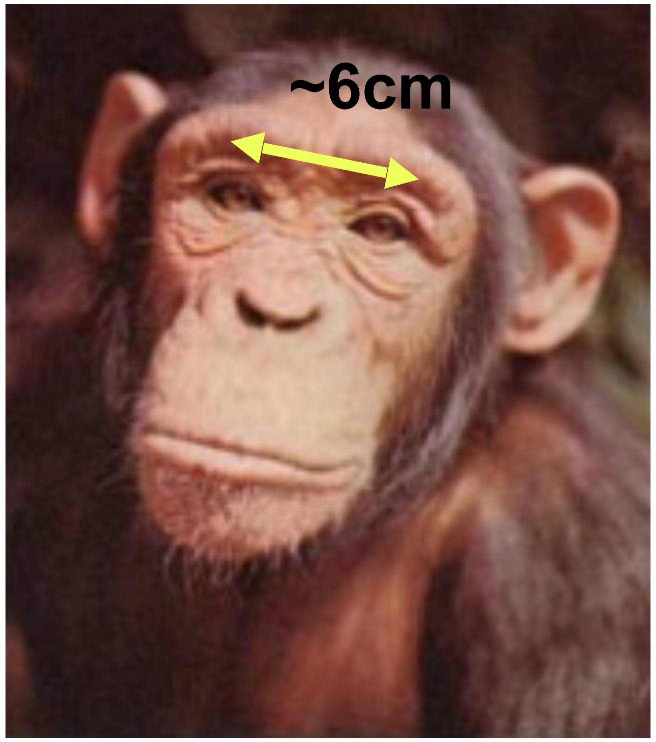
Blending

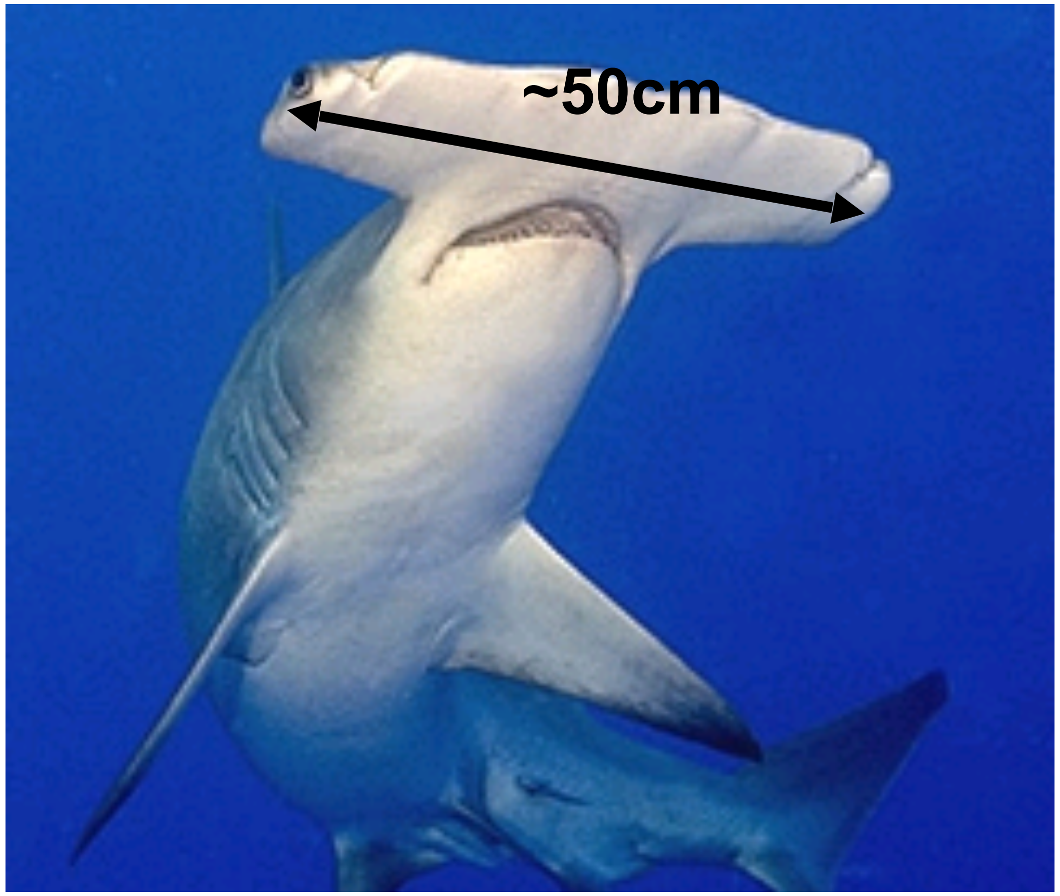
Minimum-cost cut $\Rightarrow$ no blur

Slide credit: Davis ‘98

Stereo

Stereo vision





Slide credit: Antonio Torralba

Why not put our
second eye here?

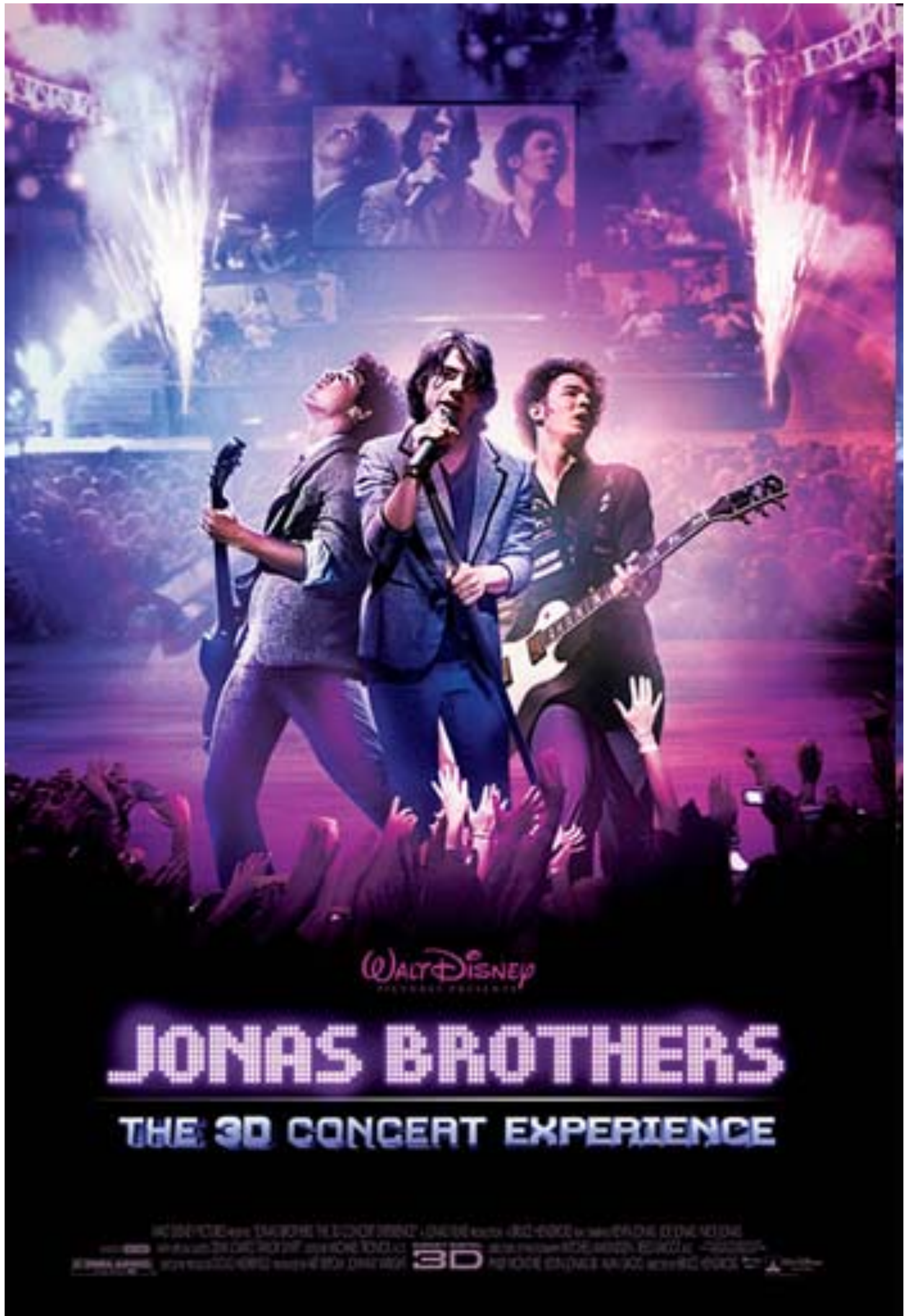
Stereoscopes: A 19th Century Pastime

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

3D Movies


Walt Disney
JONAS BROTHERS
THE 3D CONCERT EXPERIENCE


MY BLOODY
VALENTINE
3D
GET YOUR HEART BROKEN
IN THEATERS JANUARY

Depth without objects

Random dot stereograms (Bela Julesz)

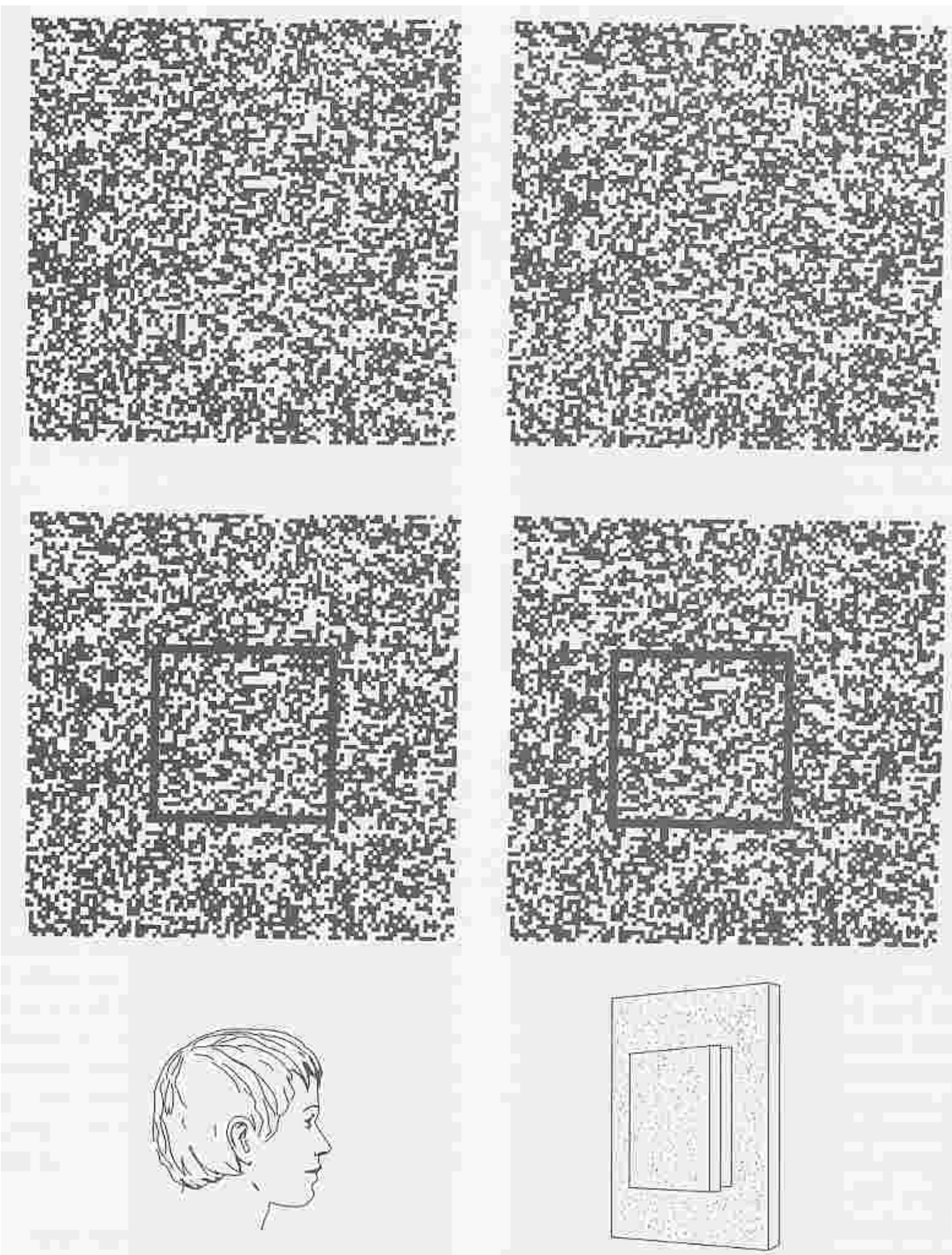


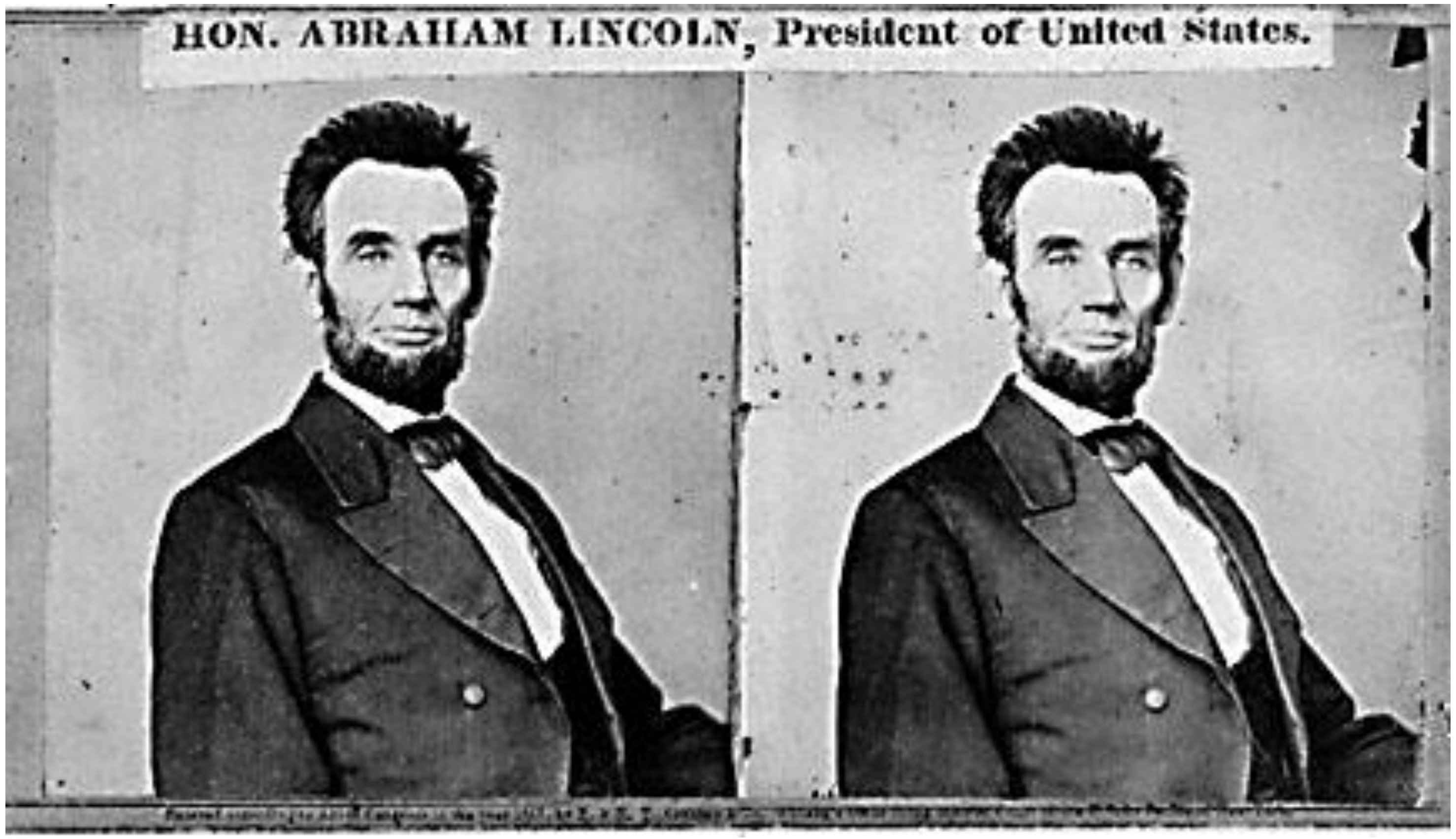
FIGURE 8.13

1	0	1	0	1	0	0	1	0	1
1	0	0	1	0	1	0	1	0	0
0	0	1	1	0	1	1	0	1	0
0	1	0	Y	A	A	B	B	0	1
1	1	1	X	B	A	B	A	0	1
0	0	1	X	A	A	B	A	1	0
1	1	1	Y	B	B	A	B	0	1
1	0	0	1	1	0	1	1	0	1
1	1	0	0	1	1	0	1	1	1
0	1	0	0	0	1	1	1	1	0

1	0	1	0	1	0	0	1	0	1
1	0	0	1	0	1	0	1	0	0
0	0	1	1	0	1	1	0	1	0
0	1	0	A	A	B	B	X	0	1
1	1	1	B	A	B	A	Y	0	1
0	0	1	A	A	B	A	Y	1	0
1	1	1	B	B	A	B	X	0	1
1	0	0	1	1	0	1	1	0	1
1	1	0	0	1	1	0	1	1	1
0	1	0	0	0	1	1	1	1	0

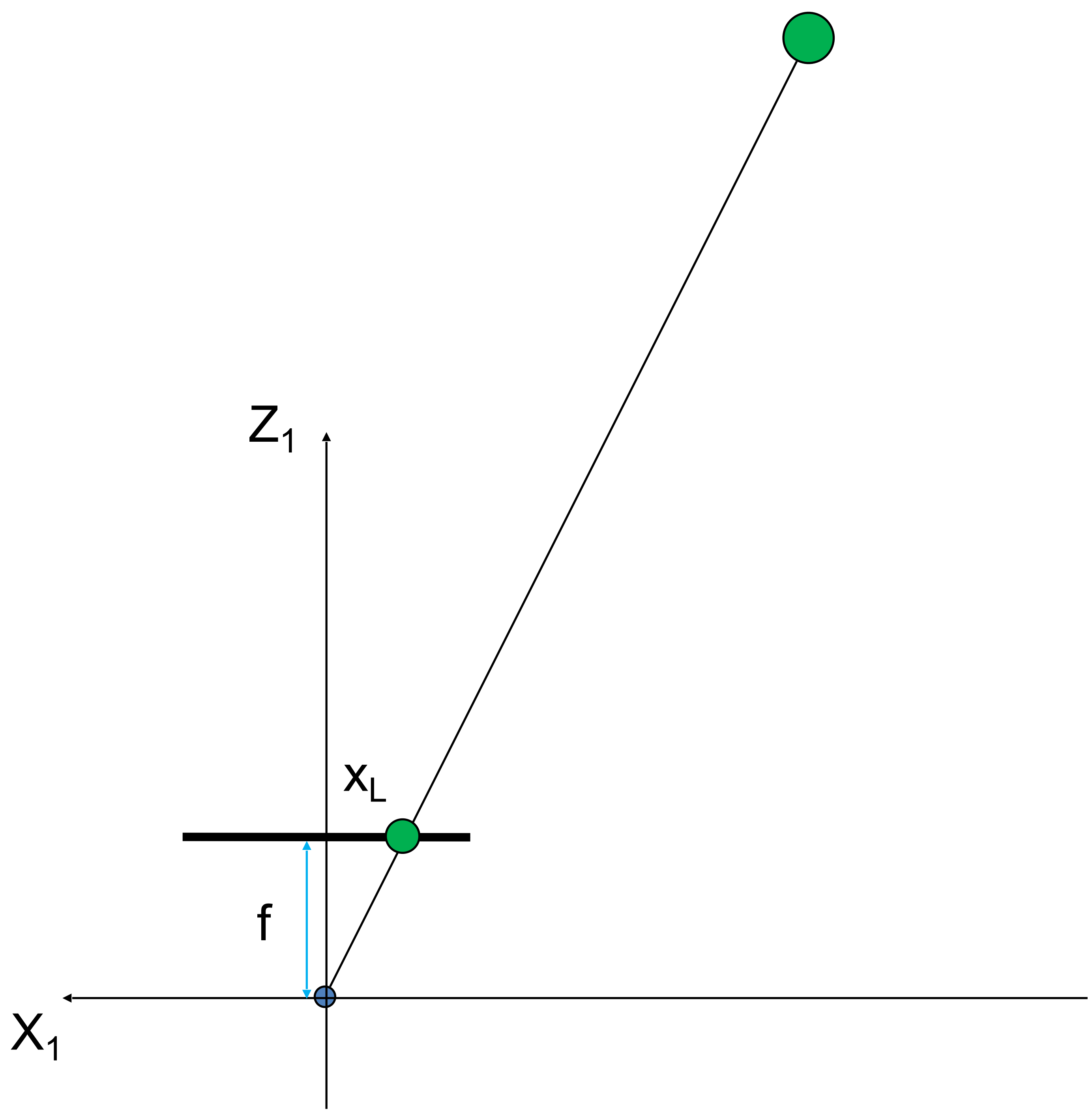
Julesz, 1971

Stereo



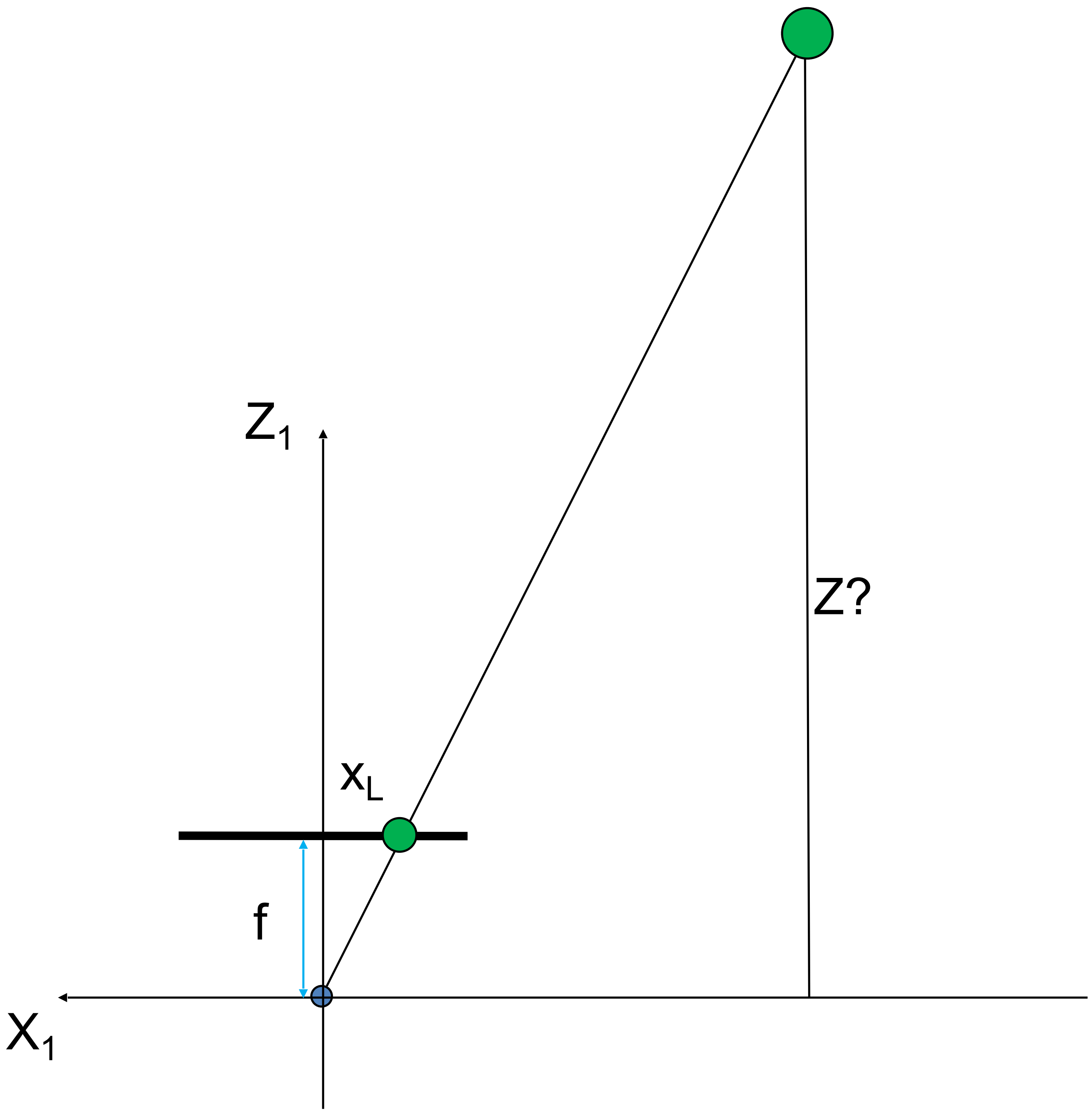
- Given two images from different viewpoints
  - How can we compute the depth of each point in the image?
  - Based on *how much each pixel moves* between the two images

Geometry for a simple stereo system



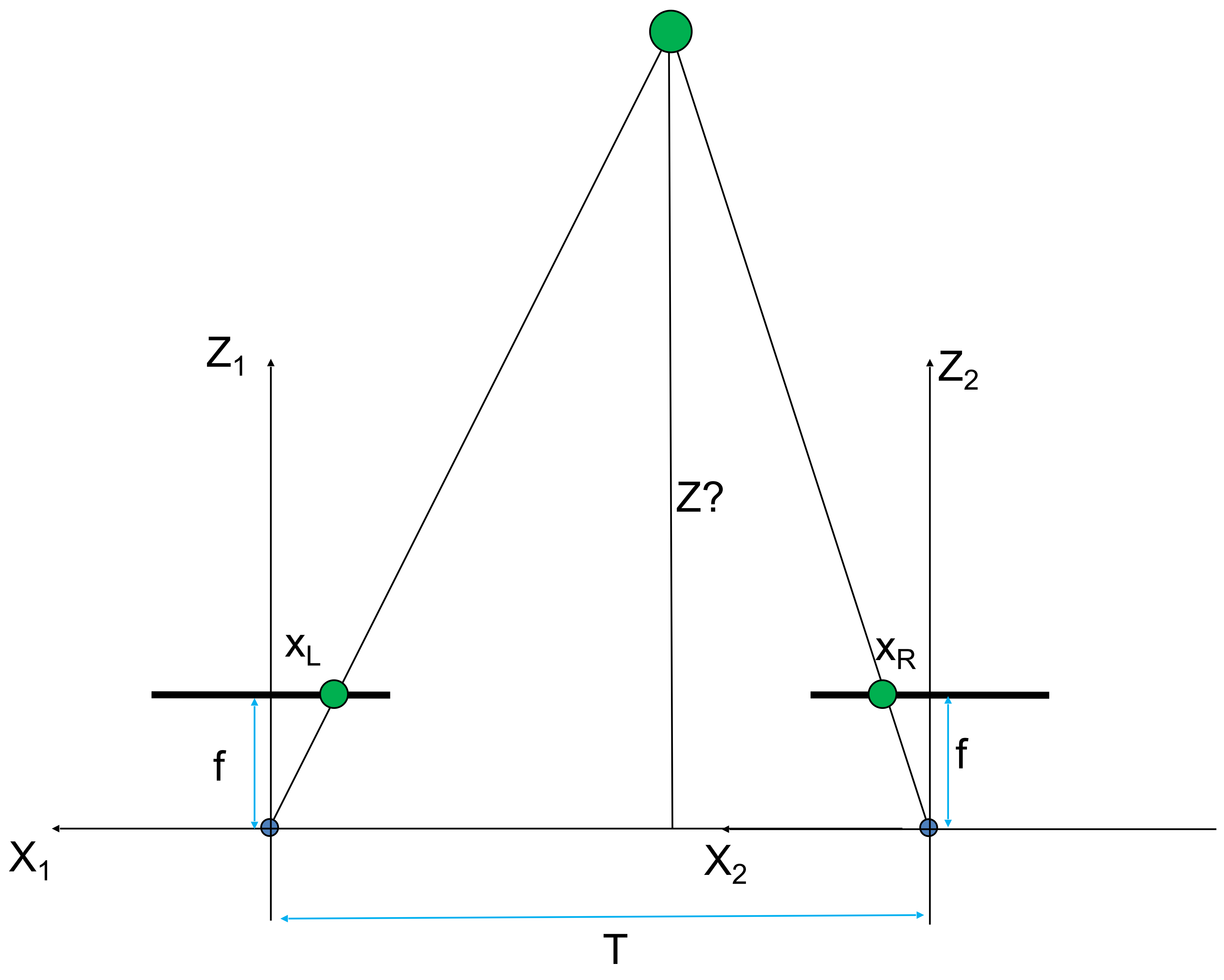
Slide credit: Antonio Torralba

Geometry for a simple stereo system



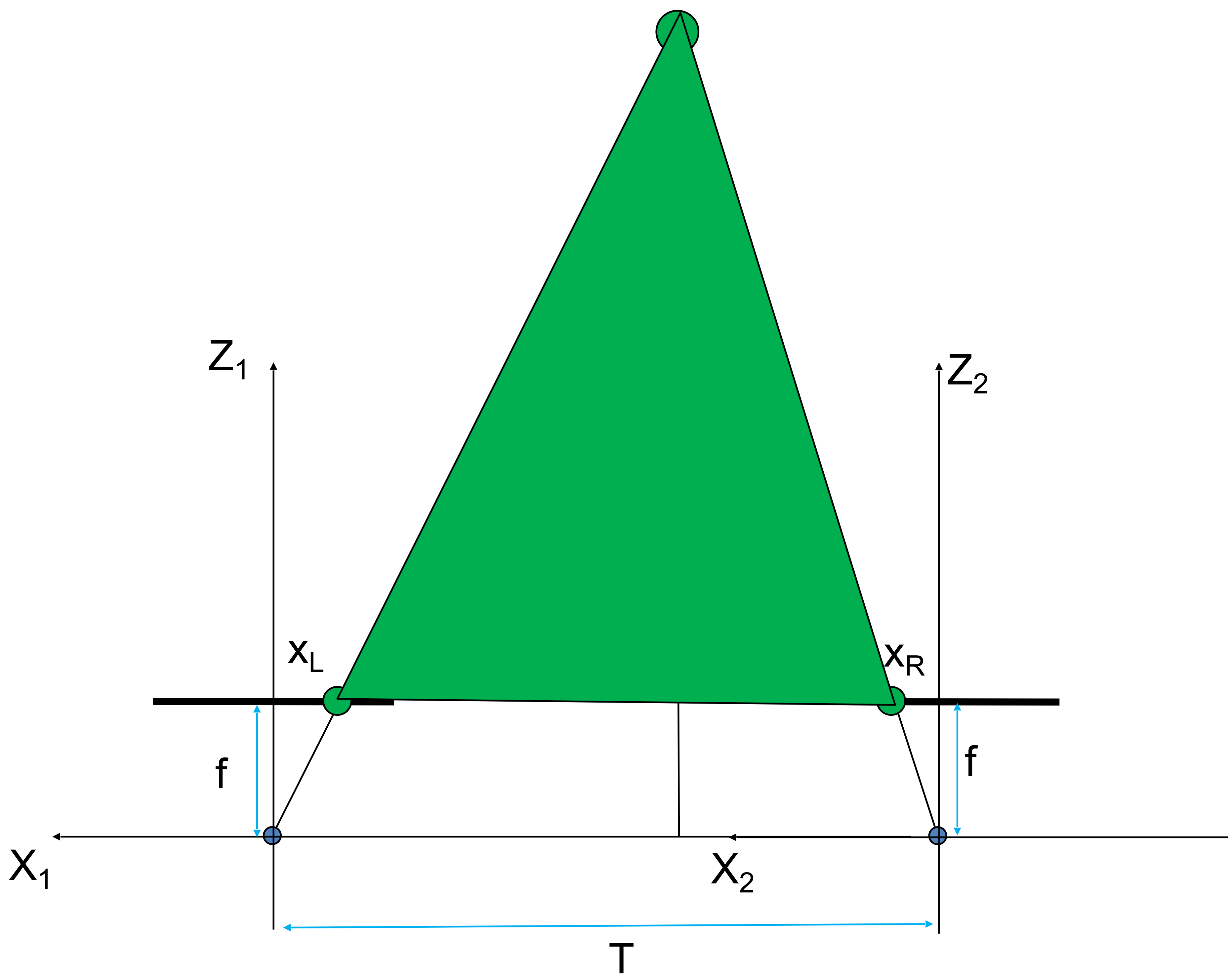
Slide credit: Antonio Torralba

Geometry for a simple stereo system



Slide credit: Antonio Torralba

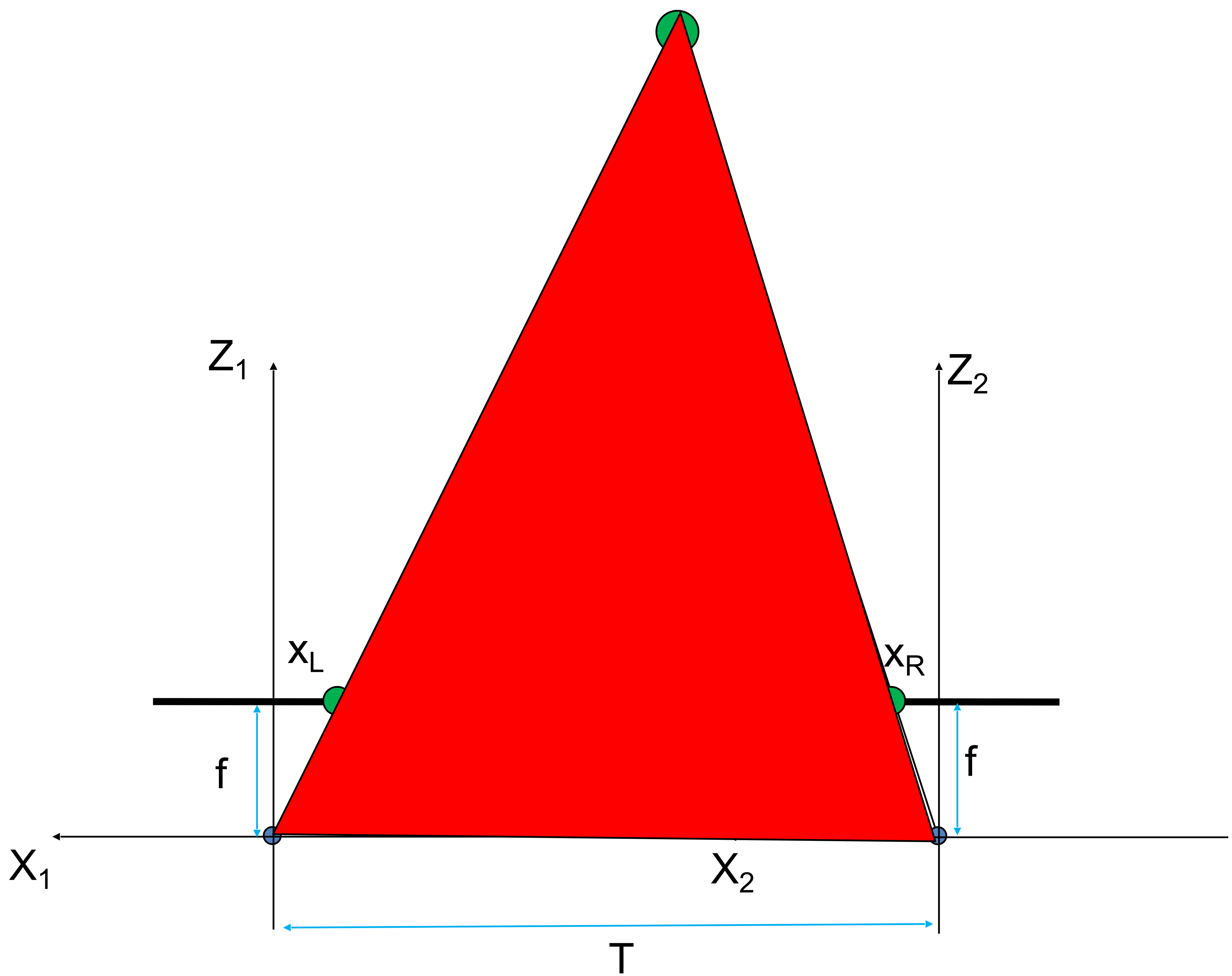
Geometry for a simple stereo system



Similar triangles

Slide credit: Antonio Torralba

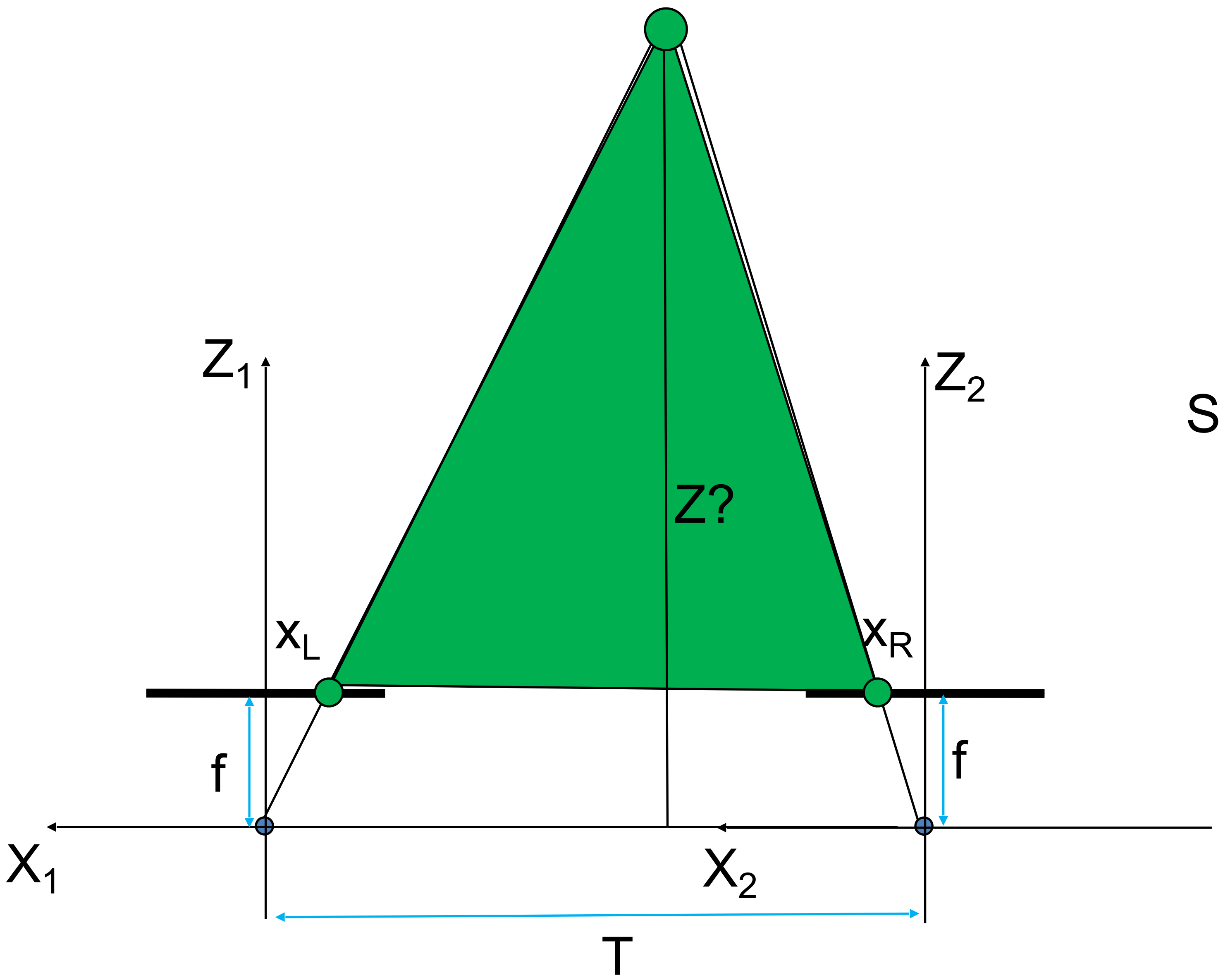
Geometry for a simple stereo system



Similar triangles

Slide credit: Antonio Torralba

Geometry for a simple stereo system

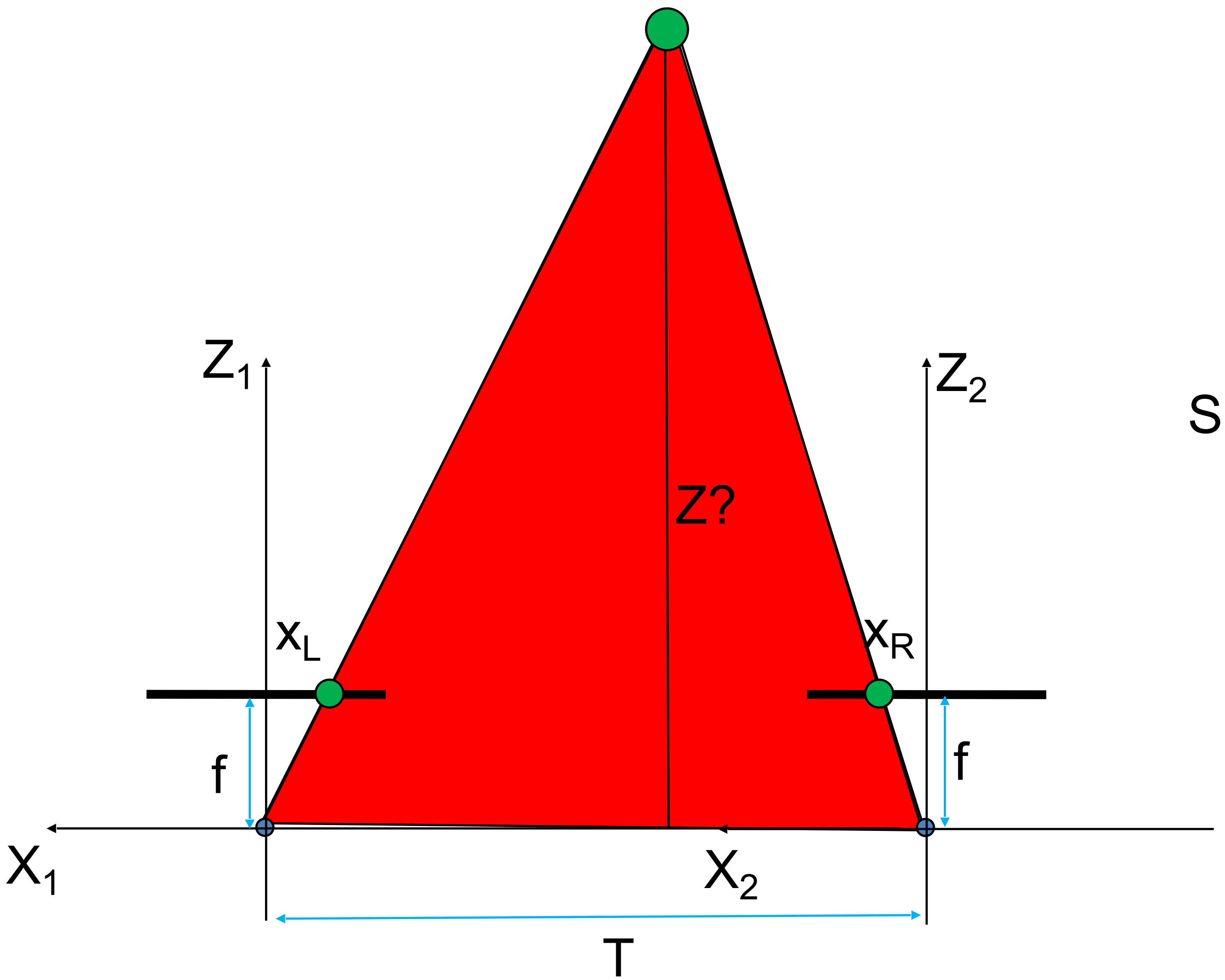


Similar triangles:

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

Slide credit: Antonio Torralba

Geometry for a simple stereo system

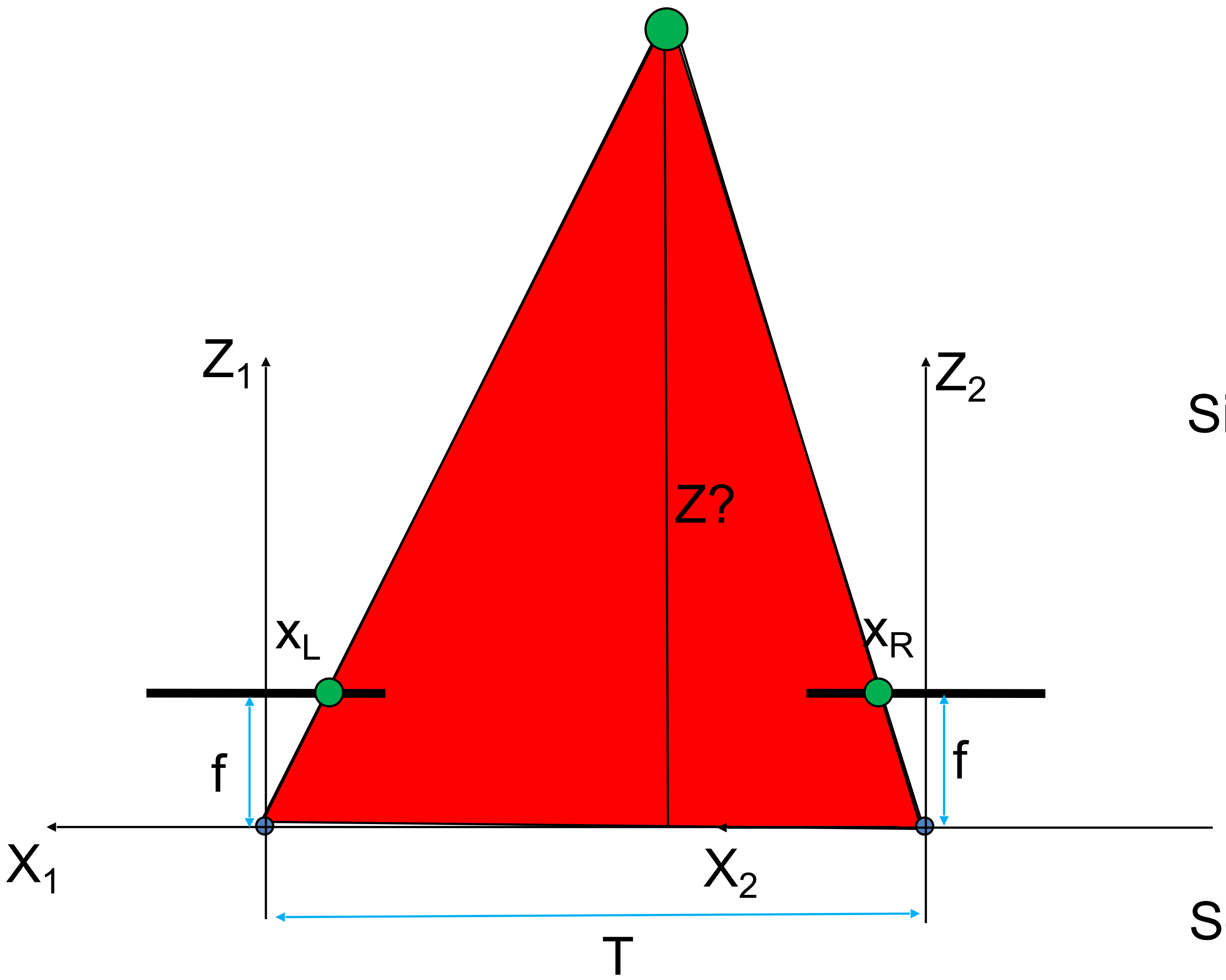


Similar triangles:

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

Slide credit: Antonio Torralba

Geometry for a simple stereo system



Similar triangles:

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

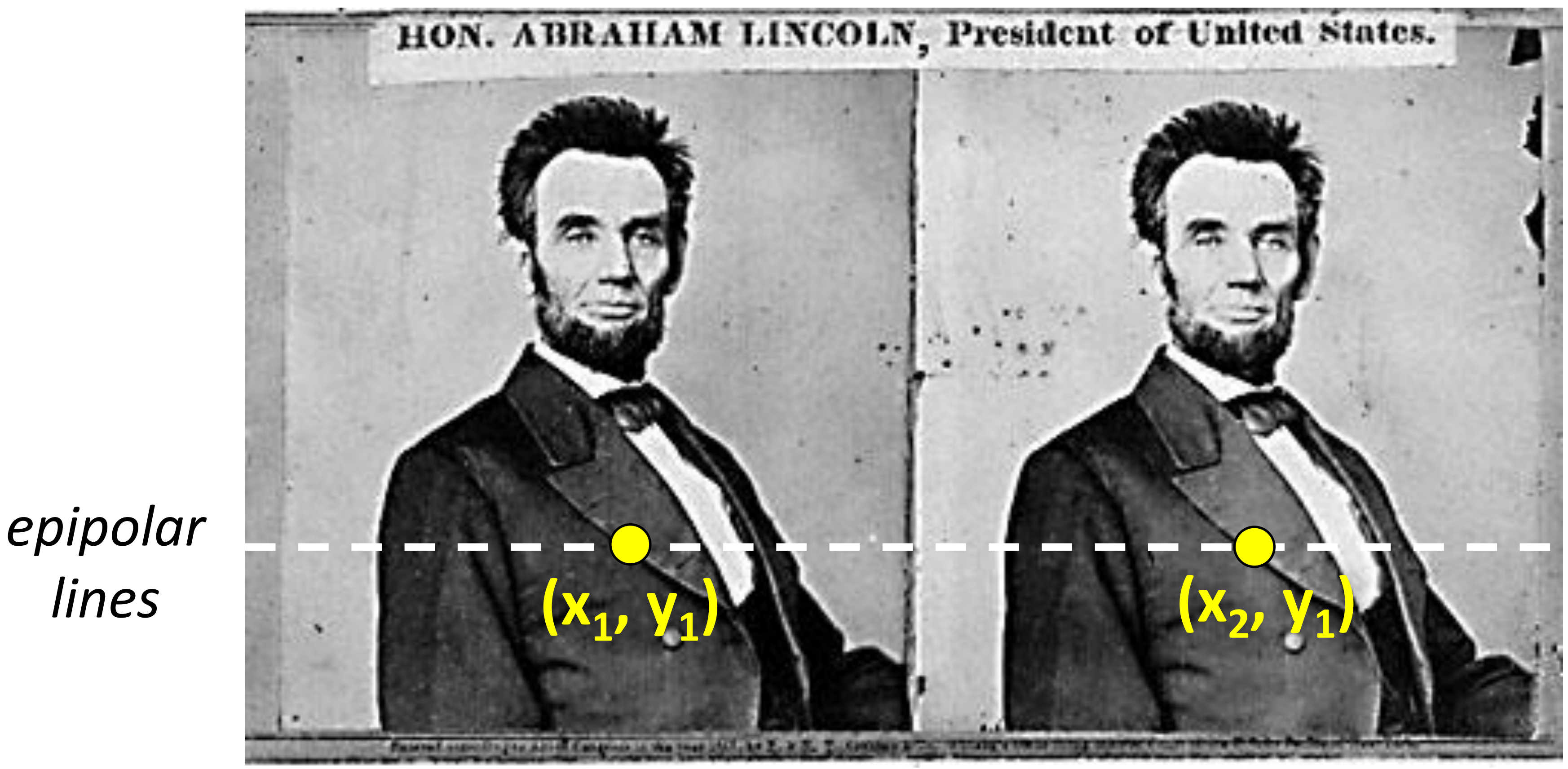
Solving for Z:

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

Disparity

Slide credit: Antonio Torralba

Epipolar geometry

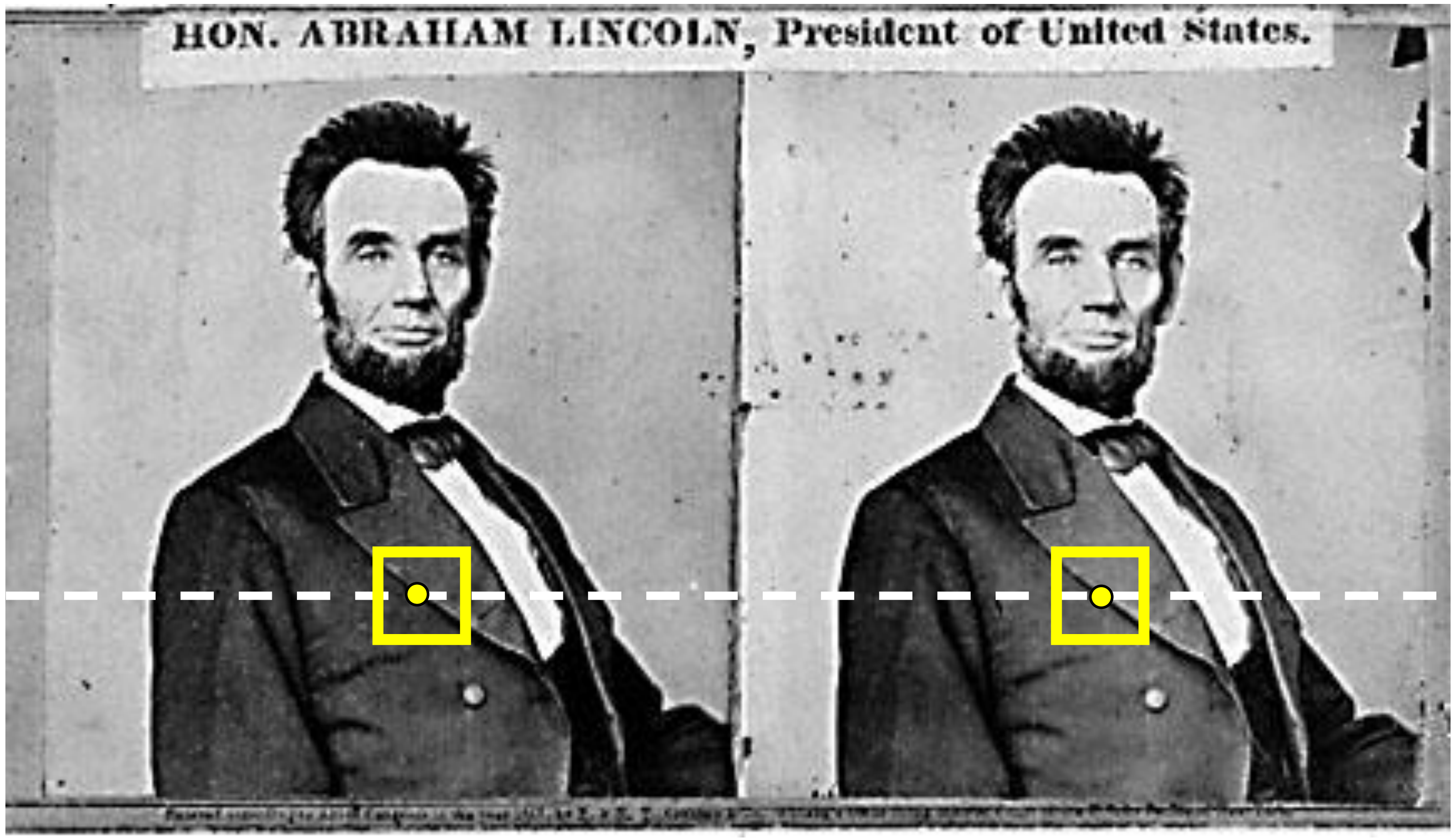


Two images captured by a purely horizontal translating camera
(*rectified* stereo pair)

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

Slide credit: Noah Snavely

Your basic stereo algorithm



For each epipolar line

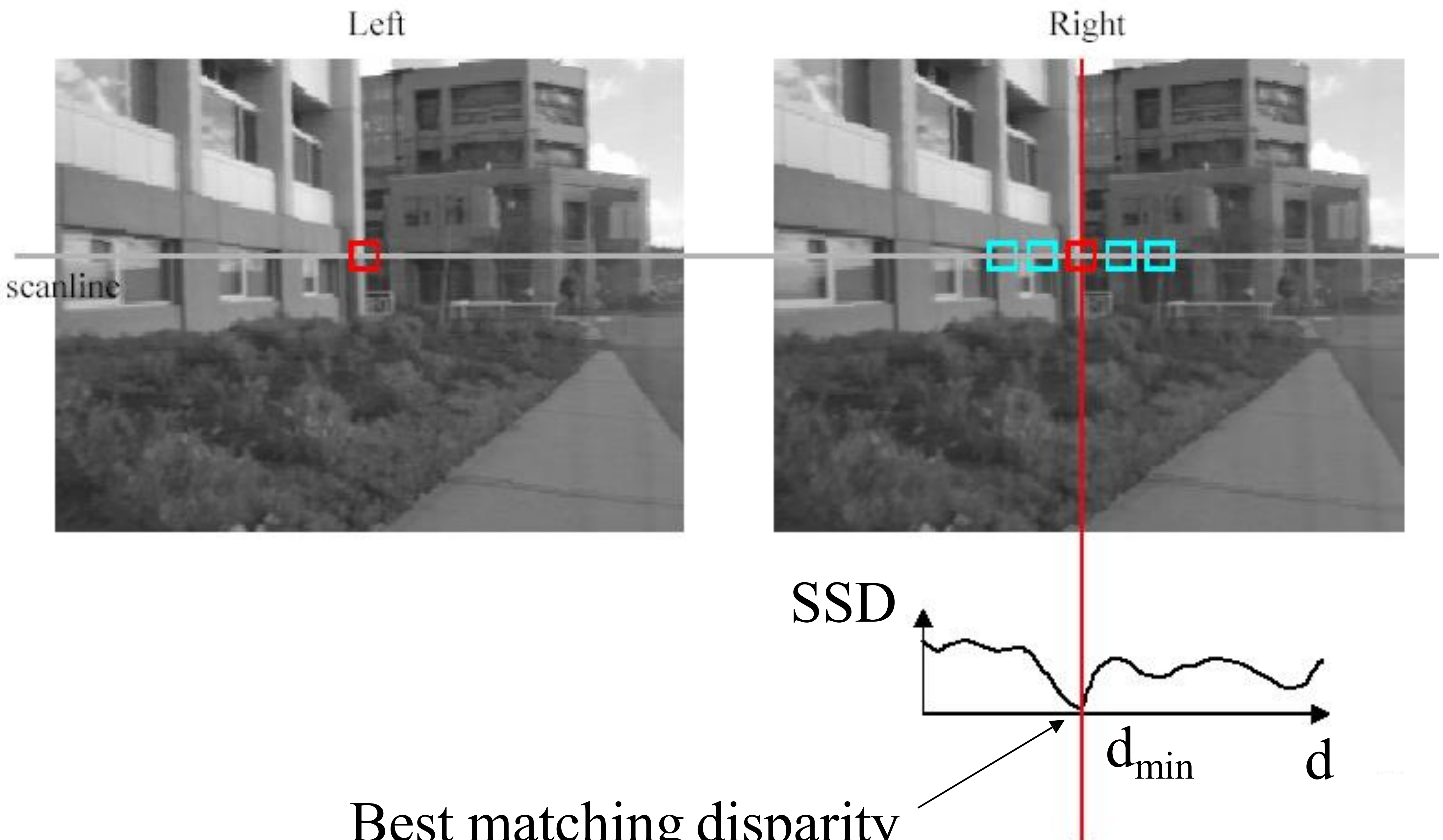
For each pixel in the left image

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

Improvement: match ***windows***

Slide credit: Noah Snavely

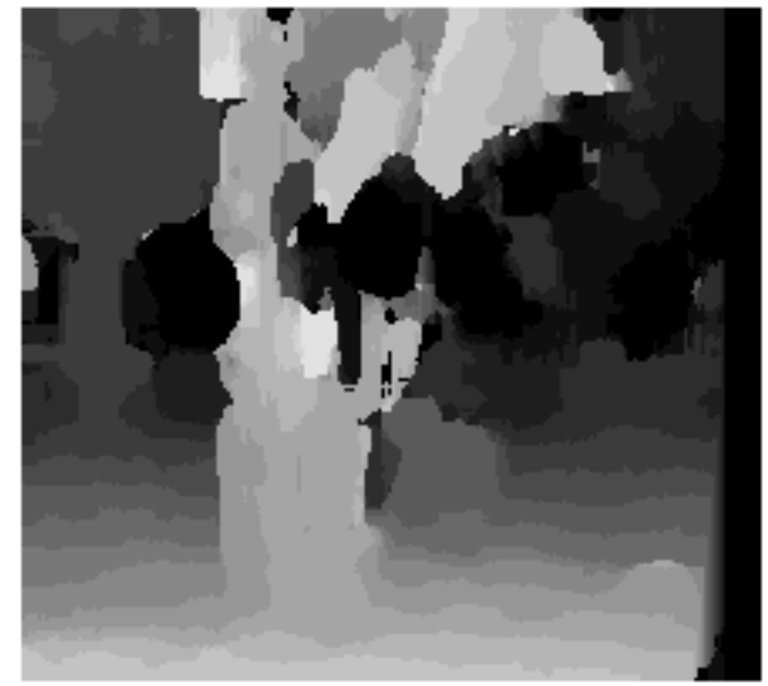
Stereo matching based on SSD



Slide credit: Noah Snavely

Window size

W = 3



W = 20

Effect of window size
- – Smaller window
  - +
  - •
- – Larger window
  - +
  - •

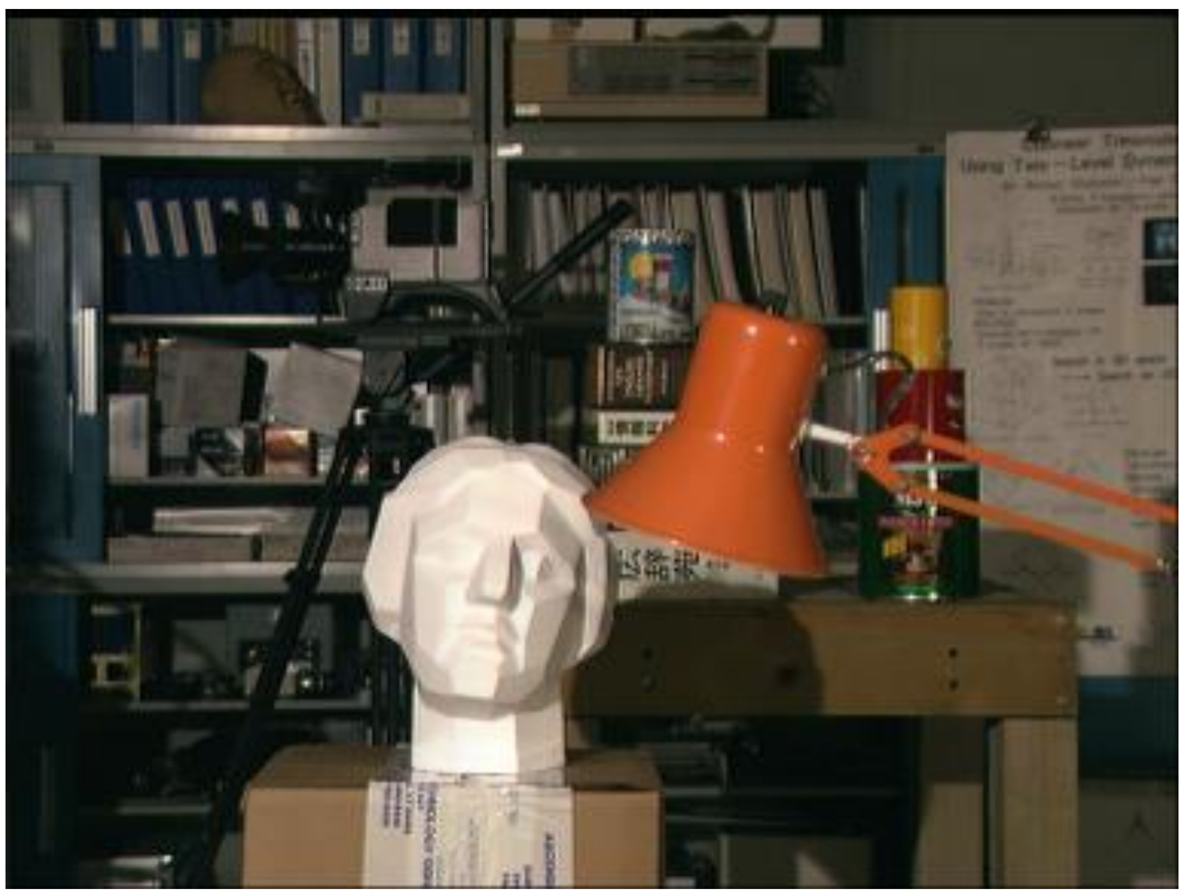
Better results with *adaptive window*

- T. Kanade and M. Okutomi, *A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment*,, Proc. International Conference on Robotics and Automation, 1991.
- D. Scharstein and R. Szeliski. Stereo matching with nonlinear diffusion. International Journal of Computer Vision, 28(2):155-174, July 1998

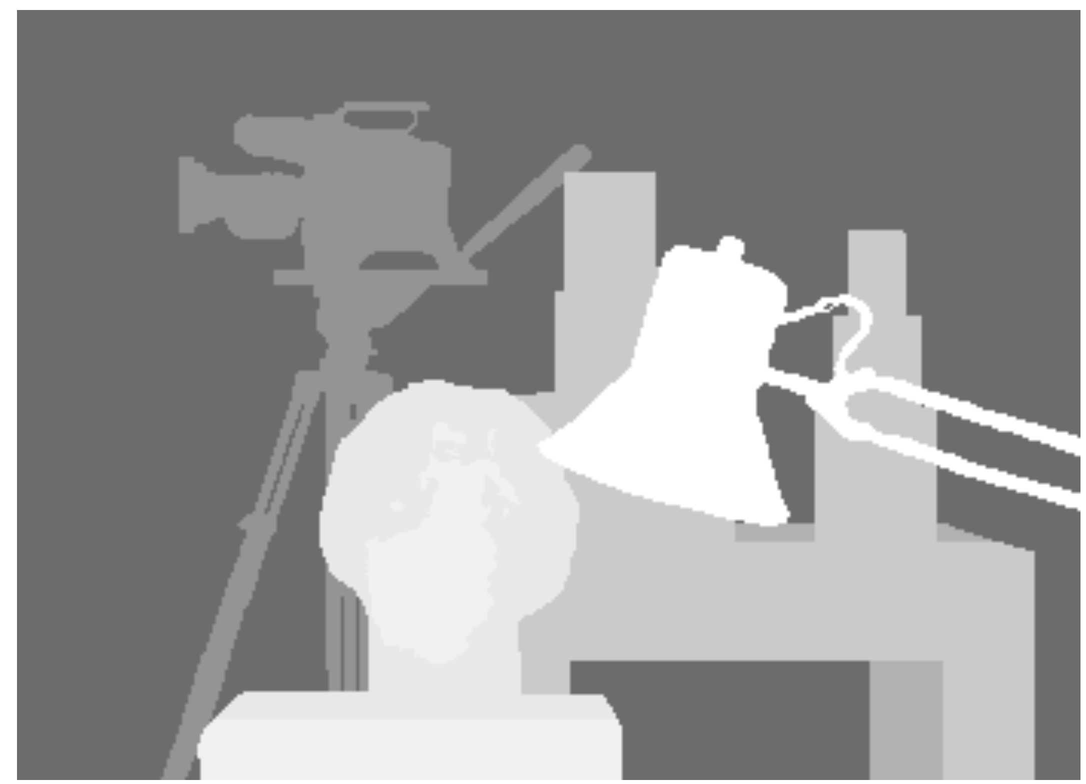
Slide credit: Noah Snavely

Stereo results

– Data from University of Tsukuba
– Similar results on other images without ground truth



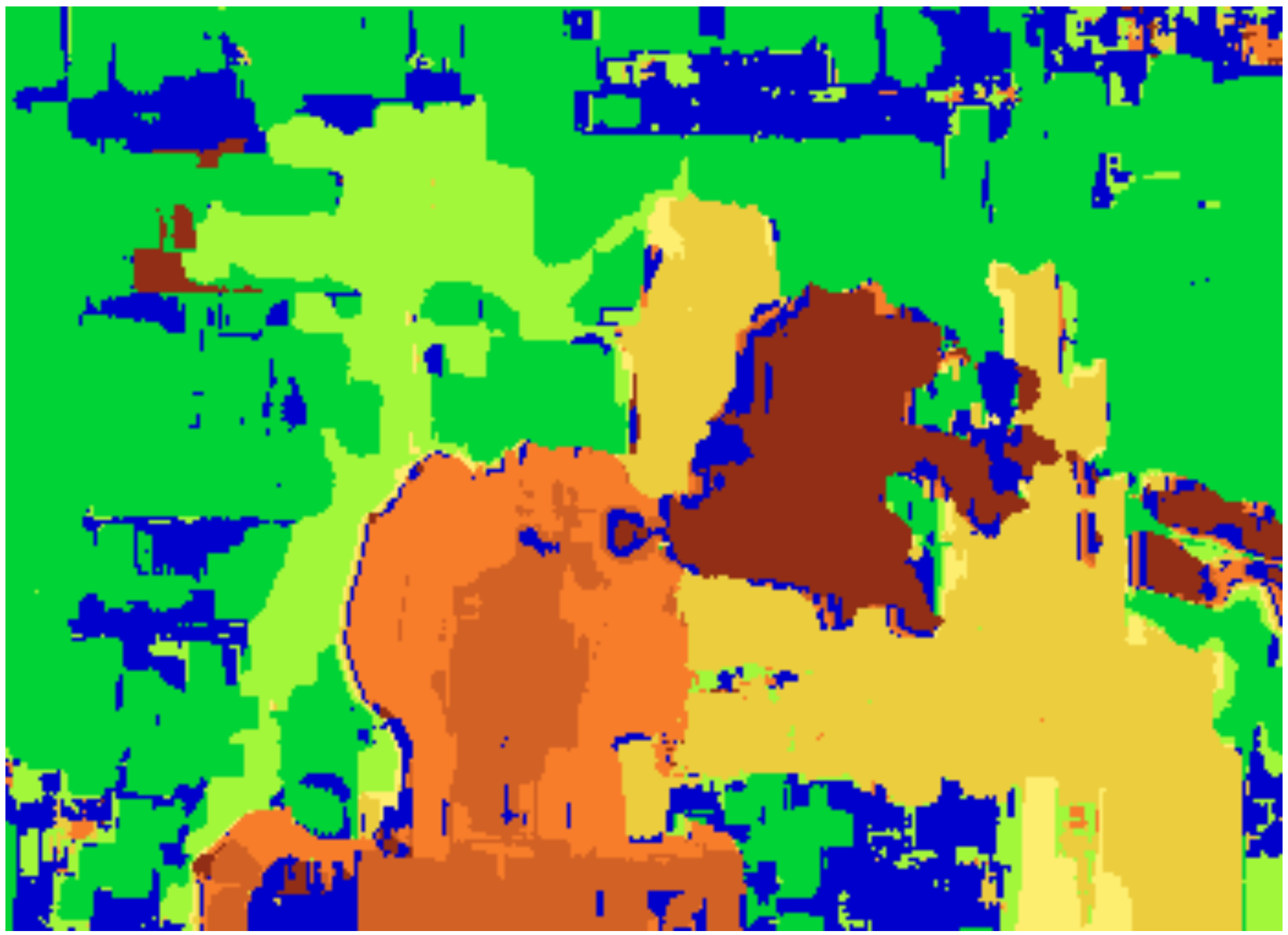
Scene



Ground truth

Slide credit: Noah Snavely

Results with window search



Window-based matching
(best window size)

Ground truth

Slide credit: Noah Snavely

Better methods exist...

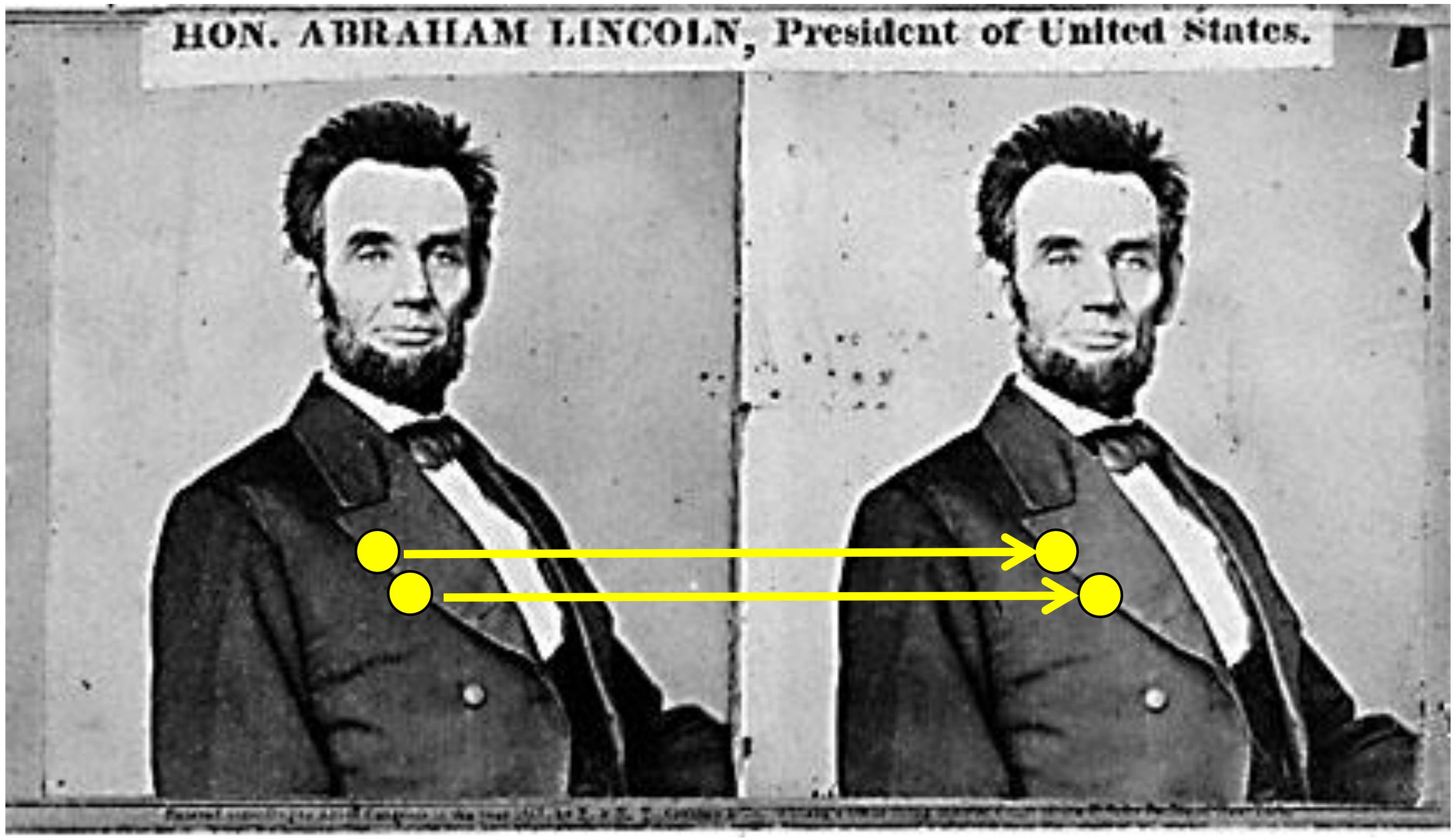
State of the art method

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

Ground truth

For the latest and greatest: http://www.middlebury.edu/stereo/

Stereo as energy minimization



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

Slide credit: Noah Snavely

Stereo as energy minimization

- Find disparity map *d* that minimizes an energy function $E(d)$

- Simple pixel / window matching

$$E(d) = \sum_{(x,y)\in I} C(x, y, d(x, y))$$

$C(x, y, d(x, y))$ = SSD distance between windows *I*(*x*, *y*) and *J*(*x* + *d*(*x*,*y*), *y*)

Slide credit: Noah Snavely

Stereo as energy minimization

$I(x, y)$

$J(x, y)$

y = 141

$C(x, y, d)$; the *disparity space image* (DSI)

Slide credit: Noah Snavely

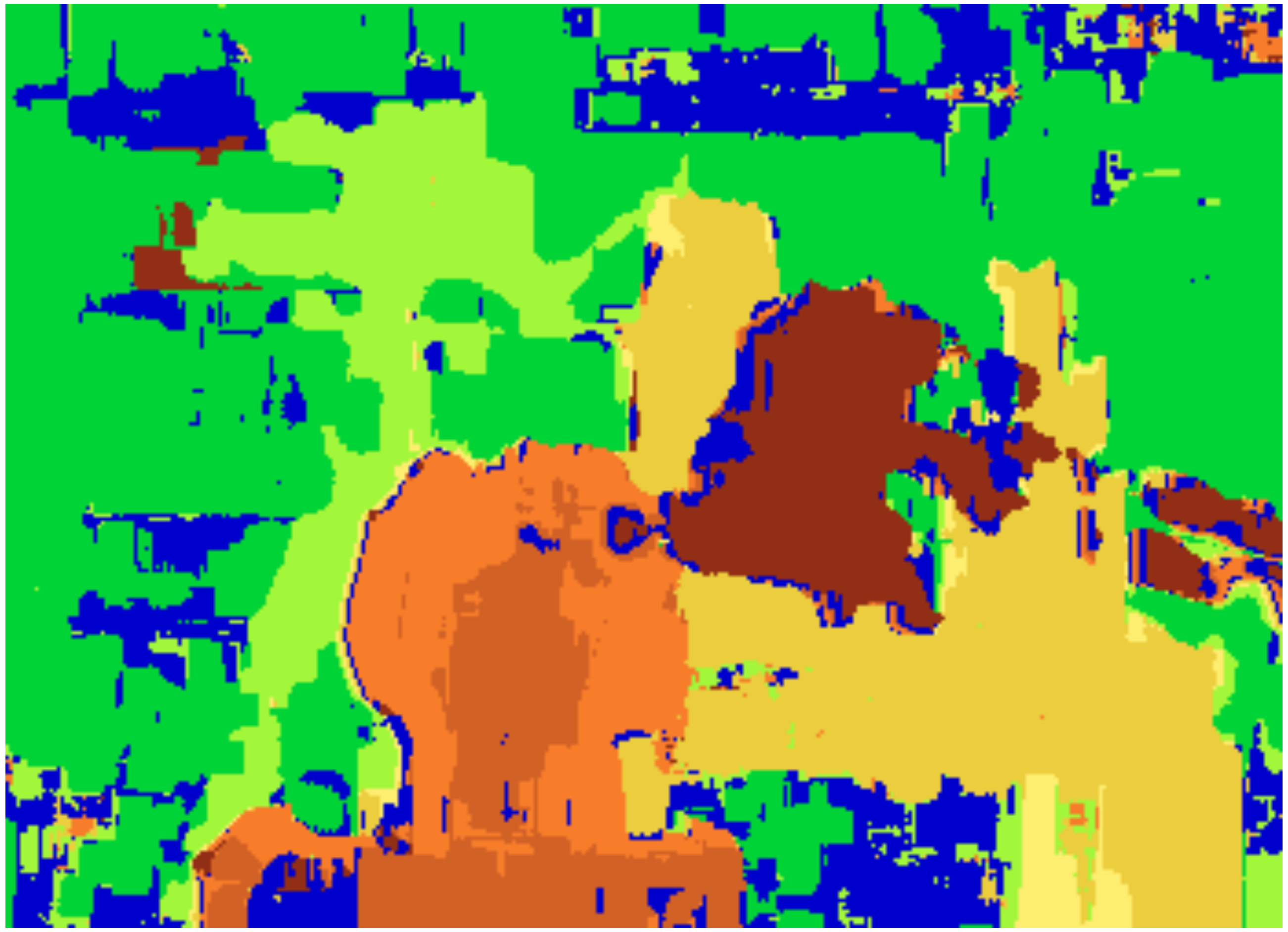
Stereo as energy minimization

Simple pixel / window matching: choose the minimum of each column in the DSI independently:

$$d(x,y) = \arg\min_{d'} \; C(x,y,d')$$

Slide credit: Noah Snavely

Greedy selection of best match



Slide credit: Noah Snavely

Stereo as energy minimization

- Better objective function

$$E(d) = \underbrace{E_d(d)}_{\text{match cost}} + \lambda \underbrace{E_s(d)}_{\text{smoothness cost}}$$

Want each pixel to find a good match in the other image

Adjacent pixels should (usually) move about the same amount

Slide credit: Noah Snavely

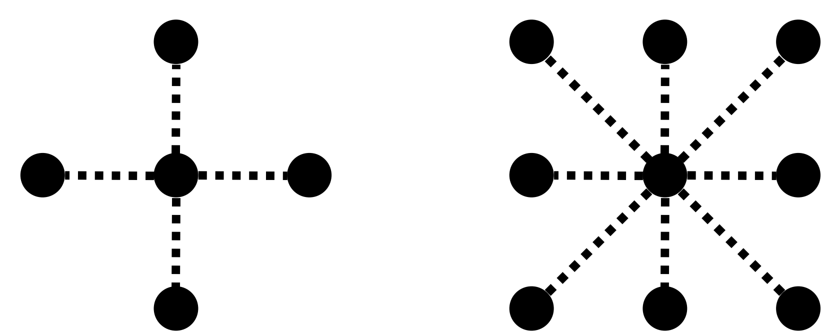
Stereo as energy minimization

$$E(d) = E_d(d) + \lambda E_s(d)$$

match cost: $$E_d(d) = \sum_{(x,y) \in I} C(x, y, d(x, y))$$

smoothness cost: $$E_s(d) = \sum_{(p,q) \in \mathcal{E}} V(d_p, d_q)$$

$\mathcal{E}$: set of neighboring pixels



4-connected neighborhood

8-connected neighborhood

Slide credit: Noah Snavely

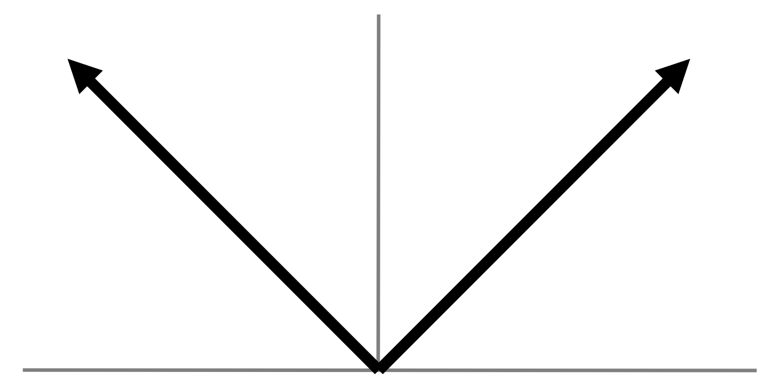
Smoothness cost

$$E_s(d) = \sum_{(p,q)\in\mathcal{E}} V(d_p, d_q)$$

How do we choose *V*?

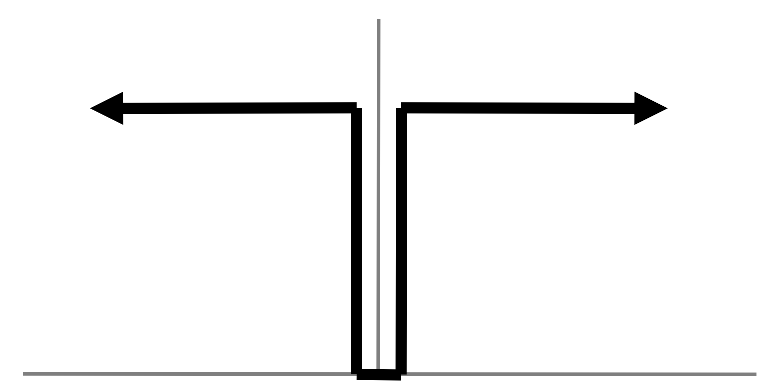
$$V(d_p, d_q) = |d_p - d_q|$$

L_1 distance



$$V(d_p, d_q) = \begin{cases} 0 & \text{if } d_p = d_q \\ 1 & \text{if } d_p \neq d_q \end{cases}$$

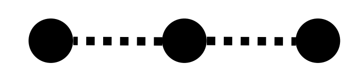
"Potts model"



Slide credit: Noah Snavely

Dynamic programming

$$E(d) = E_d(d) + \lambda E_s(d)$$

- Can minimize this independently per scanline using dynamic programming (DP) 
- Basic idea: incrementally build a table of costs *D* one column at a time

$D(x, y, i)$: minimum cost of solution such that $d(x,y) = i$

Base case: $D(0, y, i) = C(0, y, i), i = 0, \ldots, L$ (L = max disparity)

Recurrence: $D(x, y, i) = C(x, y, i) + \min_{j \in \{0,1,\ldots,L\}} D(x-1, y, j) + \lambda |i - j|$

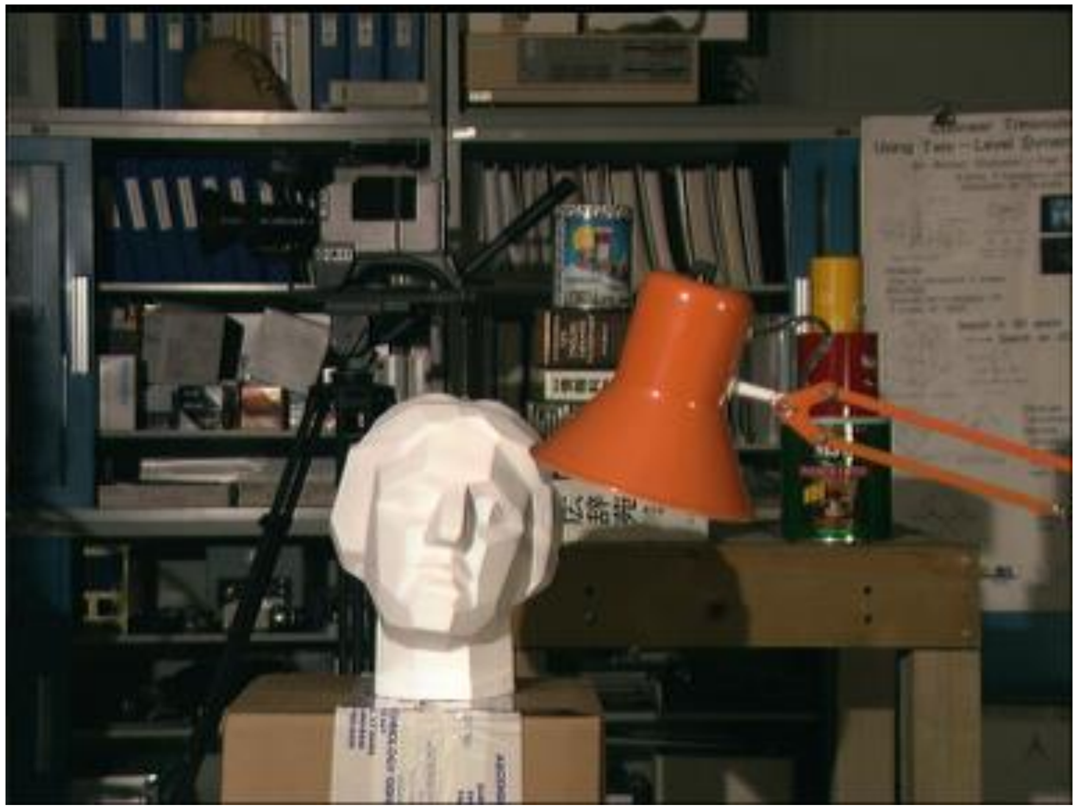
Slide credit: Noah Snavely

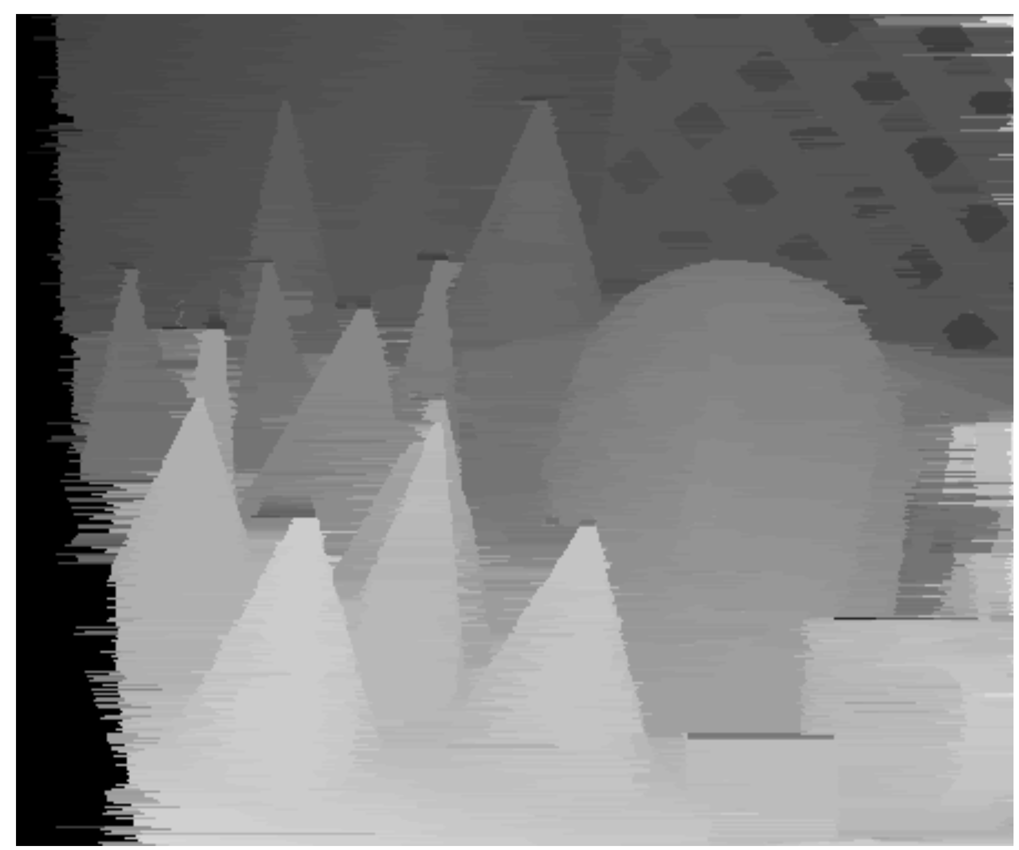
Dynamic programming

- Finds "smooth", low-cost path through DPI from left to right

Slide credit: Noah Snavely

Dynamic Programming





Stereo as a minimization problem

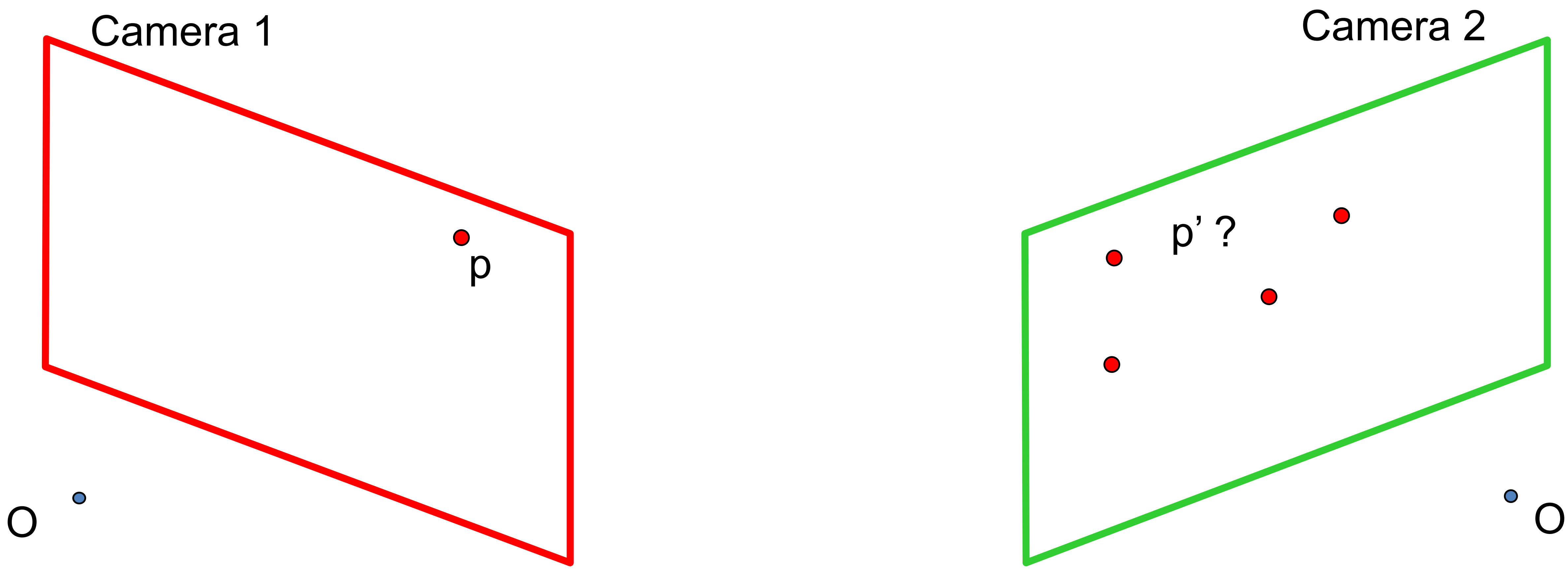
$$E(d) = E_d(d) + \lambda E_s(d)$$

- The 2D problem has many local minima
  - Gradient descent doesn't work well

- And a large search space
  - n x m image w/ k disparities has k^{nm} possible solutions
  - Finding the global minimum is NP-hard in general

Slide credit: Noah Snavely

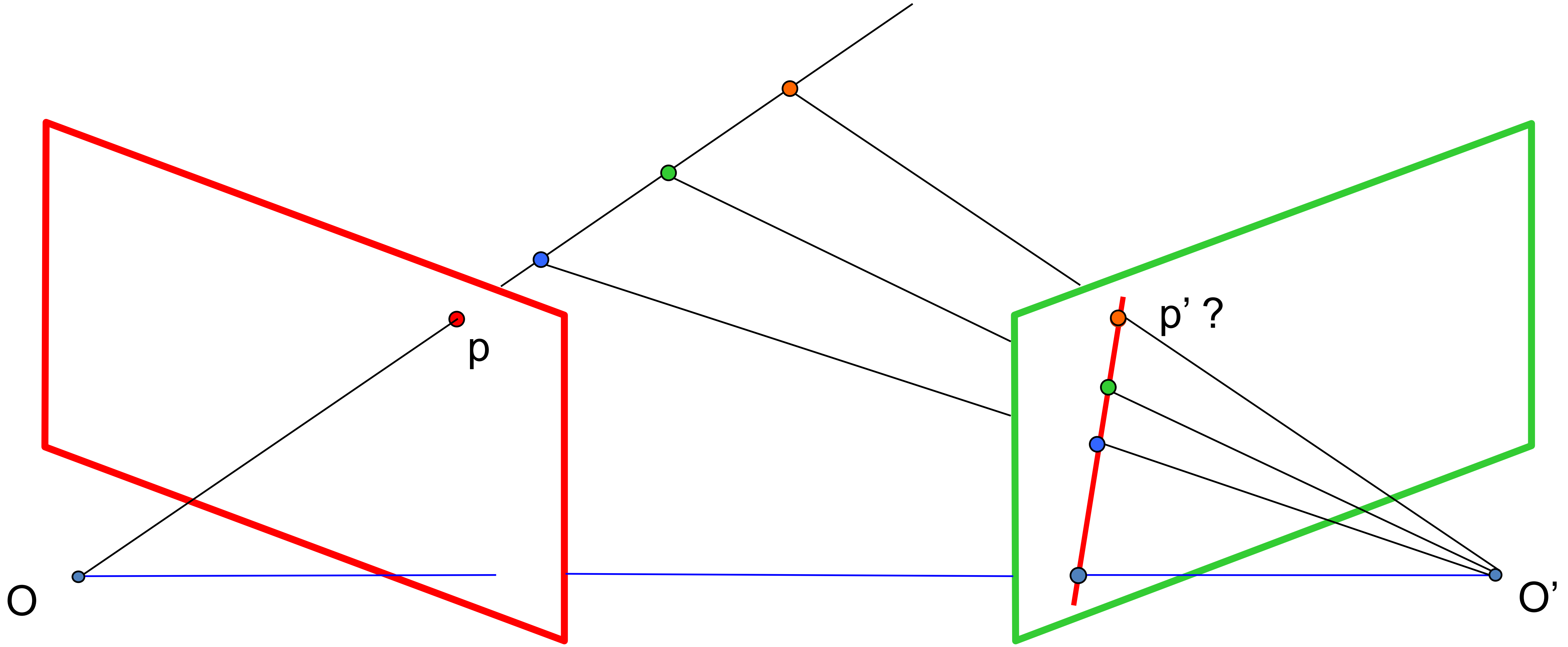
Stereo correspondence constraints



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

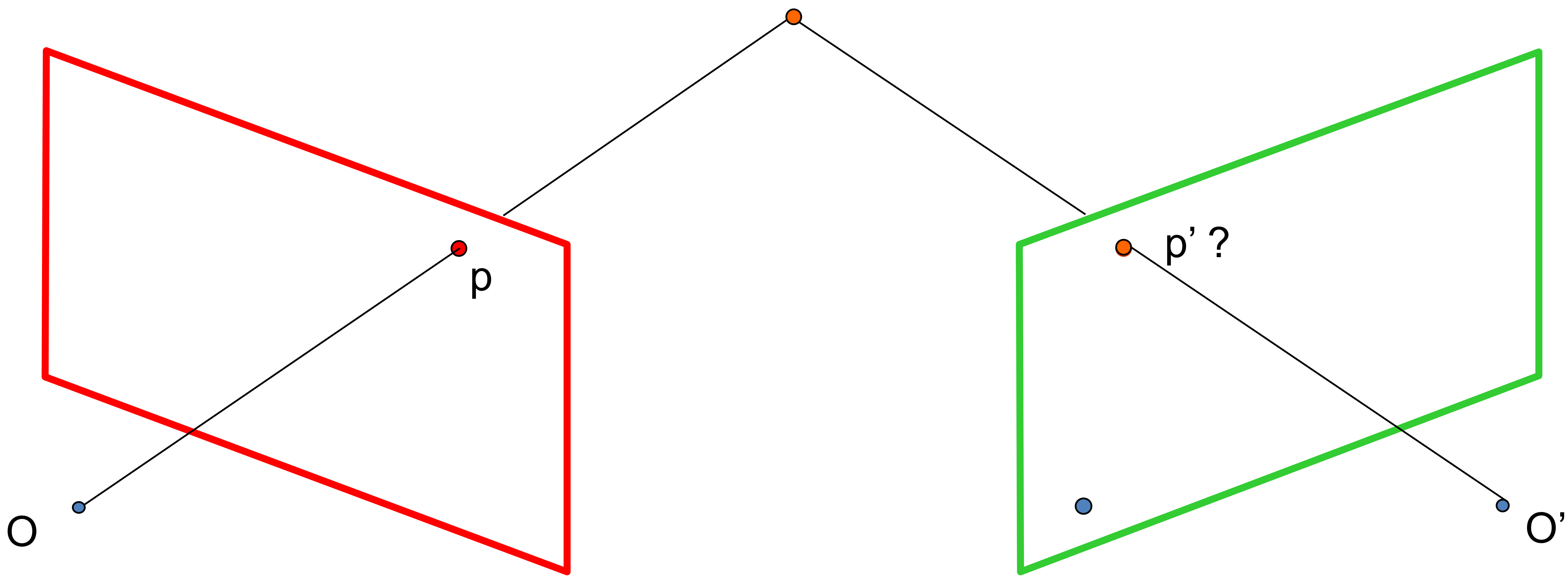
Slide credit: Antonio Torralba

Epipolar constraint



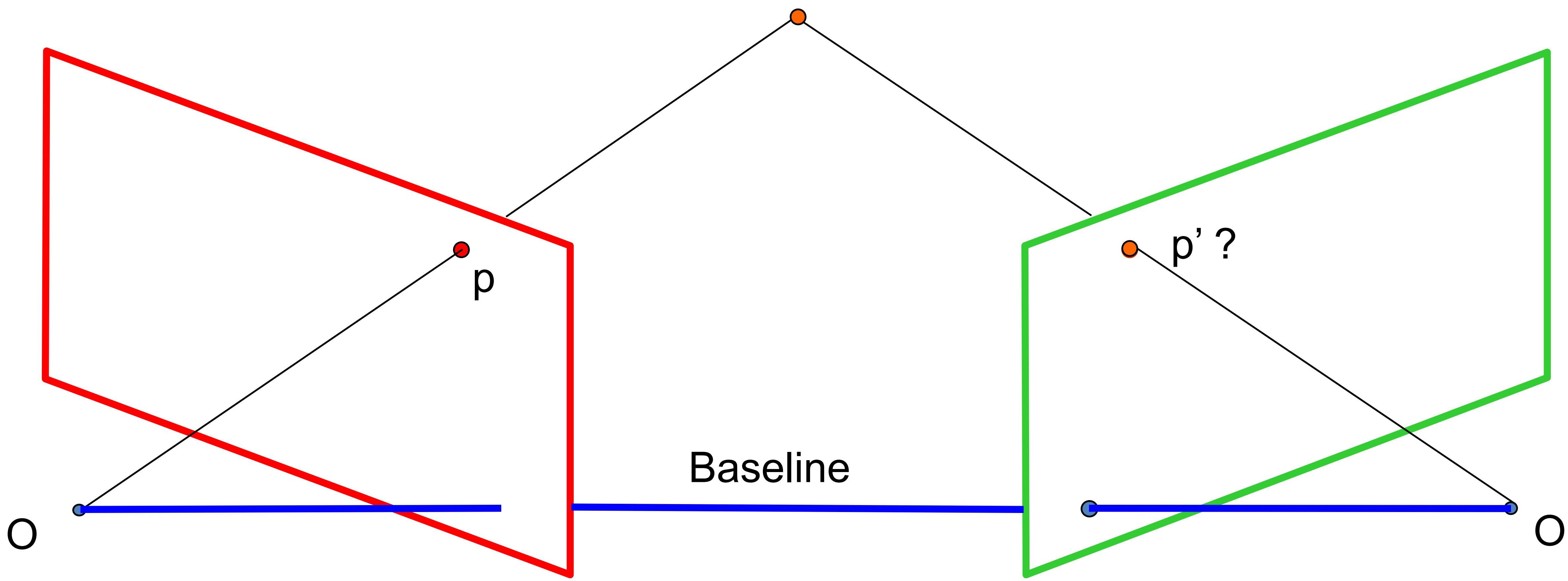
Slide credit: Antonio Torralba

Some terminology



Slide credit: Antonio Torralba

Some terminology

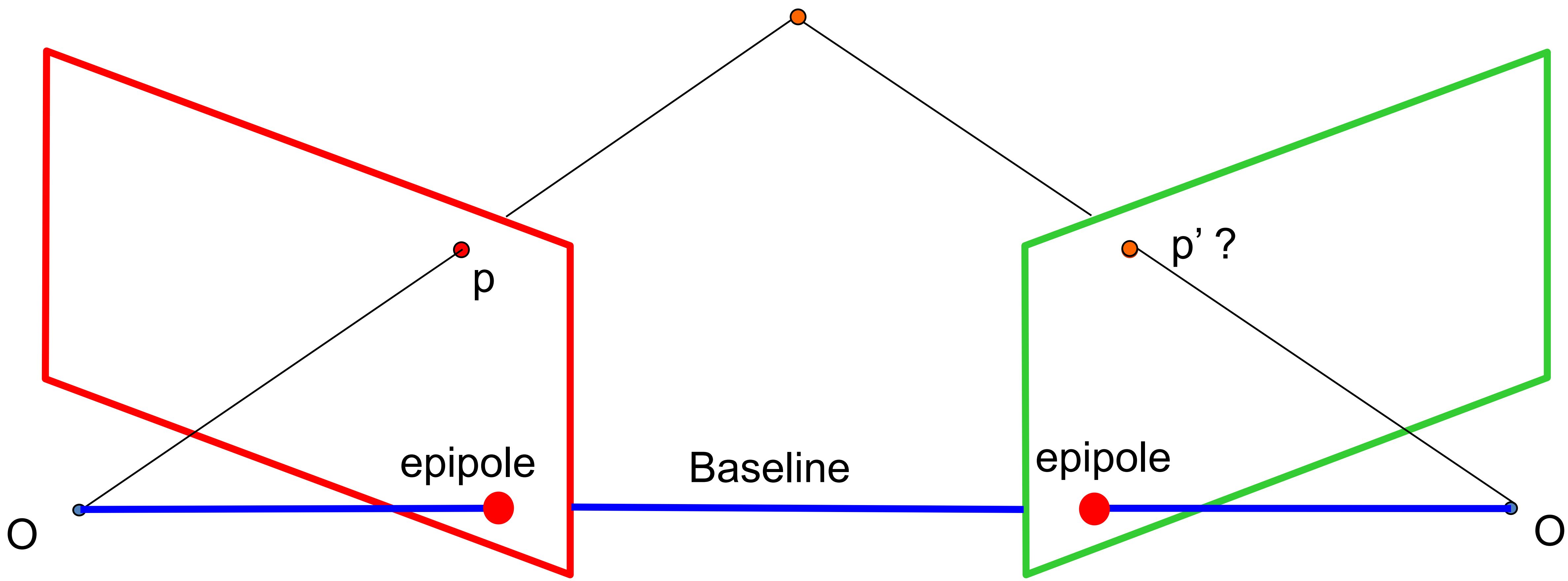


Baseline: the line connecting the two camera centers

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

Slide credit: Antonio Torralba

Some terminology

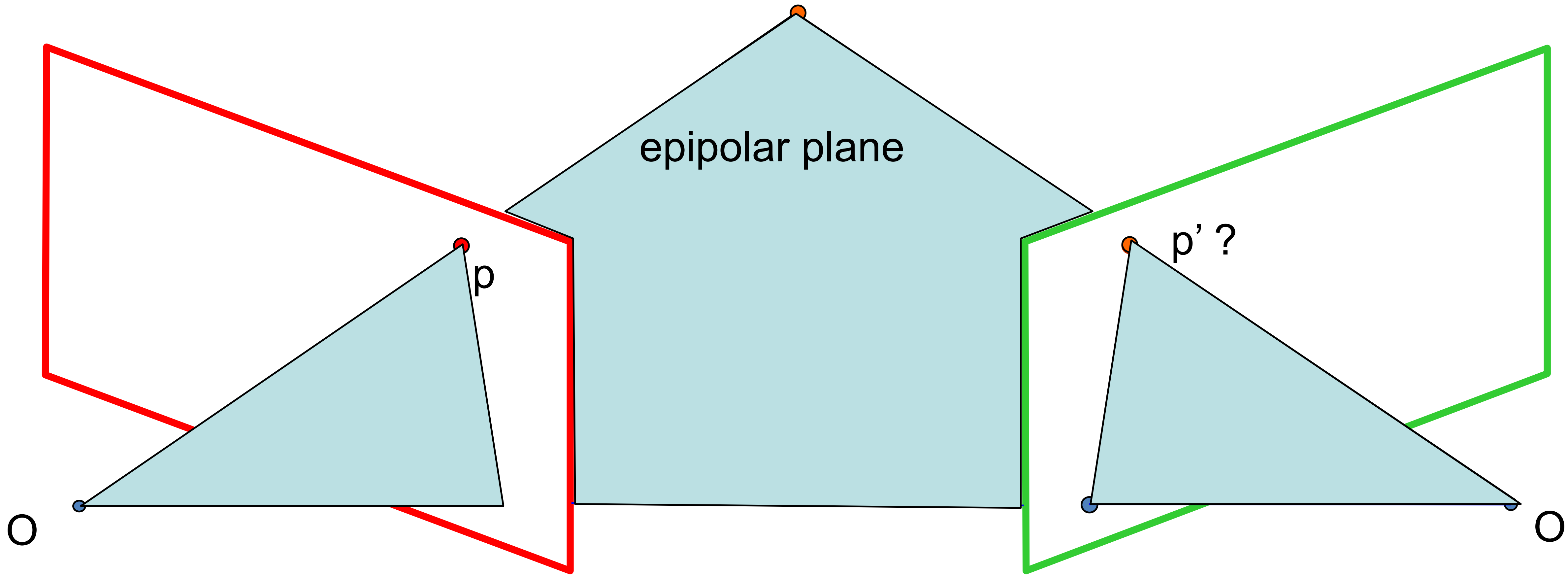


Baseline: the line connecting the two camera centers

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

Slide credit: Antonio Torralba

Some terminology



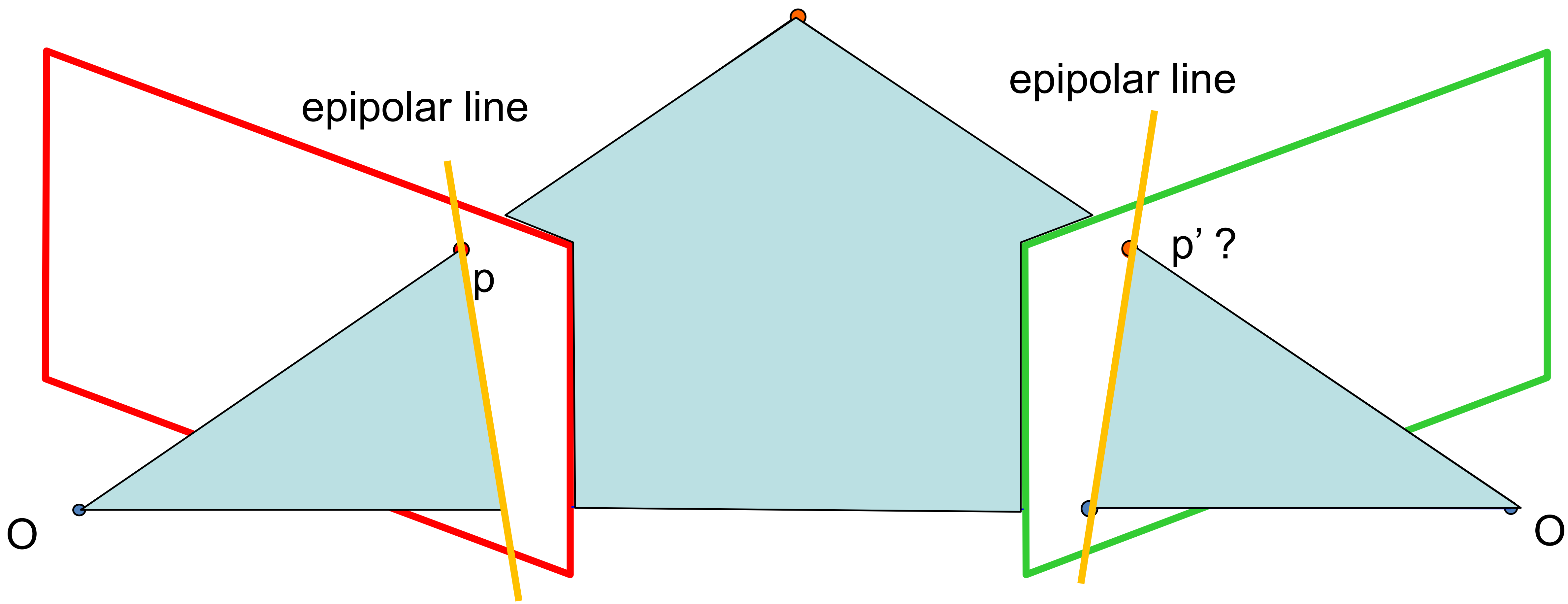
Baseline: the line connecting the two camera centers

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

Slide credit: Antonio Torralba

Some terminology



Baseline: the line connecting the two camera centers

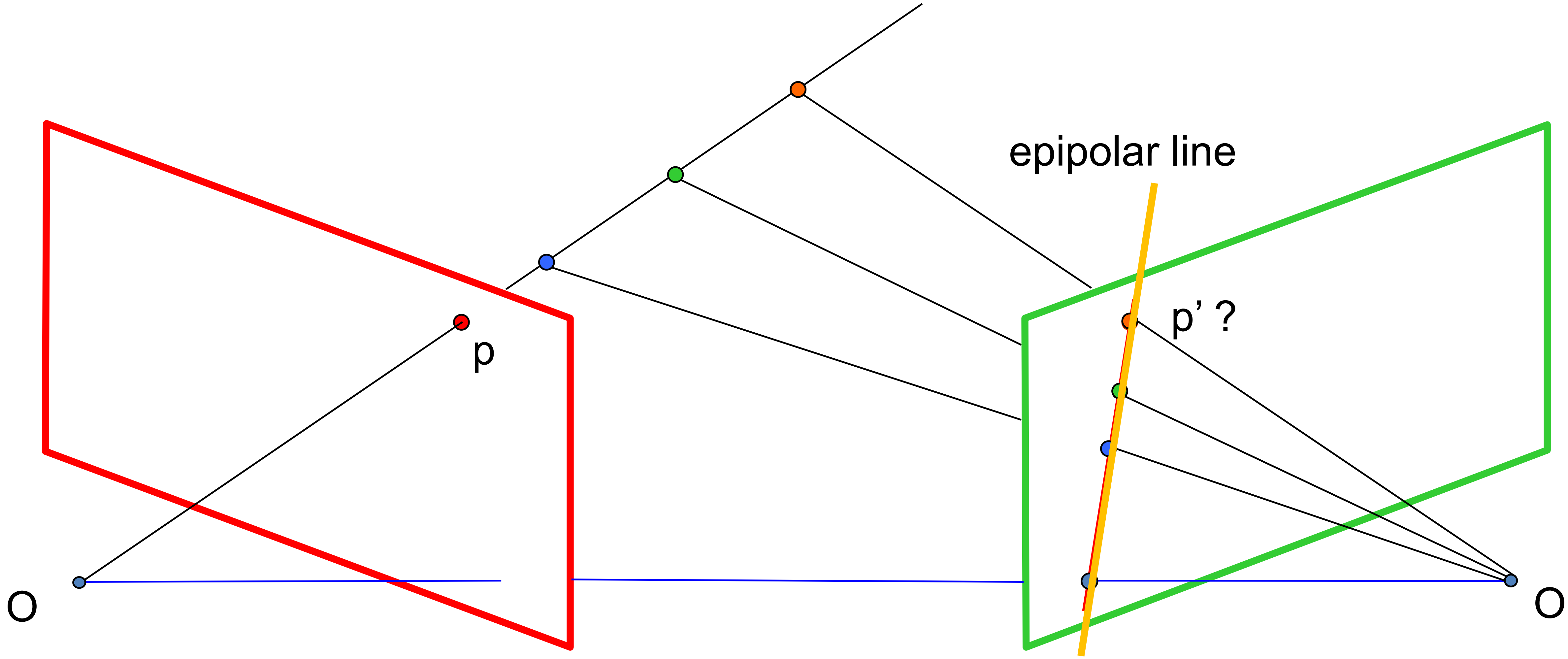
Epipole: point of intersection of *baseline* with the image plane

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

Slide credit: Antonio Torralba

Epipolar constraint

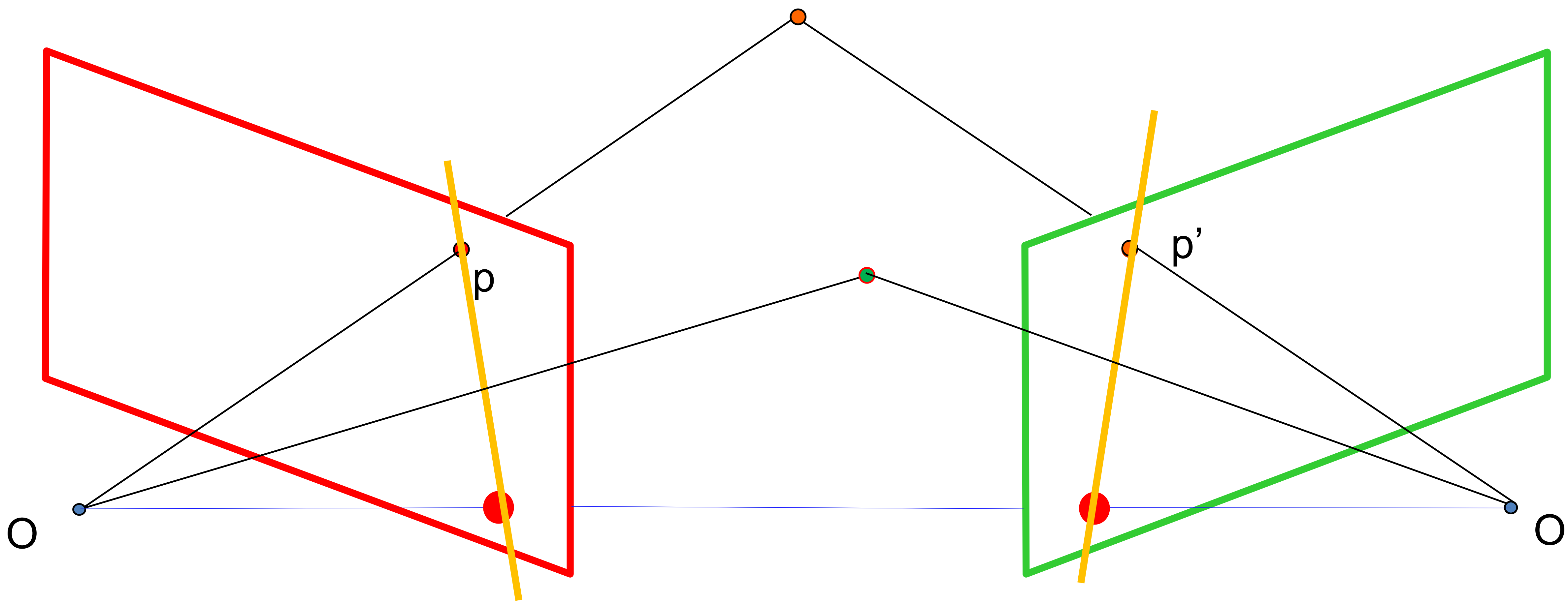


We can search for matches across epipolar lines

All epipolar lines intersect at the epipoles

Slide credit: Antonio Torralba

The essential matrix



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

$$p^T E p' = 0$$

E: essential matrix
p, p': image points in homogeneous coordinates

Slide credit: Antonio Torralba

Real-time stereo

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

- Used for robot navigation (and other tasks)
  - Several real-time stereo techniques have been developed (most based on simple discrete search)

Stereo reconstruction pipeline

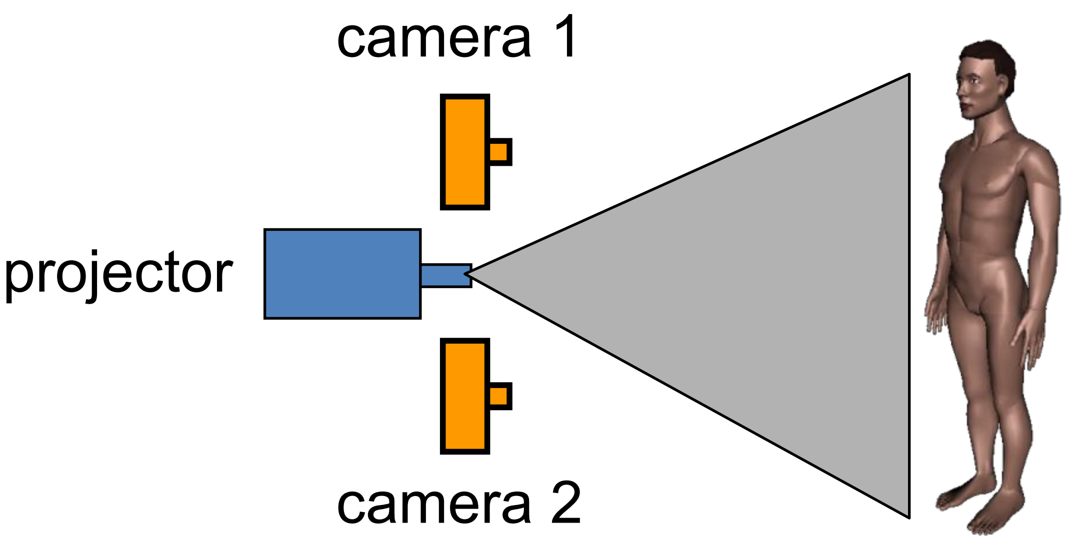
- Steps
  - Calibrate cameras
  - Rectify images
  - Compute disparity
  - Estimate depth

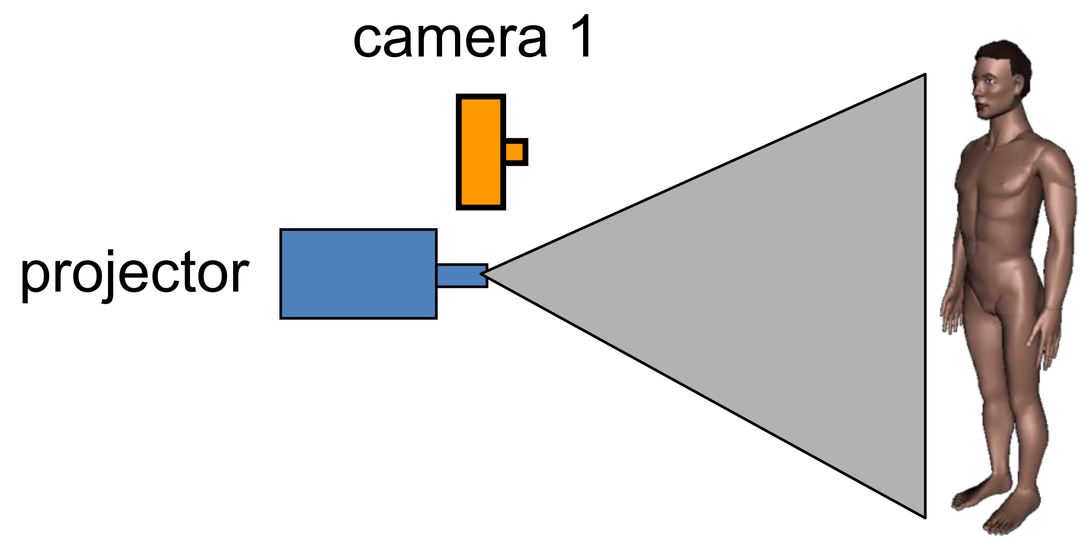
What will cause errors?

- Camera calibration errors
- Poor image resolution
- Occlusions
- Violations of brightness constancy (specular reflections)
- Large motions
- **Low-contrast image regions**

Active stereo with structured light

Li Zhang's one-shot stereo



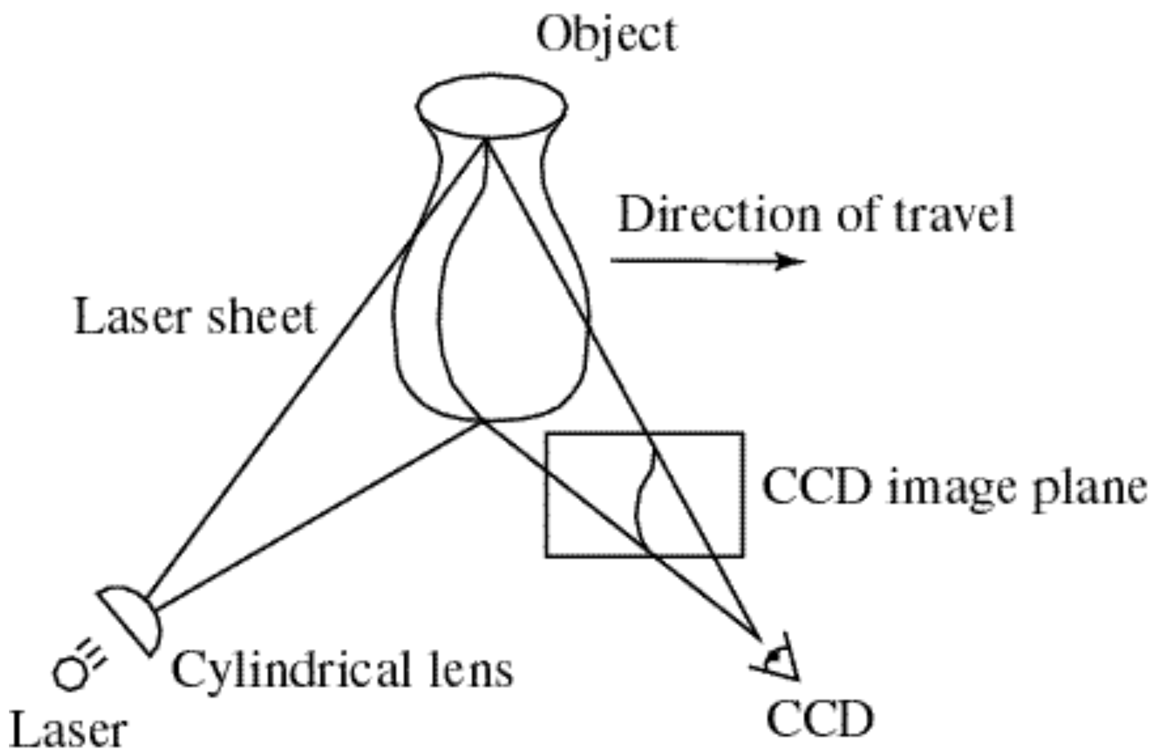


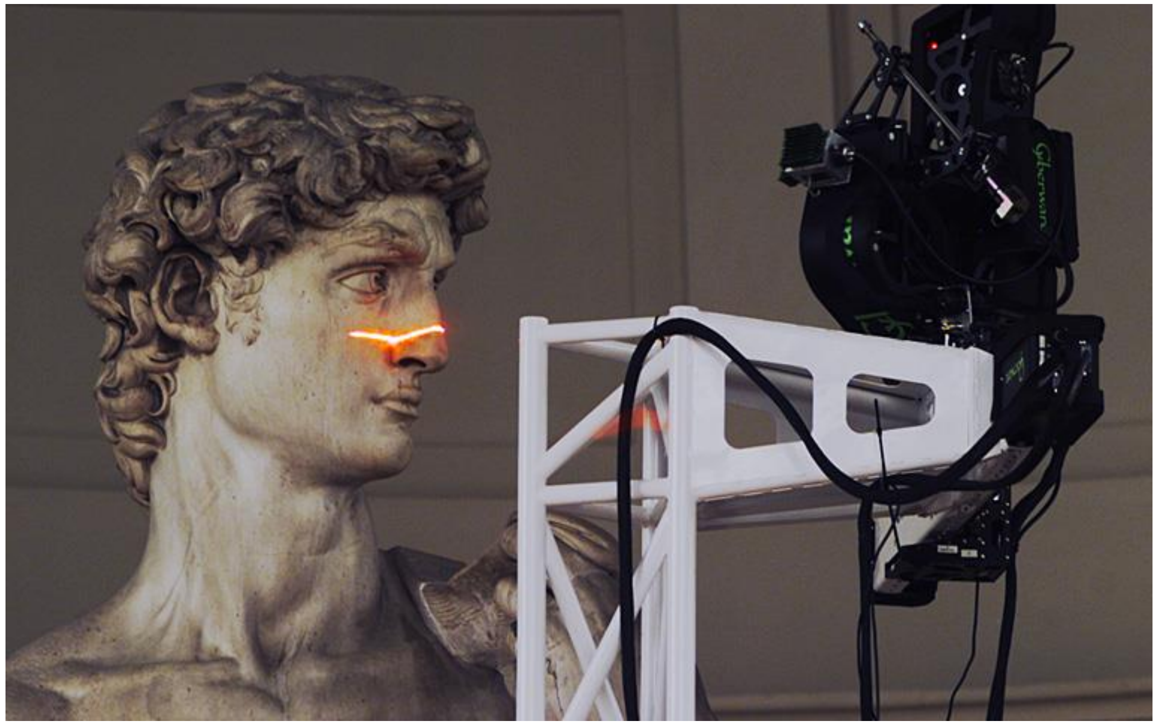
- Project "structured" light patterns onto the object
  - simplifies the correspondence problem
  - basis for active depth sensors, such as Kinect and iPhone X (using IR)

Active stereo with structured light

https://ios.gadgethacks.com/news/watch-iphone-xs-30k-ir-dots-scan-your-face-0180944/

Laser scanning





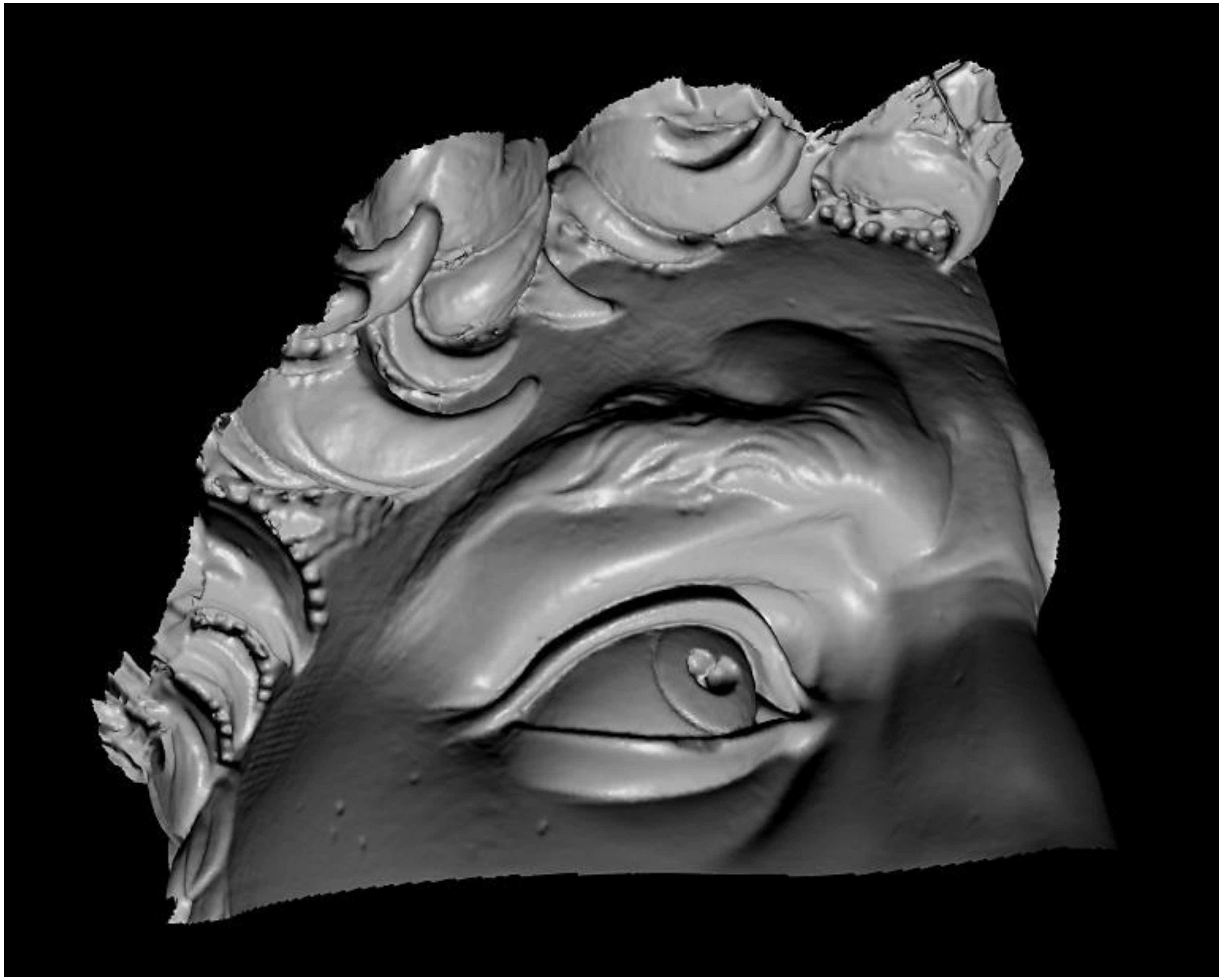
Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/

- 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

Laser scanned models

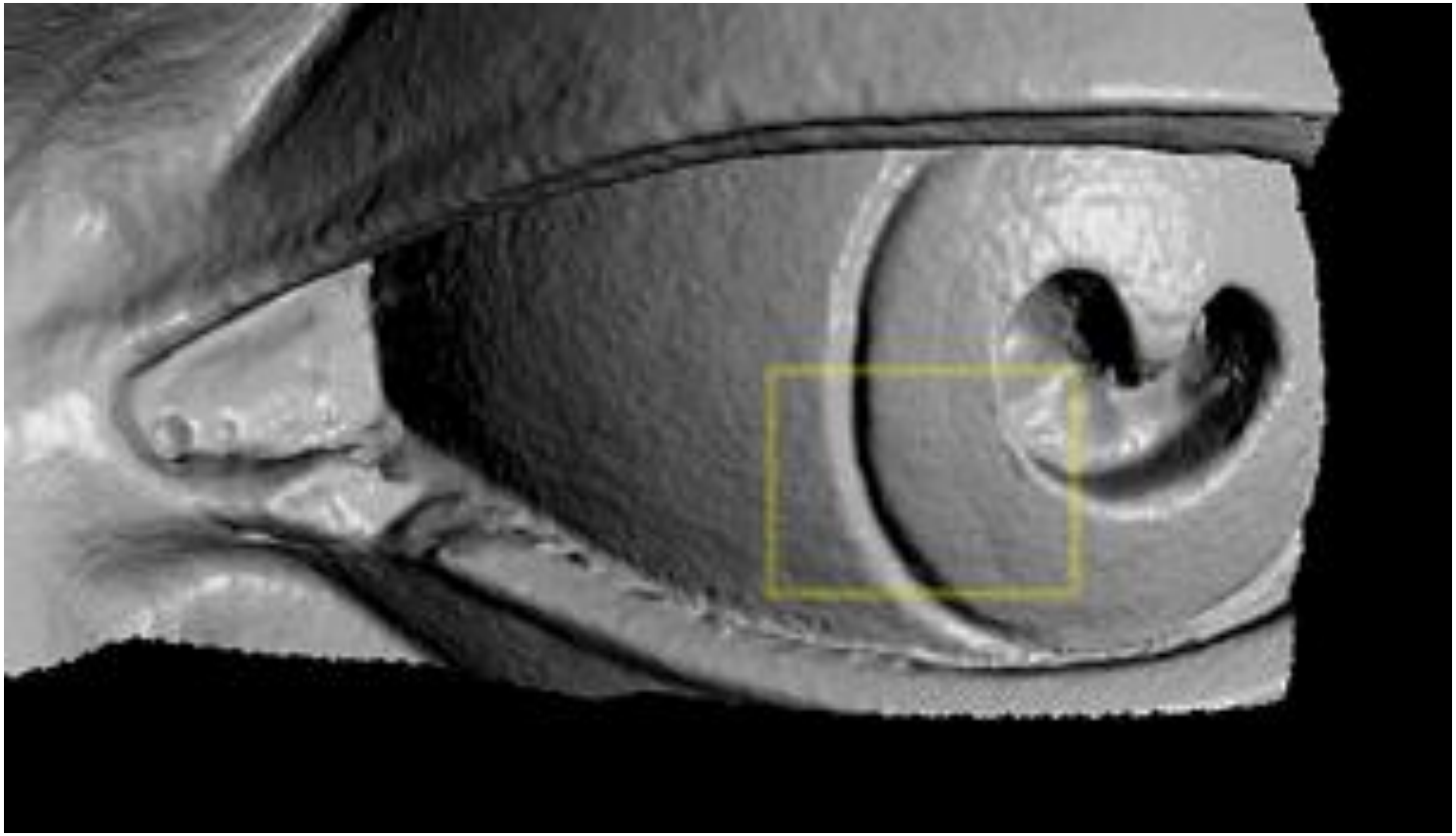
The Digital Michelangelo Project, Levoy et al.

Laser scanned models



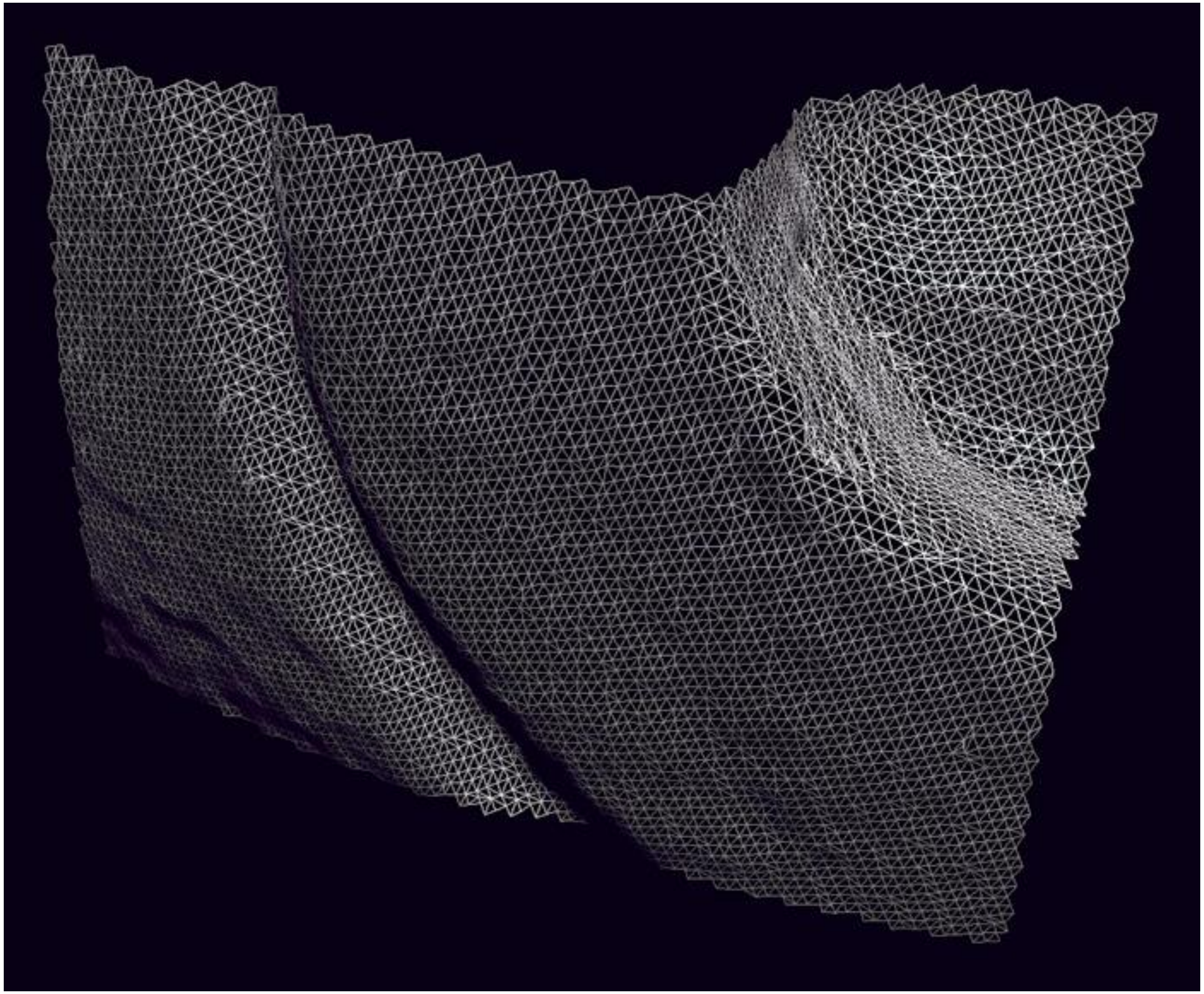
The Digital Michelangelo Project, Levoy et al.

Laser scanned models



The Digital Michelangelo Project, Levoy et al.

Laser scanned models



The Digital Michelangelo Project, Levoy et al.