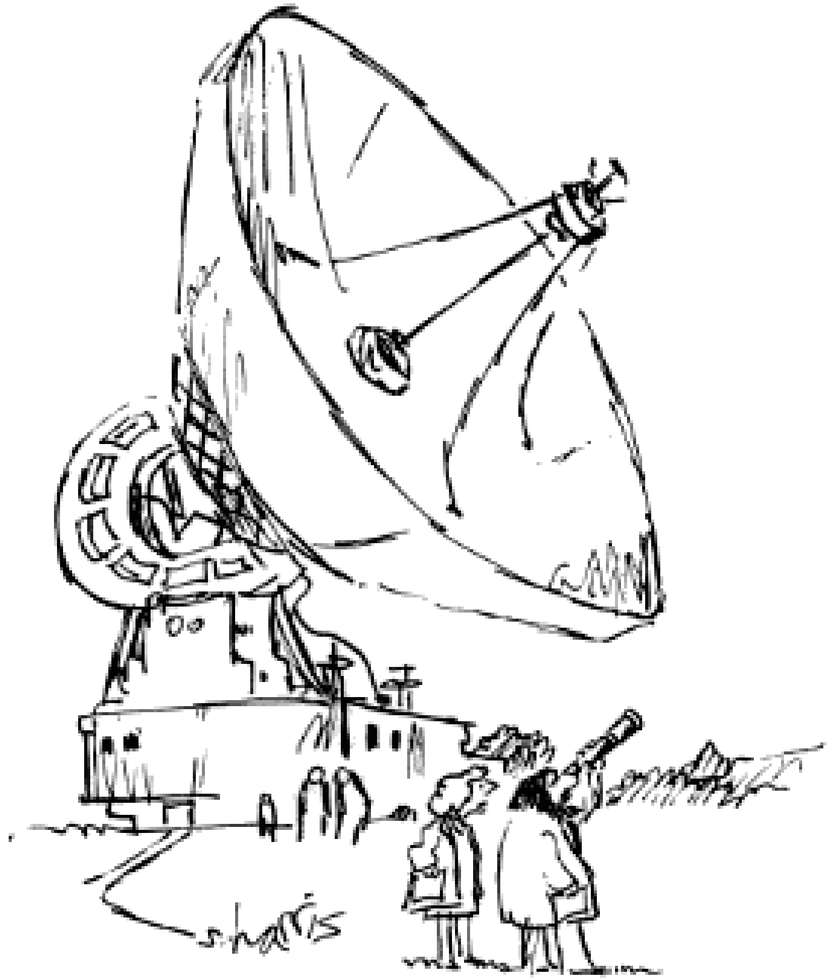
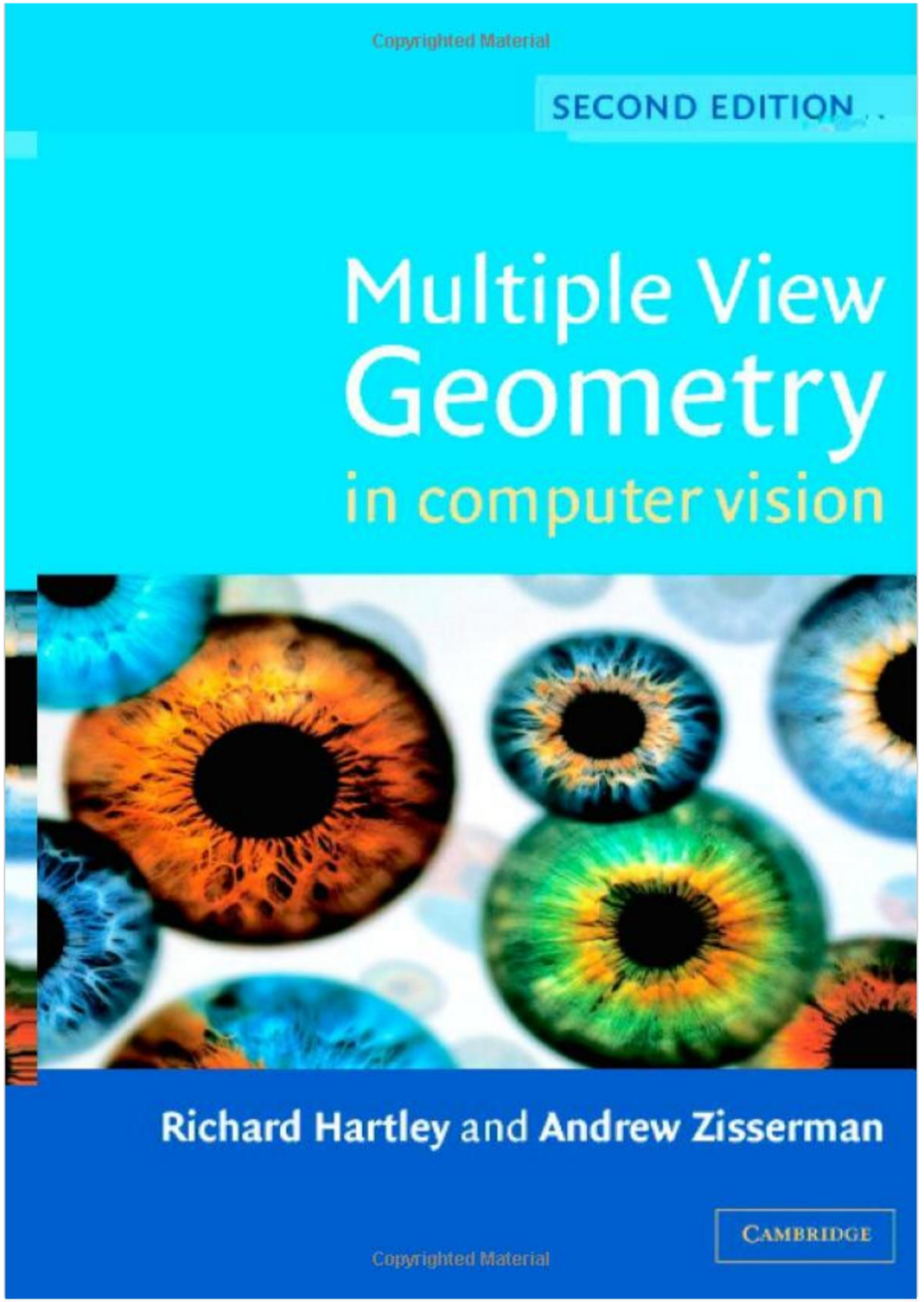
Stereopsis and Epipolar Geometry



"Just checking."

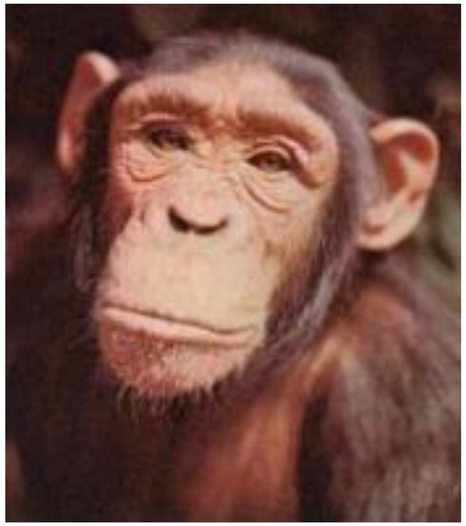
CS280: Computer Vision
Alexei Efros, UC Berkeley, Spring 2024


Copyrighted Material
SECOND EDITION
Multiple View Geometry
in computer vision
Richard Hartley and Andrew Zisserman
CAMBRIDGE
Copyrighted Material

Vision systems

One camera

Two cameras

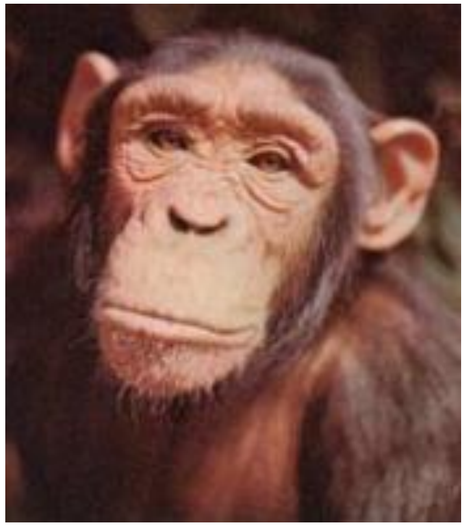


N cameras

Let’s consider two eyes

One camera

Two cameras



N cameras

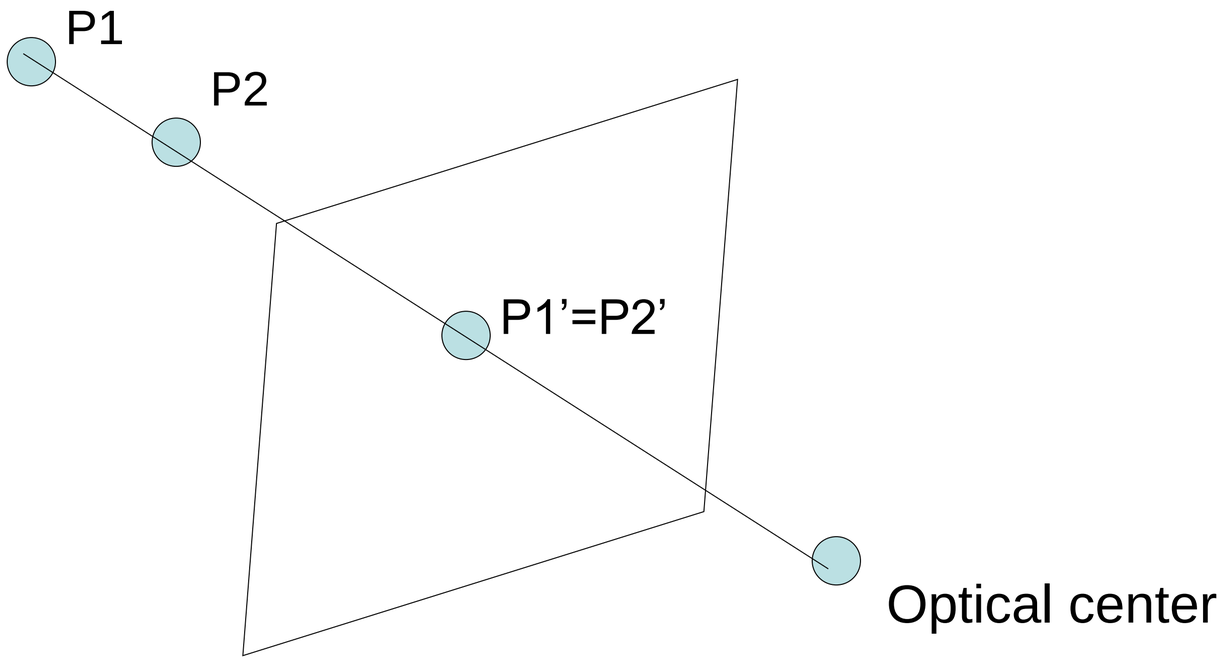
Why multiple views?

- Structure and depth are inherently ambiguous from single views.

Images from Lana Lazebnik

Why multiple views?

- Structure and depth are inherently ambiguous from single views.



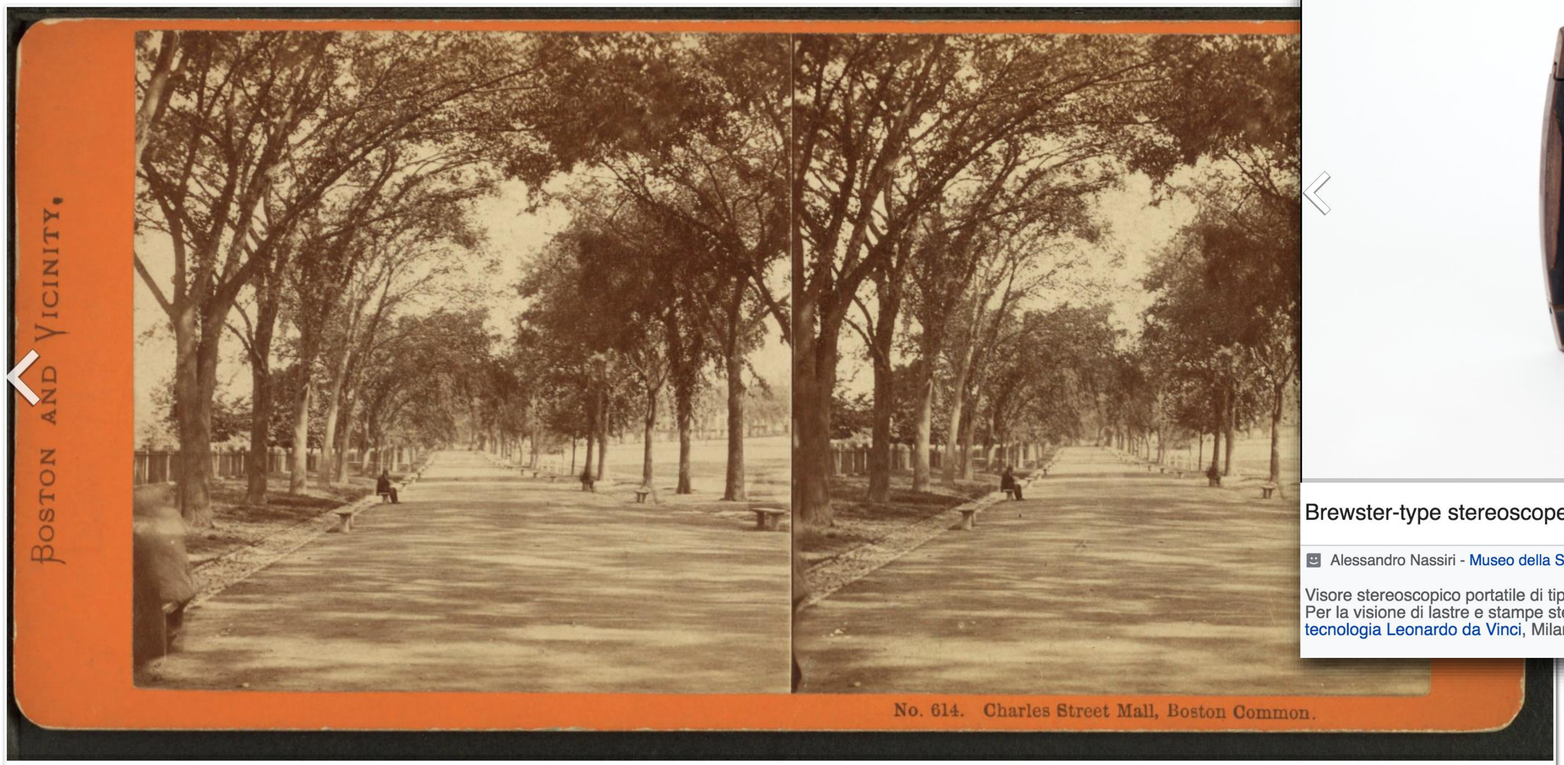
Stereo images of the Titanic

(a)

(b)

Figure 1.1: (a) Stereo anaglyph of the ocean liner, the Titanic [McManus2022]. The red image shows the right eye's view, and cyan the left eye's view. When viewed through stereo red/cyan stereo glasses, as in (b), the cyan contrast appears in the left eye image and the red variations appear to the right eye, creating a the perception of 3d.

Stereoscope



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.

Brewster-type stereoscope, 1870

More details

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

CC BY-SA 4.0

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.

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

Created: 1 July 2014

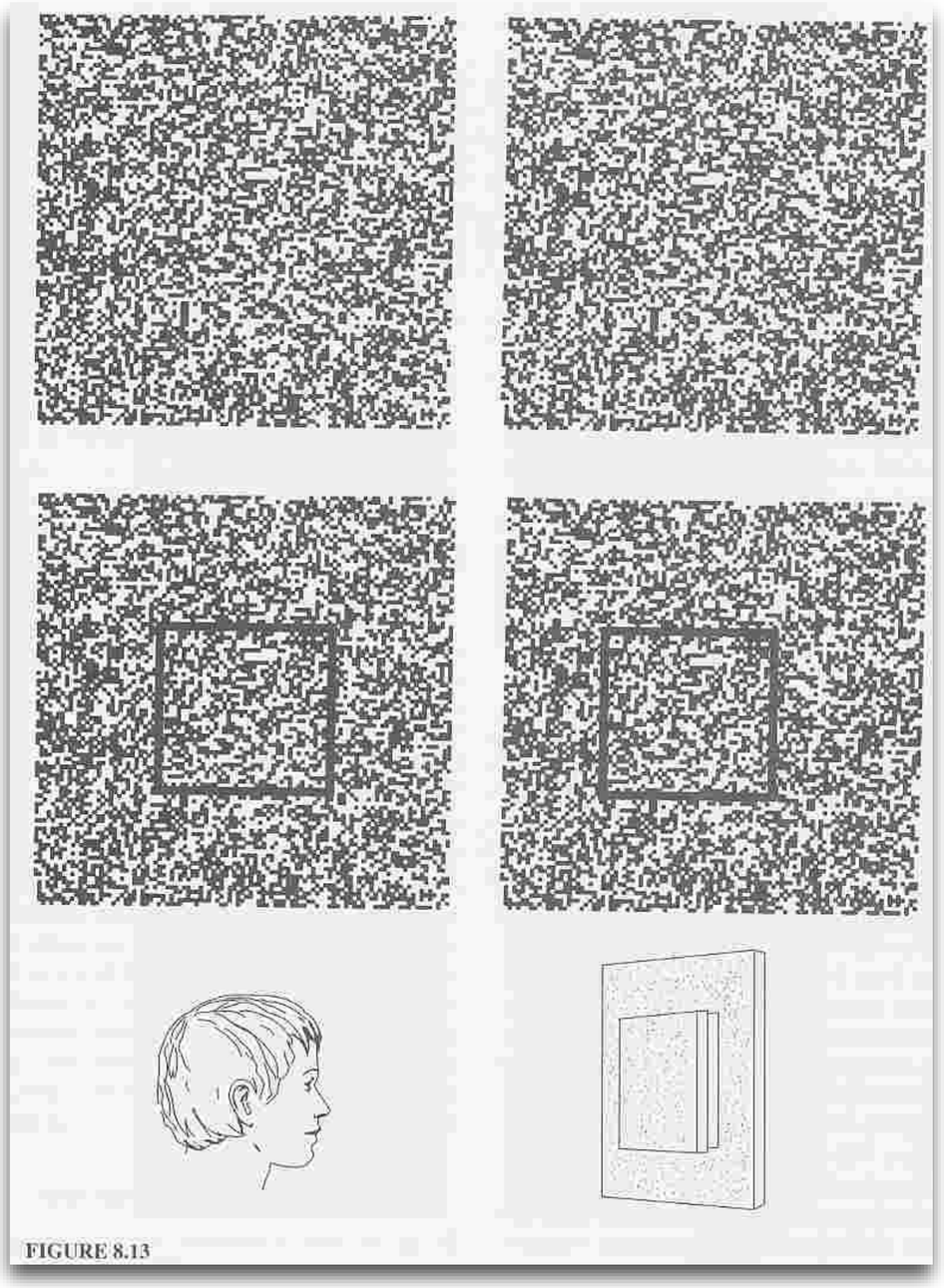
Depth without objects

Random dot stereograms (Bela Julesz)

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

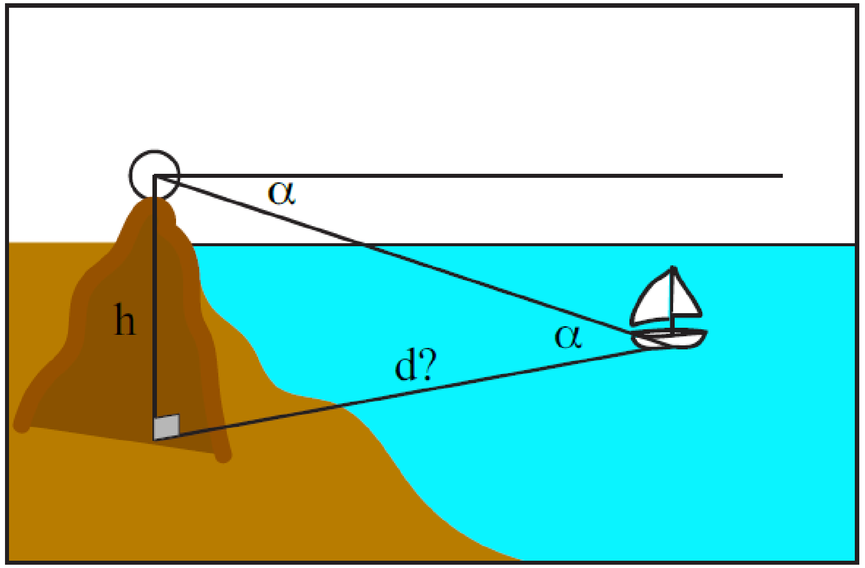
1	0	1	0	1	0	0	1	0	1
1	0	0	1	0	1	0	1	0	0
0	0	1	1	0	1	1	0	1	0
0	1	0	A	A	B	B	X	0	1
1	1	1	B	A	B	A	Y	0	1
0	0	1	A	A	B	A	Y	1	0
1	1	1	B	B	A	B	X	0	1
1	0	0	1	1	0	1	1	0	1
1	1	0	0	1	1	0	1	1	1
0	1	0	0	0	1	1	1	1	0

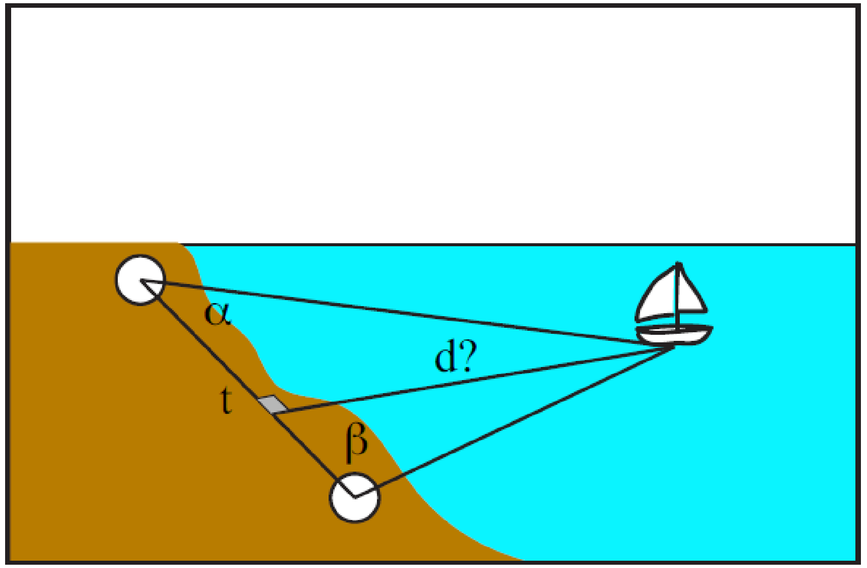
Julesz, 1971



Ship Navigation

Figure 40.2: Two methods to estimate the distance of a boat from the coast. (left) The first method uses a single observation point, with knowledge of the observer's height above the water. (right) The second method uses two observation points.

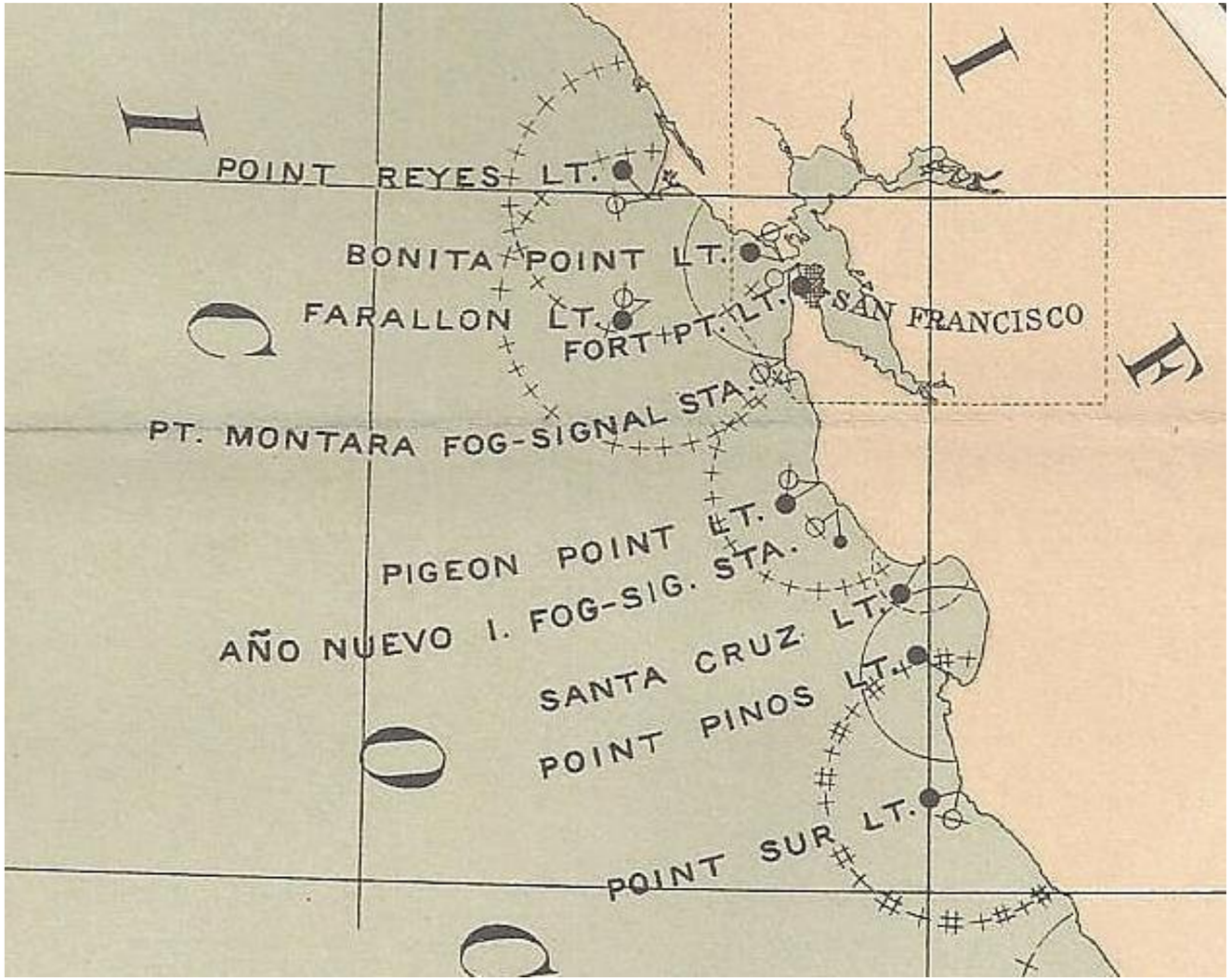


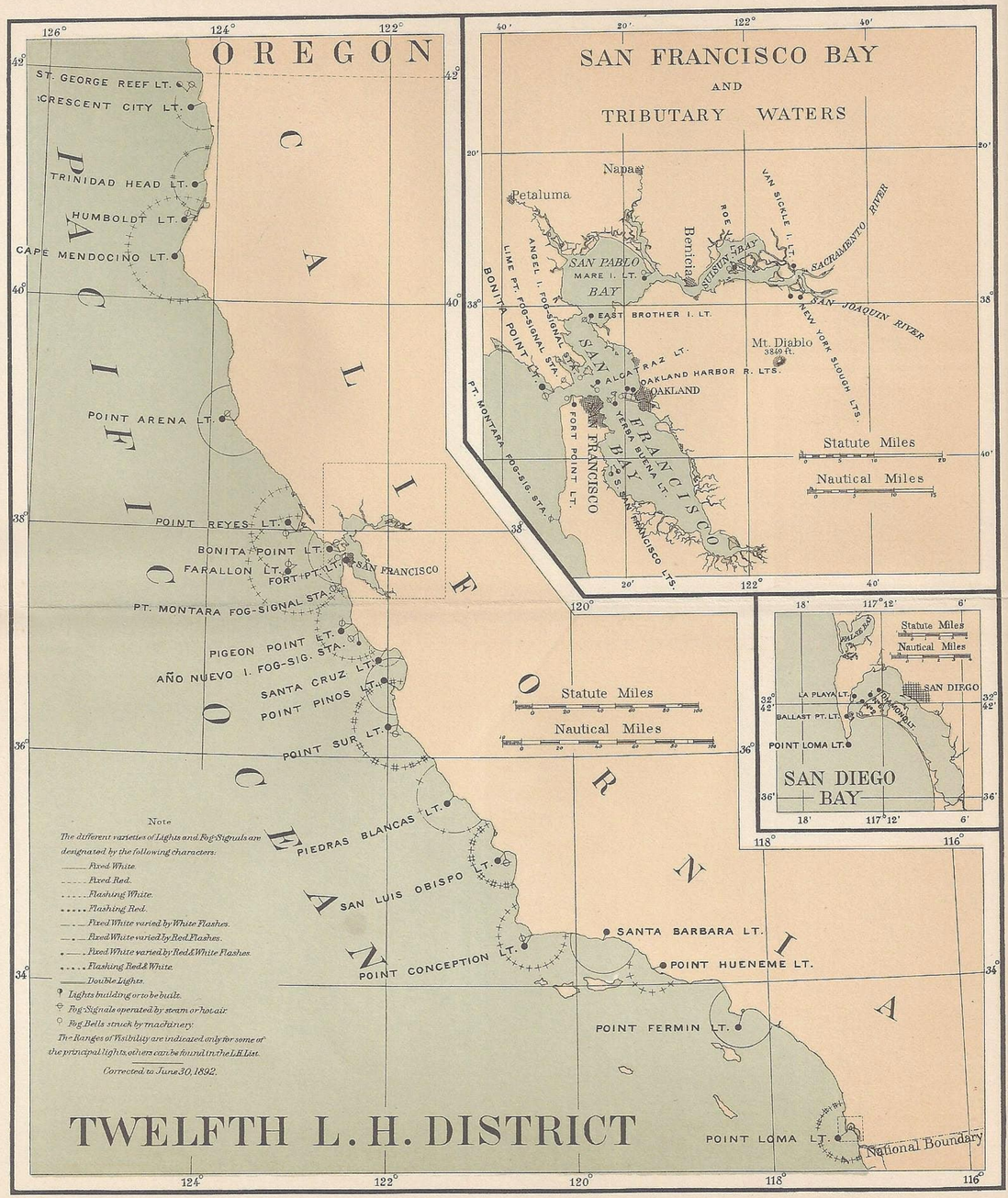


$$d = \frac{h}{\tan(\alpha)}$$

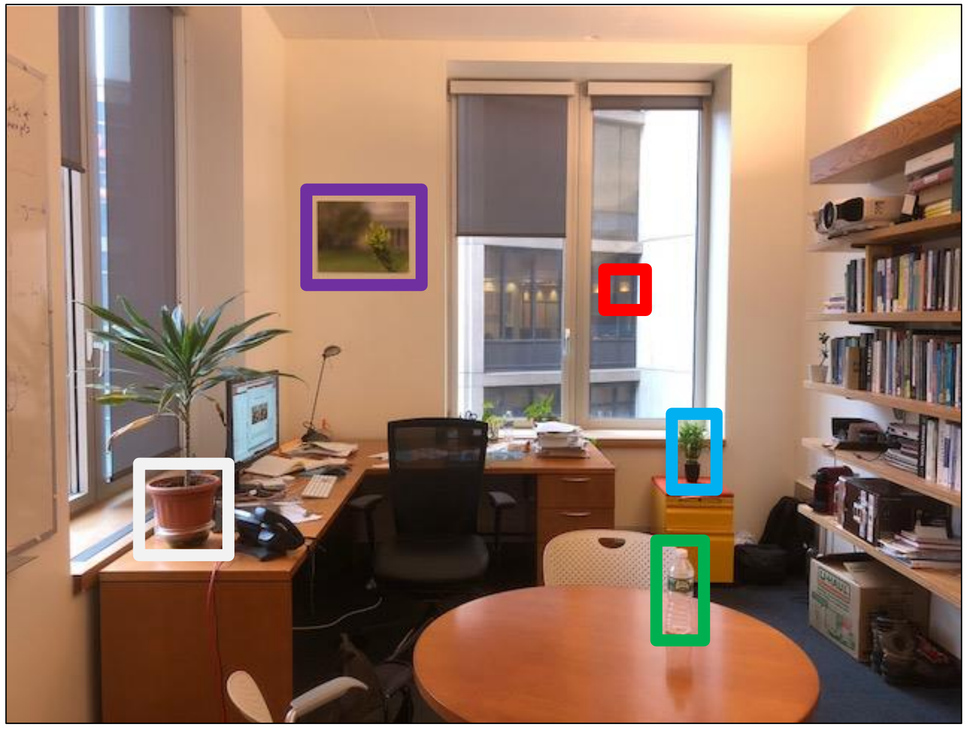
$$d = t\frac{\sin(\alpha)\sin(\beta)}{\sin(\alpha+\beta)}$$

Triangulation





disparity

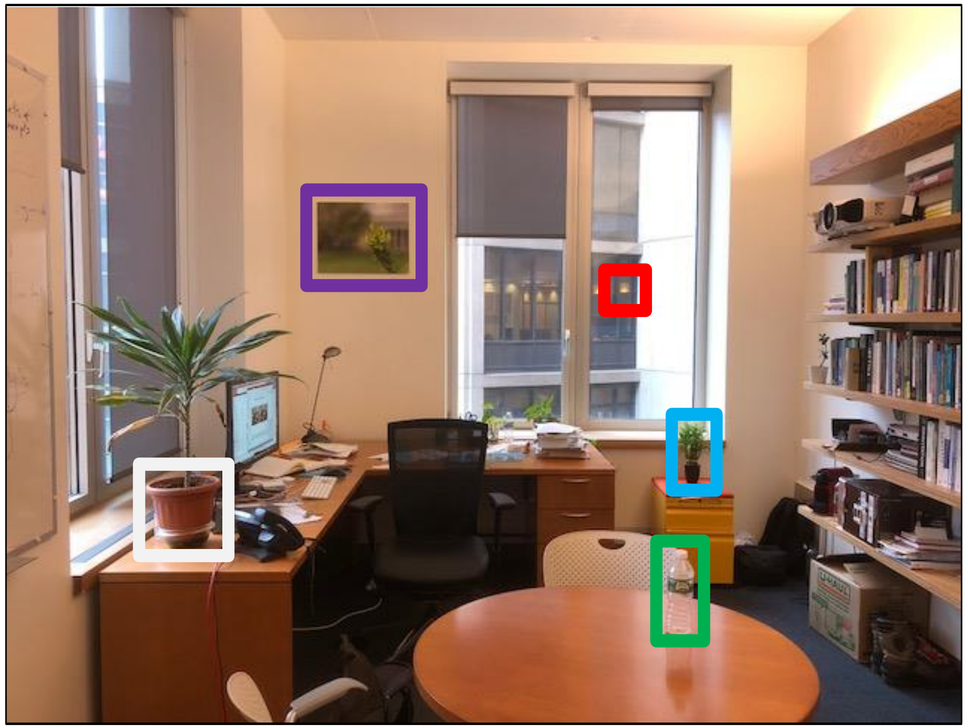


Left image

Right image

Antonio took one picture, then he moved ~1m to the right and took a second picture.

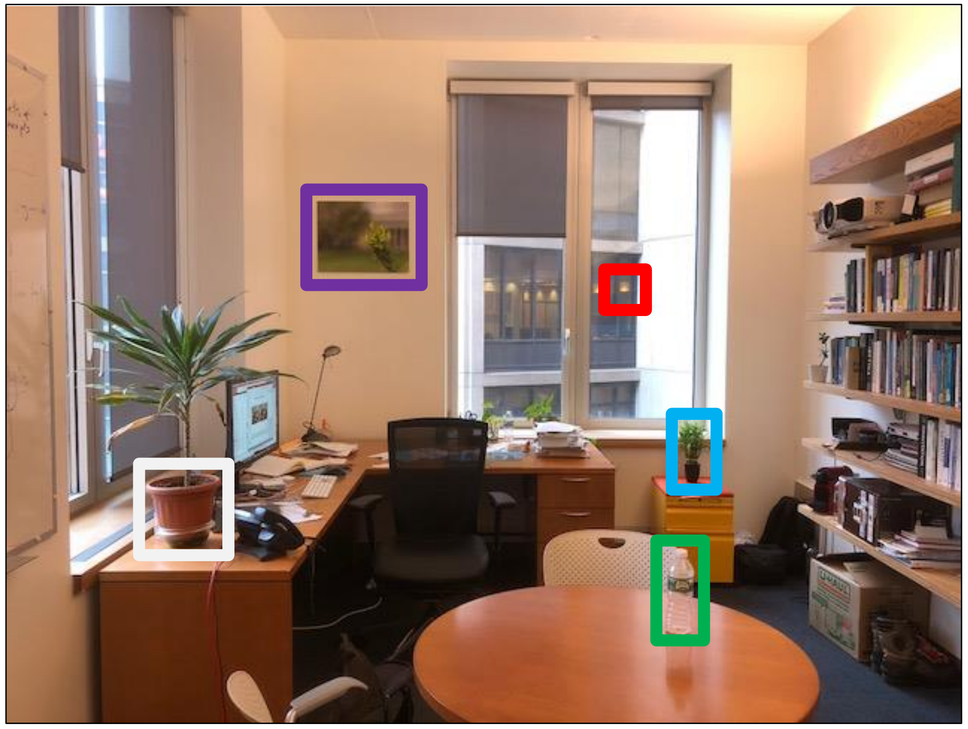
disparity



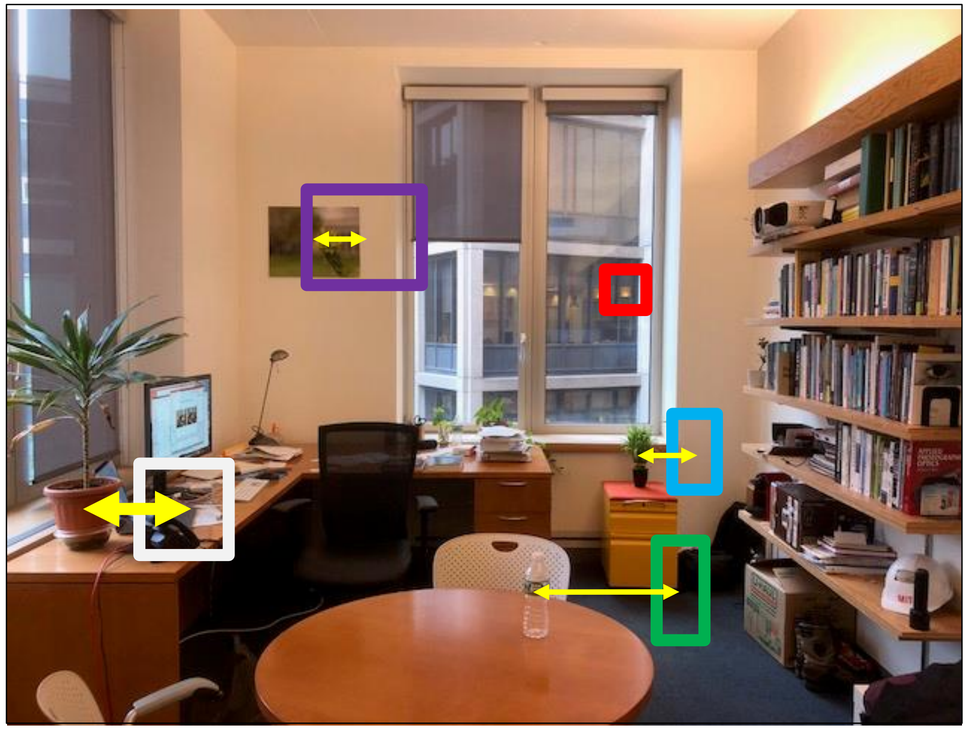
Left image

Right image

disparity



Left image



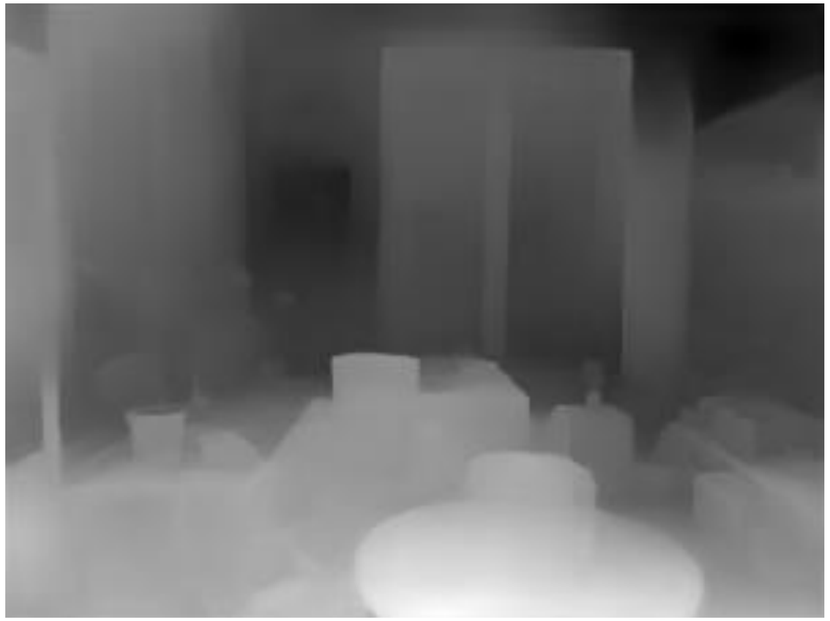
Right image

Disparity map

I(x,y)

I’(x,y) = I(x+D(x,y), y)

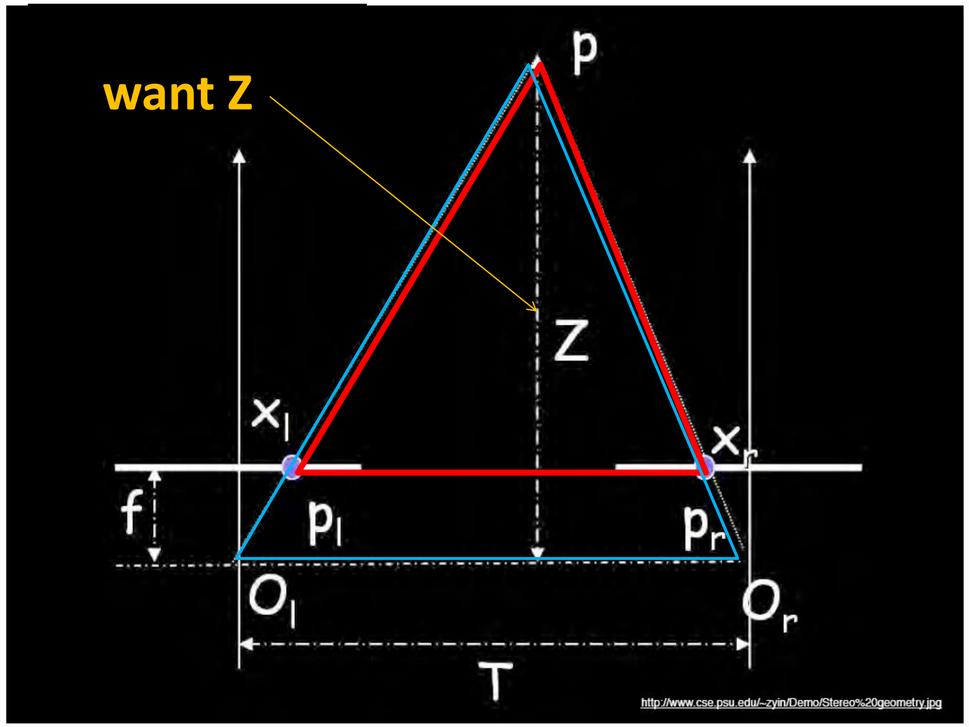
D(x,y)



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

Geometry for a simple stereo system

- Assume **parallel** optical axes, known camera parameters (i.e., calibrated cameras).



Use similar triangles (p_l, P, p_r) and (O_l, P, O_r):

$$\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}$$

$$Z = f \frac{T}{x_r - x_l}$$

disparity

Non-parametric transformation!

image I(x,y)

Disparity map D(x,y)

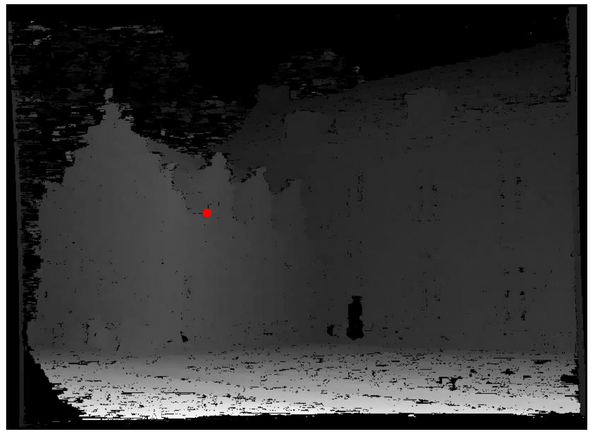


image I´(x´,y´)

$$(x´,y´)=(x+D(x,y),\ y)$$

Review: Camera parameters

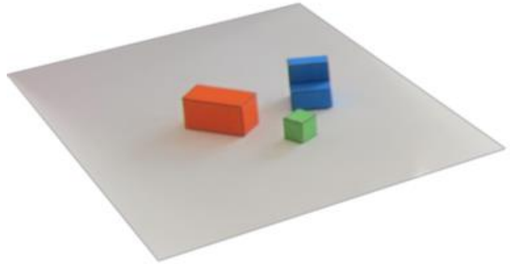
(x, y)

2D image

?

3D world

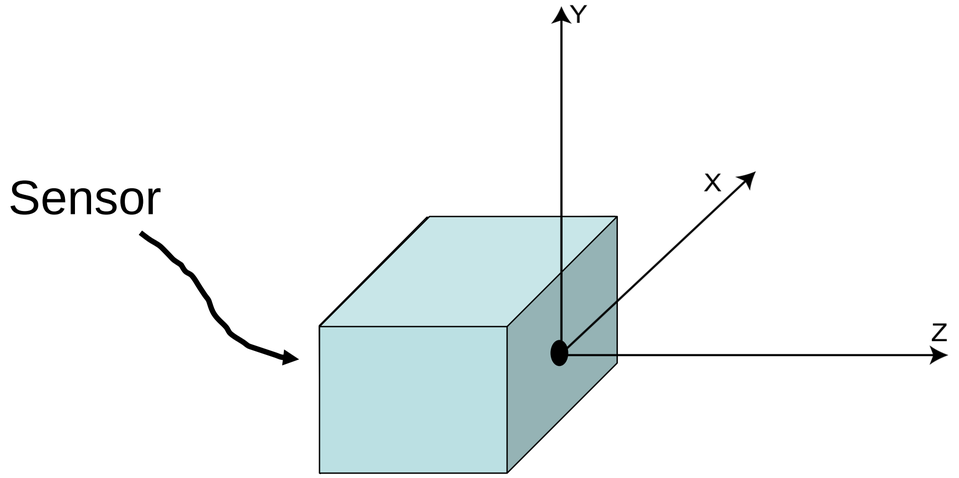
(X, Y, Z)



$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Pixels **World coordinates**

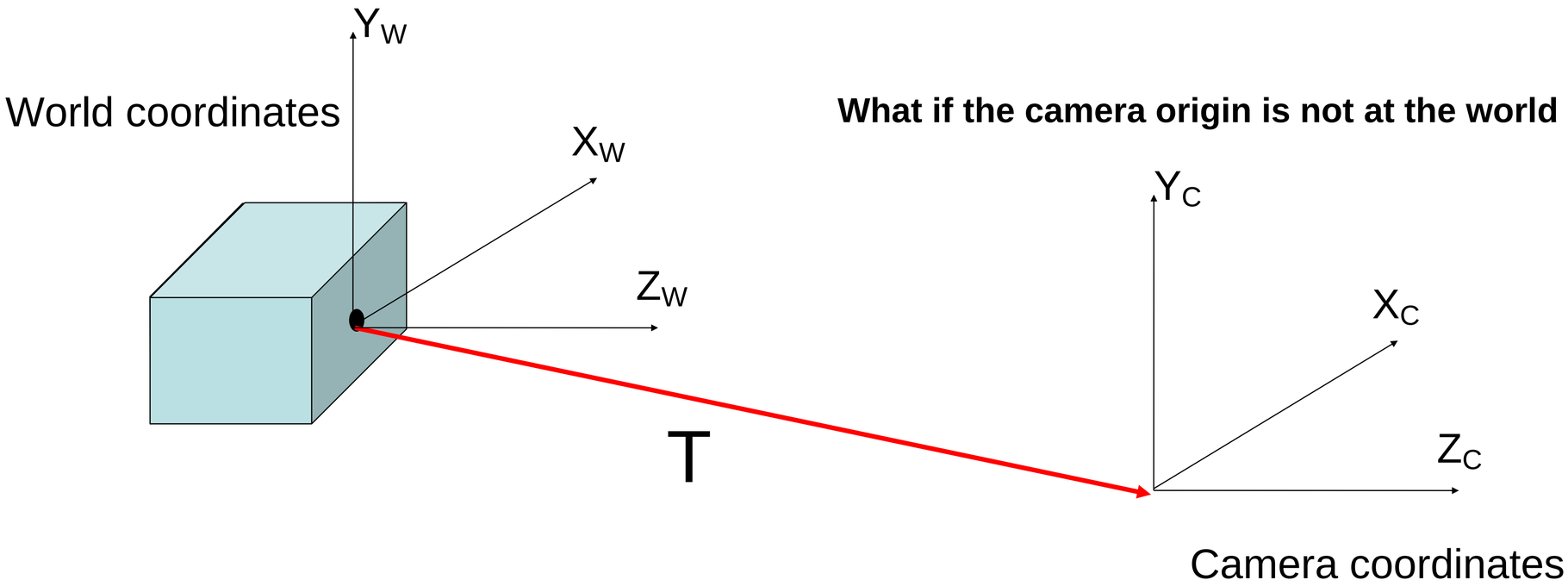
Camera parameters



if pixels are rectangular

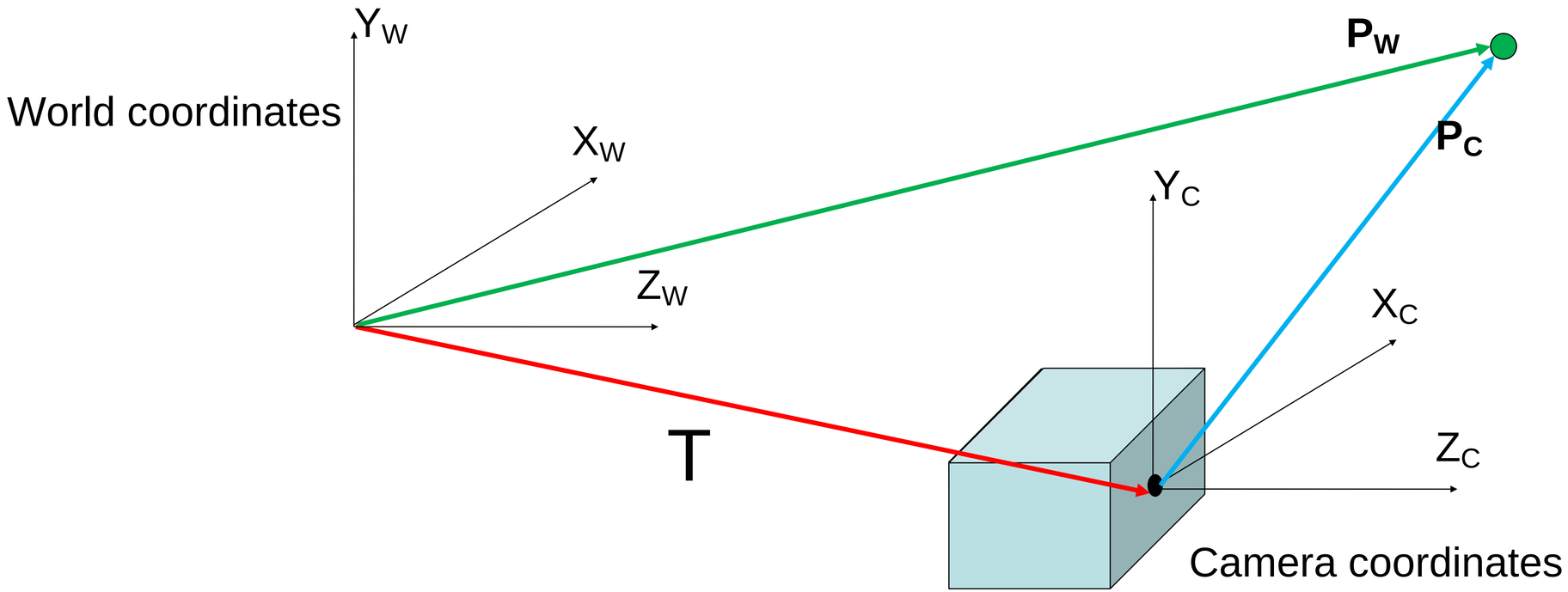
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} aX \\ bY \\ Z \end{bmatrix} \longrightarrow (a\,X/Z,\ b\,Y/Z)$$

Camera parameters



What if the camera origin is not at the world coordinates origin?

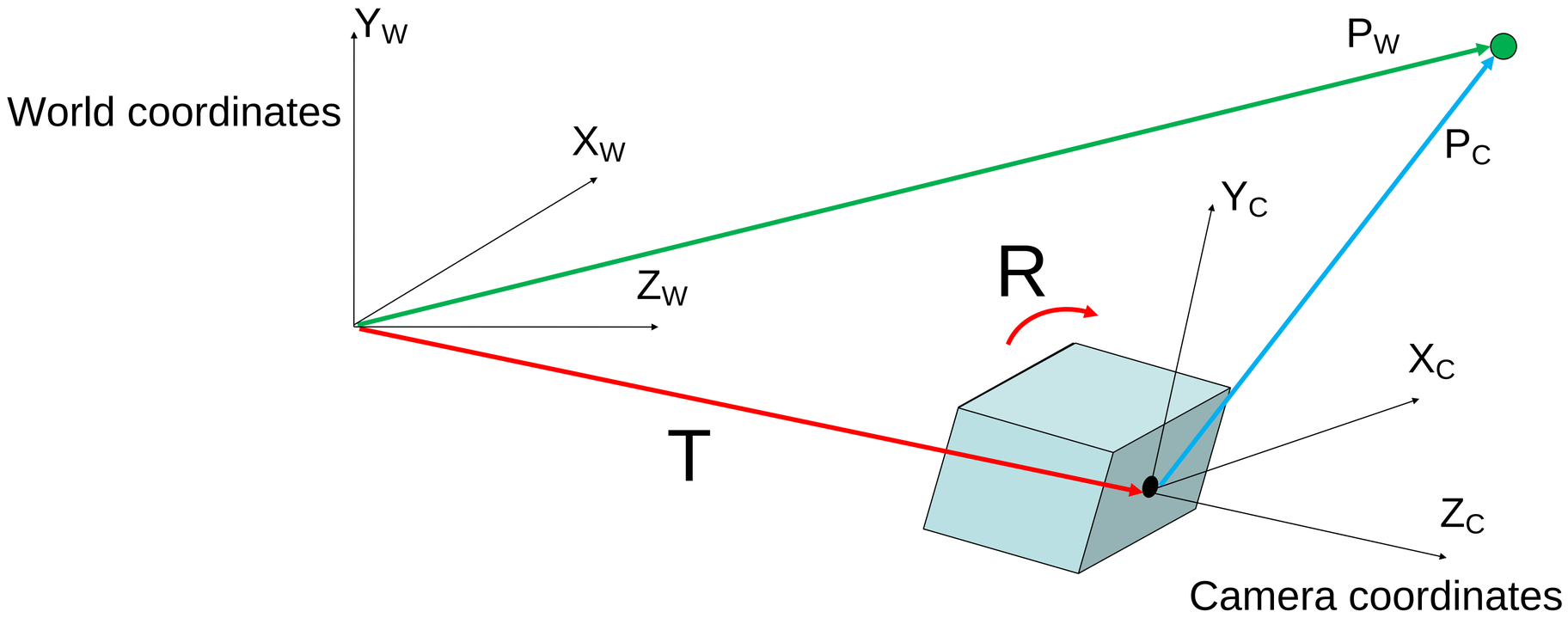
Camera parameters



In heterogeneous coordinates:

$$P_C = P_W - T$$

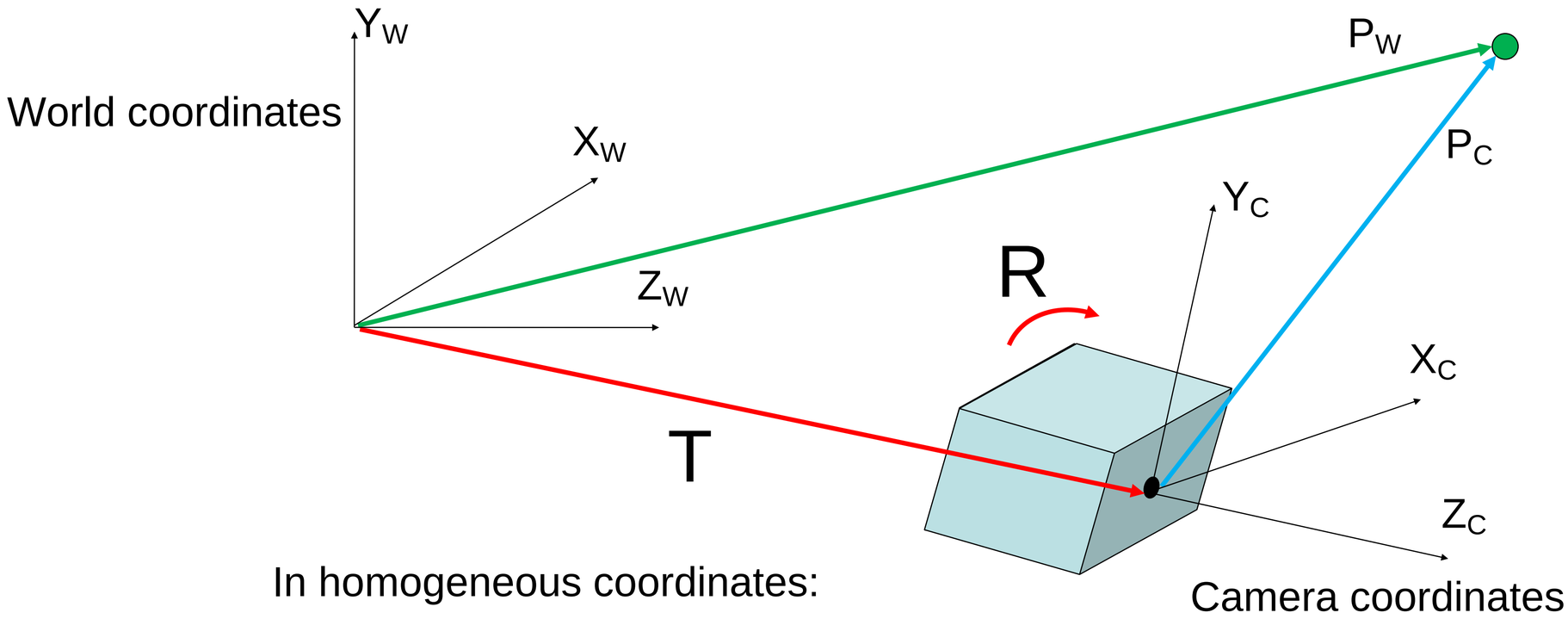
Camera parameters



In heterogeneous coordinates:

$$P_C = R(P_W - T)$$

Camera parameters

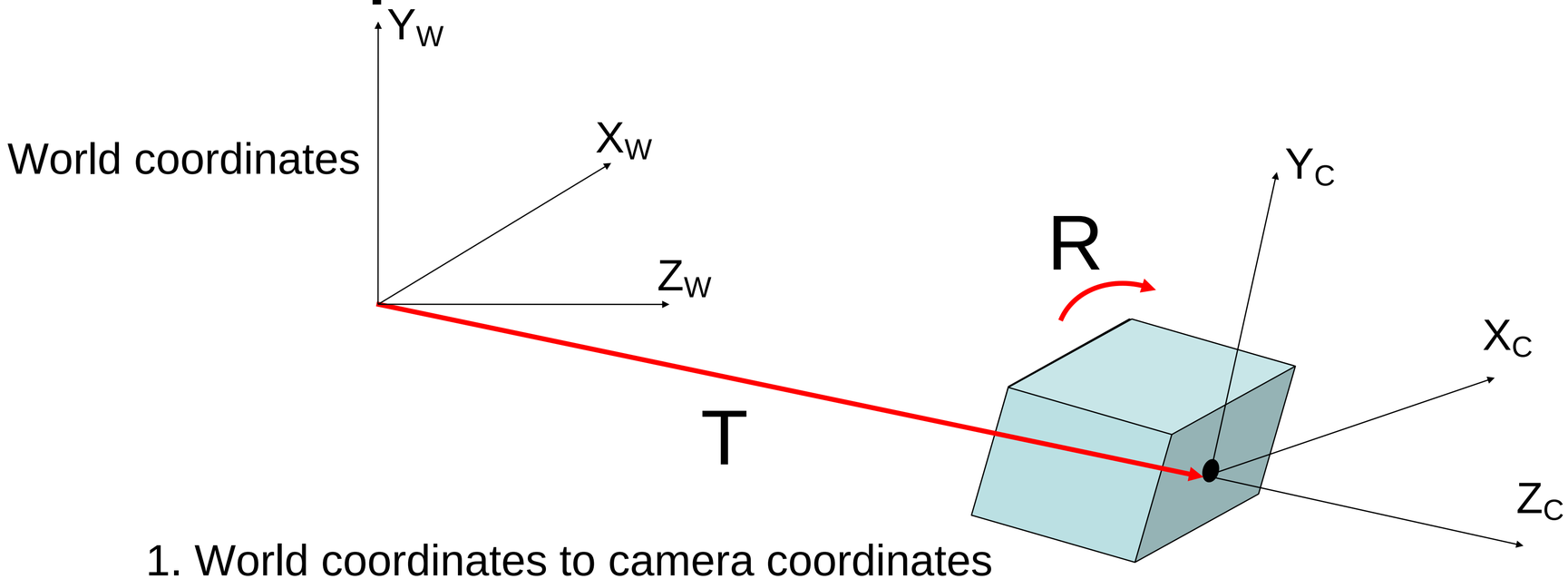


In homogeneous coordinates:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{-RT} \\ \mathbf{0} & 1 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

[3x3] [3x1] [1x3] [1x1]

Camera parameters



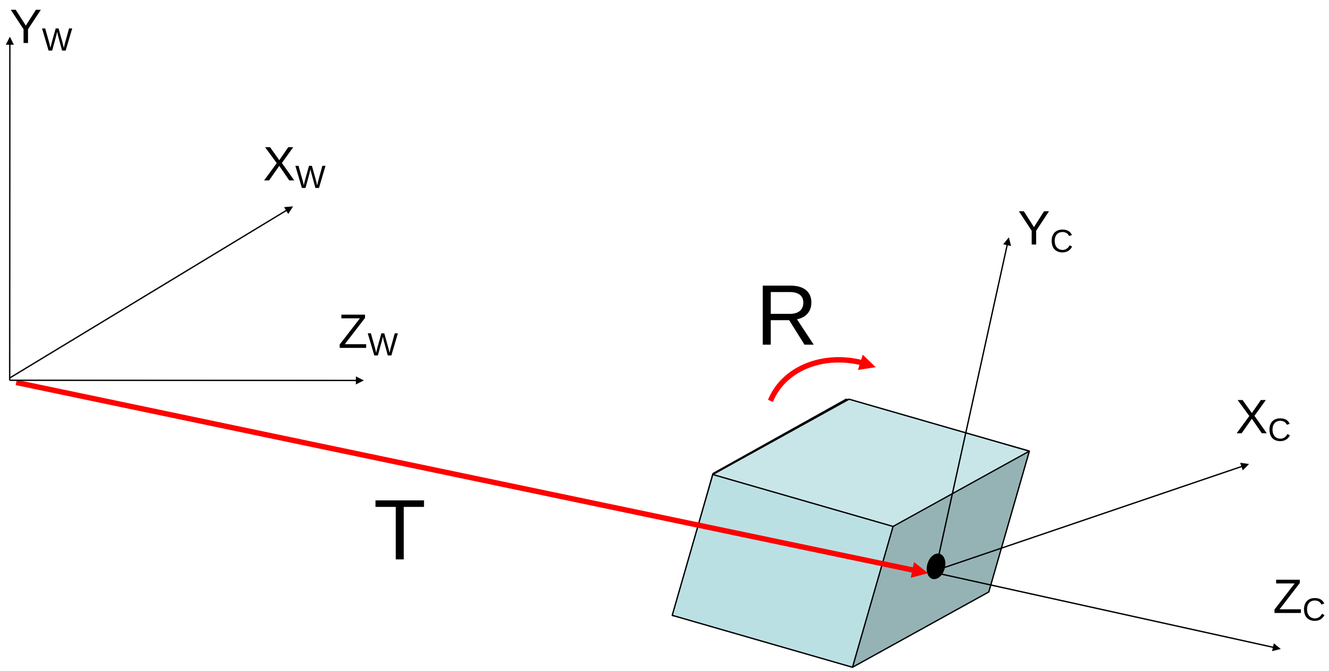
1. World coordinates to camera coordinates

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{-RT} \\ \mathbf{0} & 1 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

2. Camera coordinates to image coordinates (square pixels)

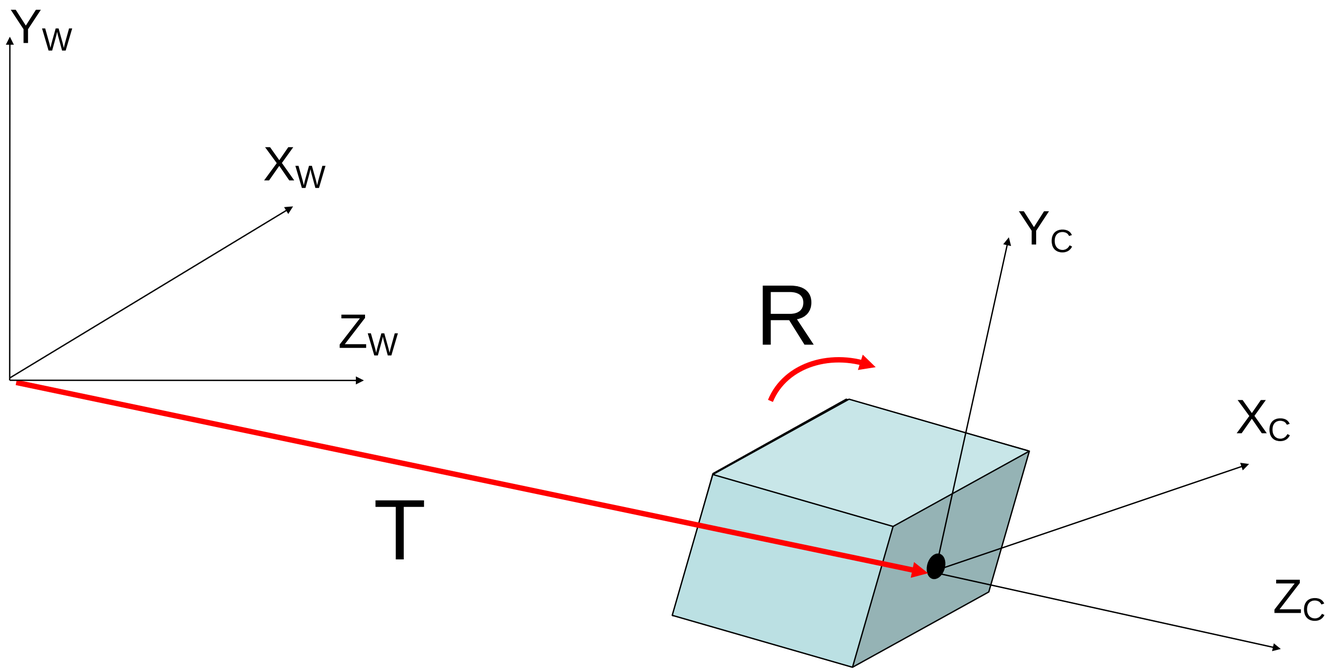
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

Camera parameters



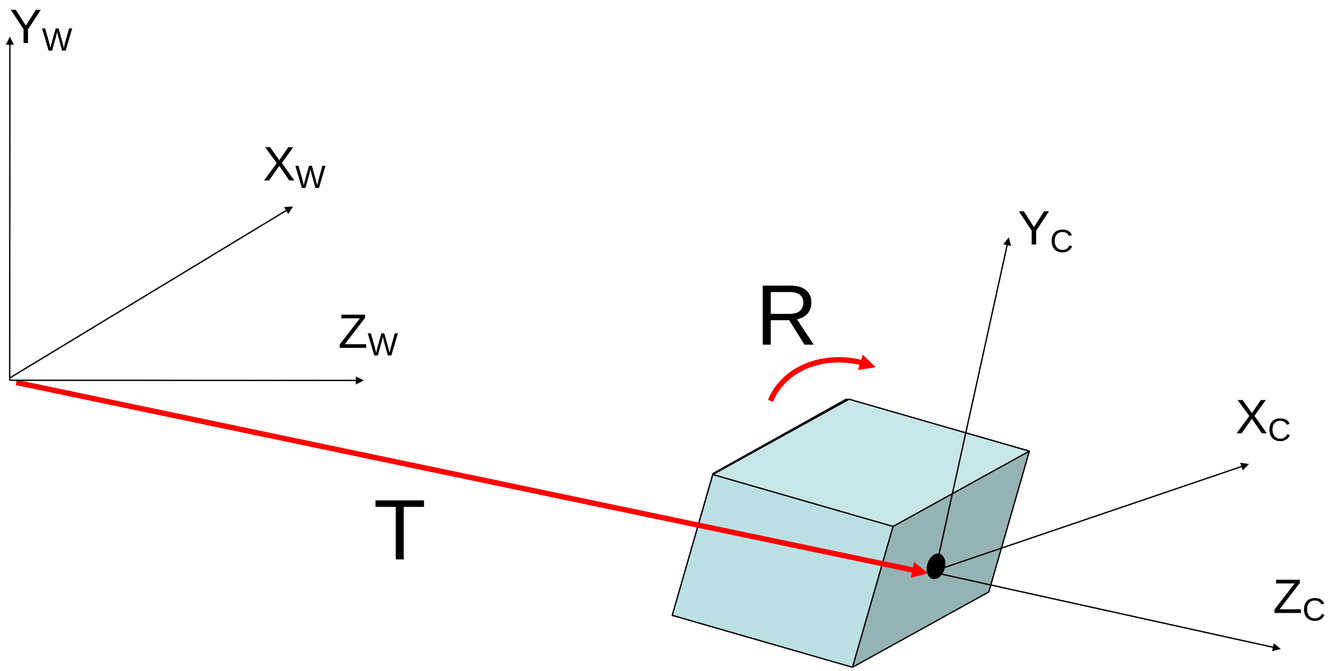
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Camera parameters



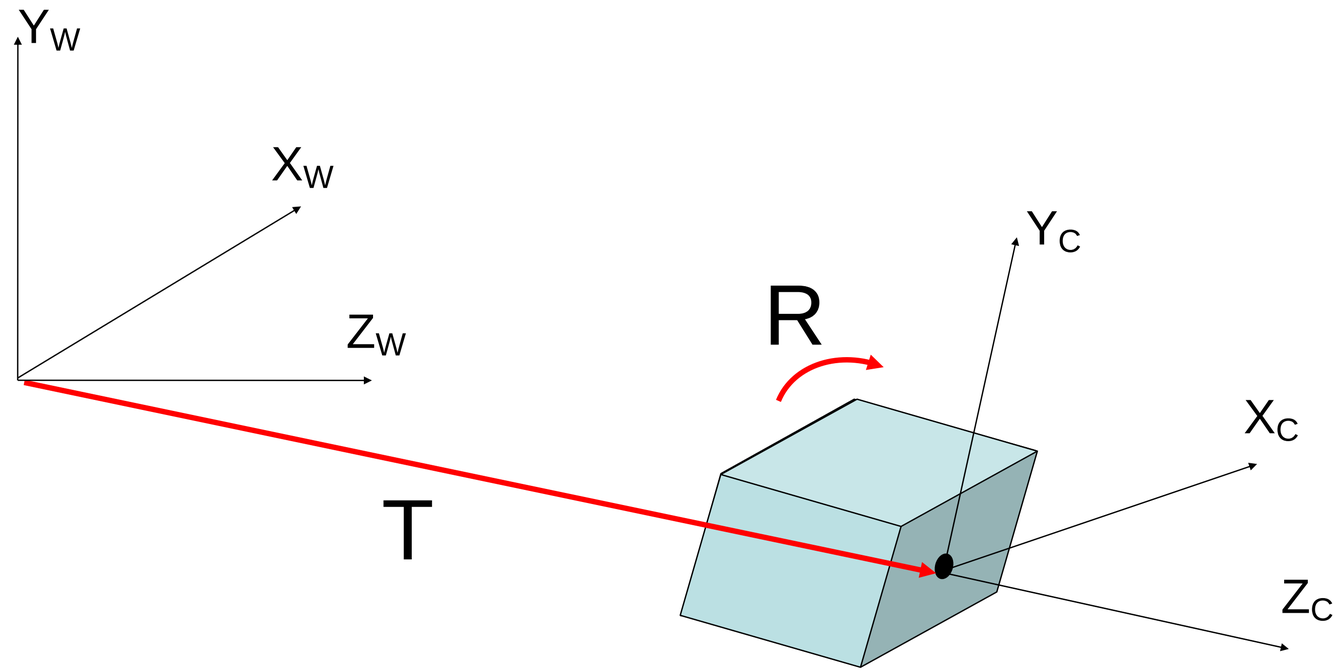
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \overbrace{\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}^{[3x4]} \cdot \overbrace{\begin{bmatrix} \mathbf{R} & \mathbf{-RT} \\ \mathbf{0} & 1 \end{bmatrix}}^{[4x4]} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Camera parameters



$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \underset{[3x3]}{\begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 1 \end{bmatrix}} \cdot \underset{[3x4]}{\begin{bmatrix} \mathbf{R} & \mathbf{-RT} \end{bmatrix}} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

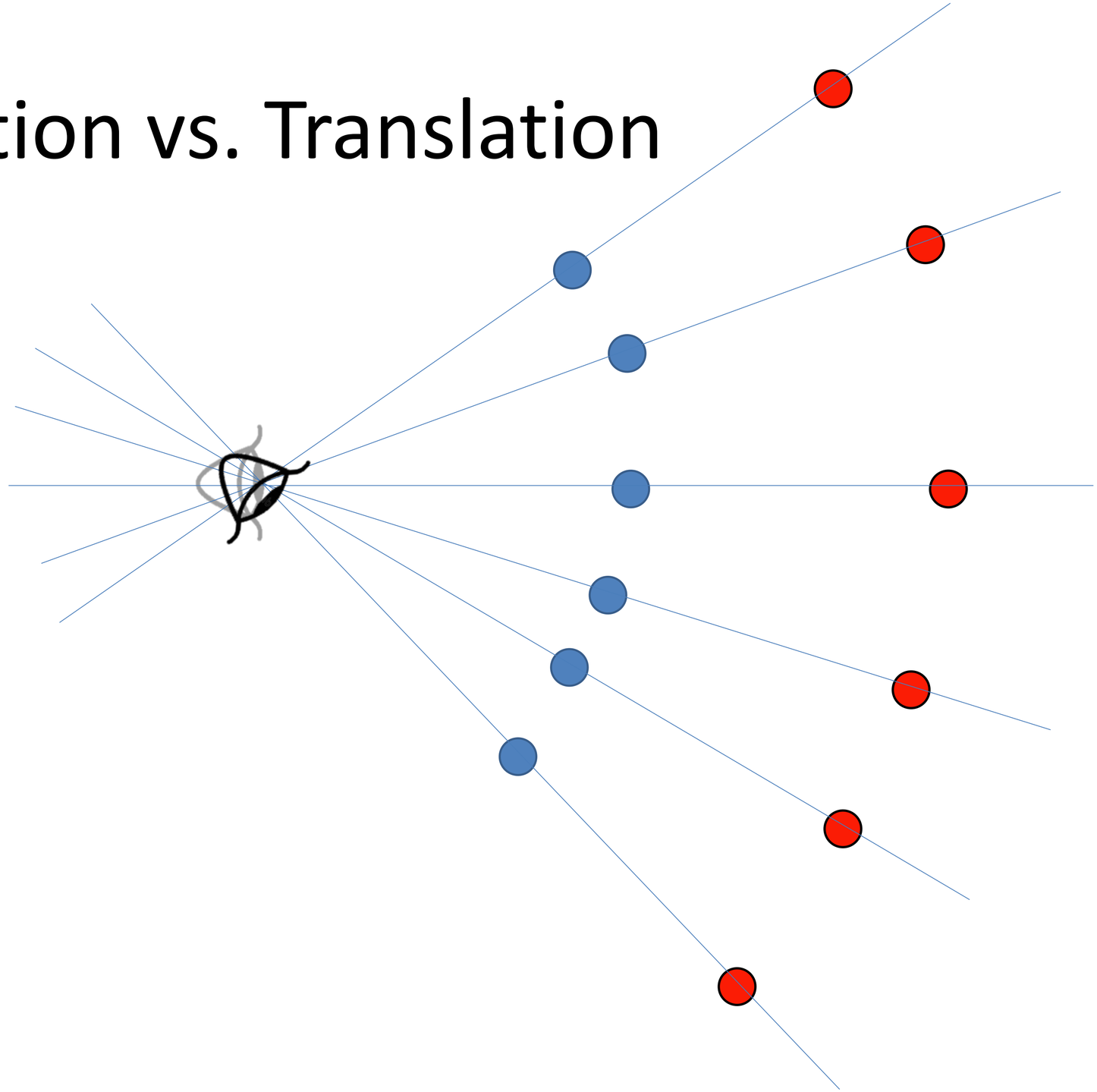
Camera parameters



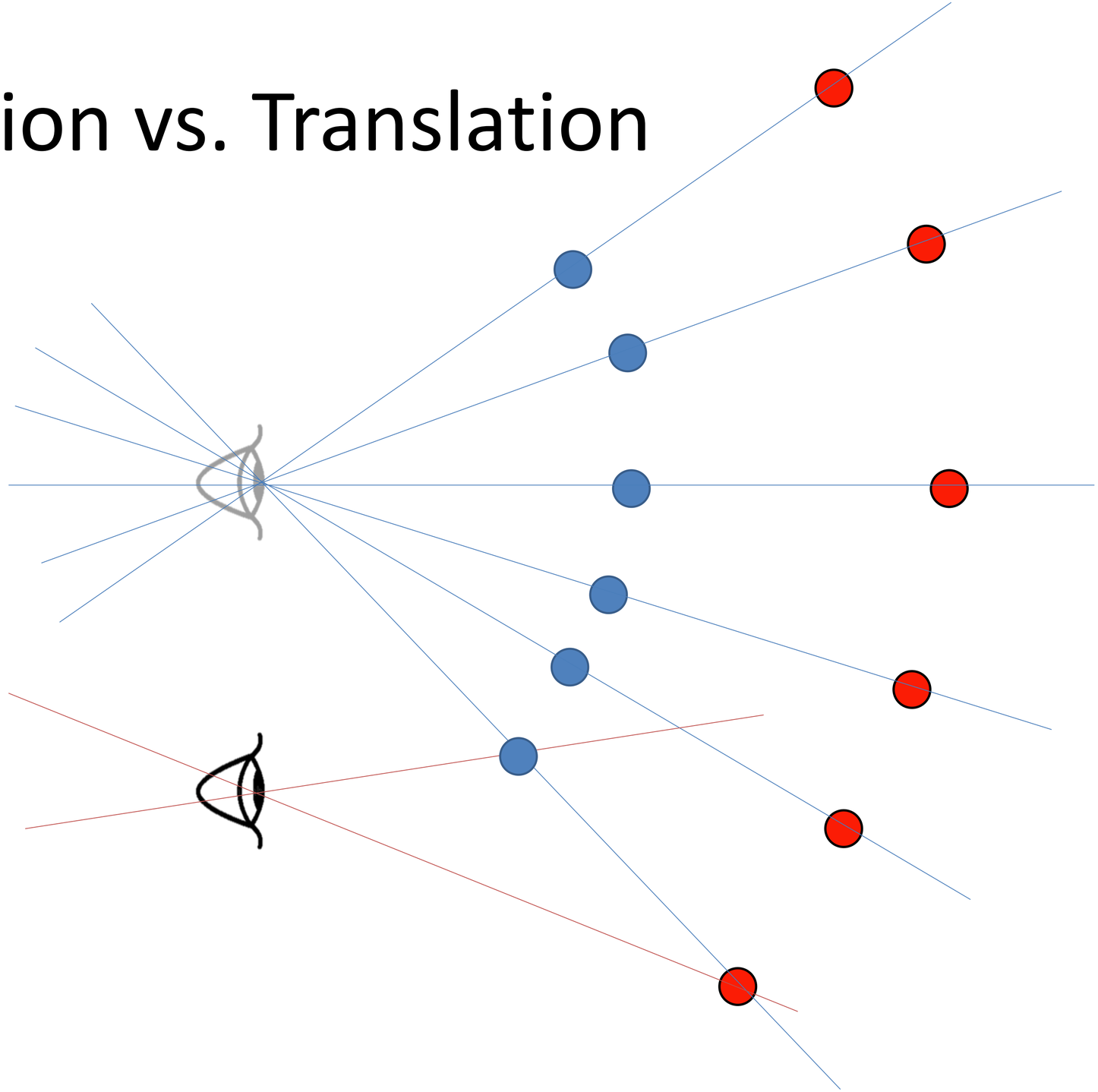
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \underbrace{\overbrace{\begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 1 \end{bmatrix}}^{[3x3]} \cdot \overbrace{\begin{bmatrix} \mathbf{R} \end{bmatrix}}^{[3x3]} \cdot \overbrace{\begin{bmatrix} \mathbf{I} & \mathbf{-T} \end{bmatrix}}^{[3x4]}}_{\text{Extrinsic parameters}} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Intrinsic parameters

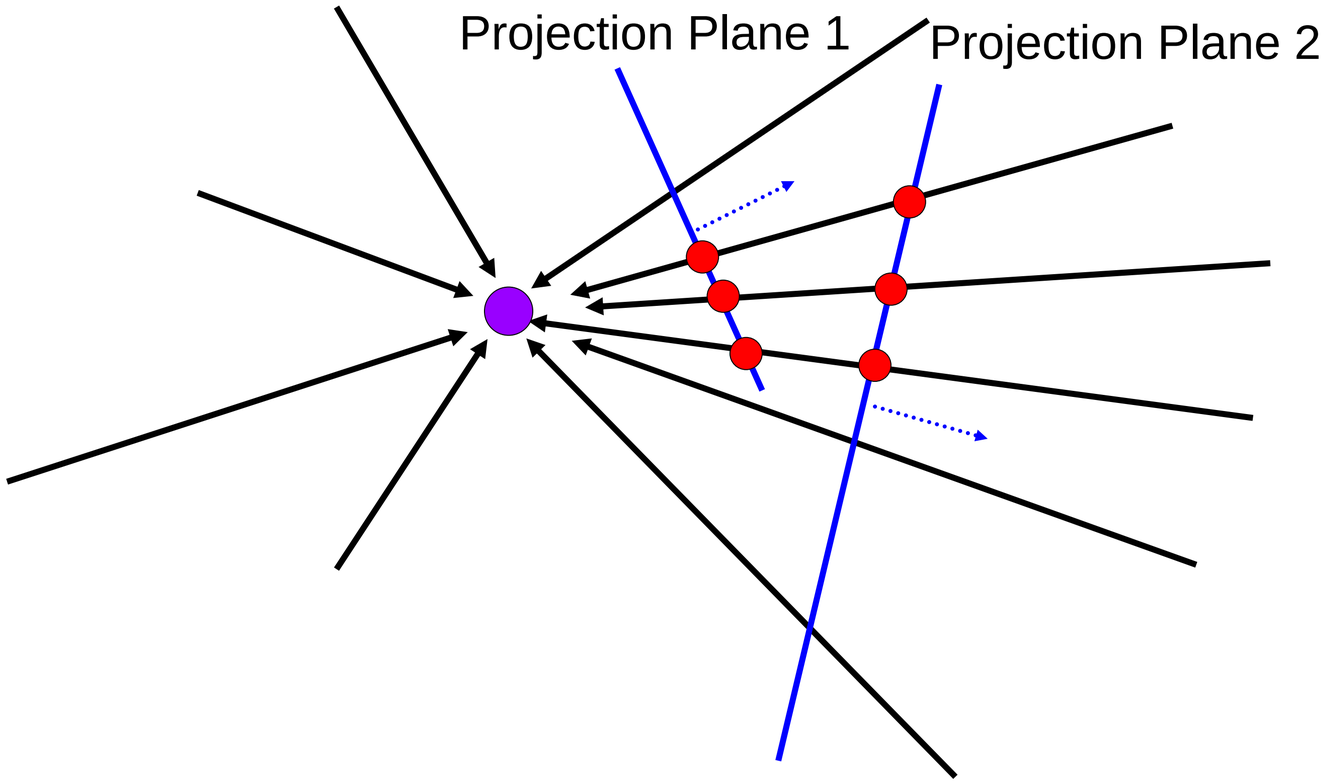
Rotation vs. Translation



Rotation vs. Translation

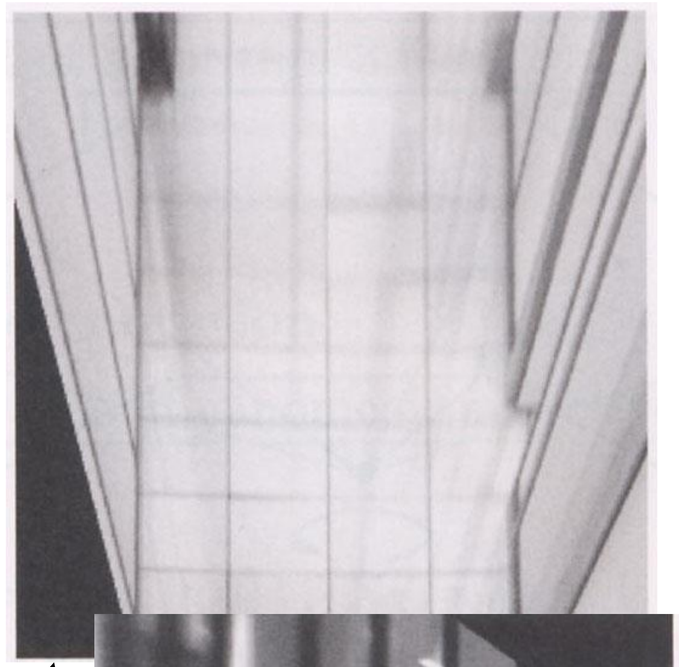


Review: homographies



So, we can generate any synthetic camera view as long as it has **the same center of projection**!

Review Homographies



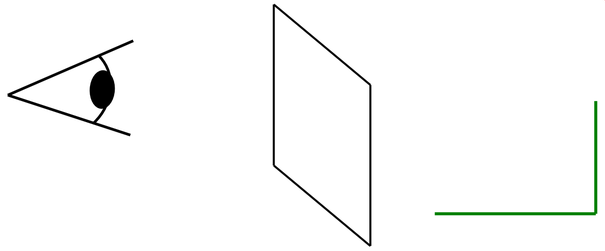
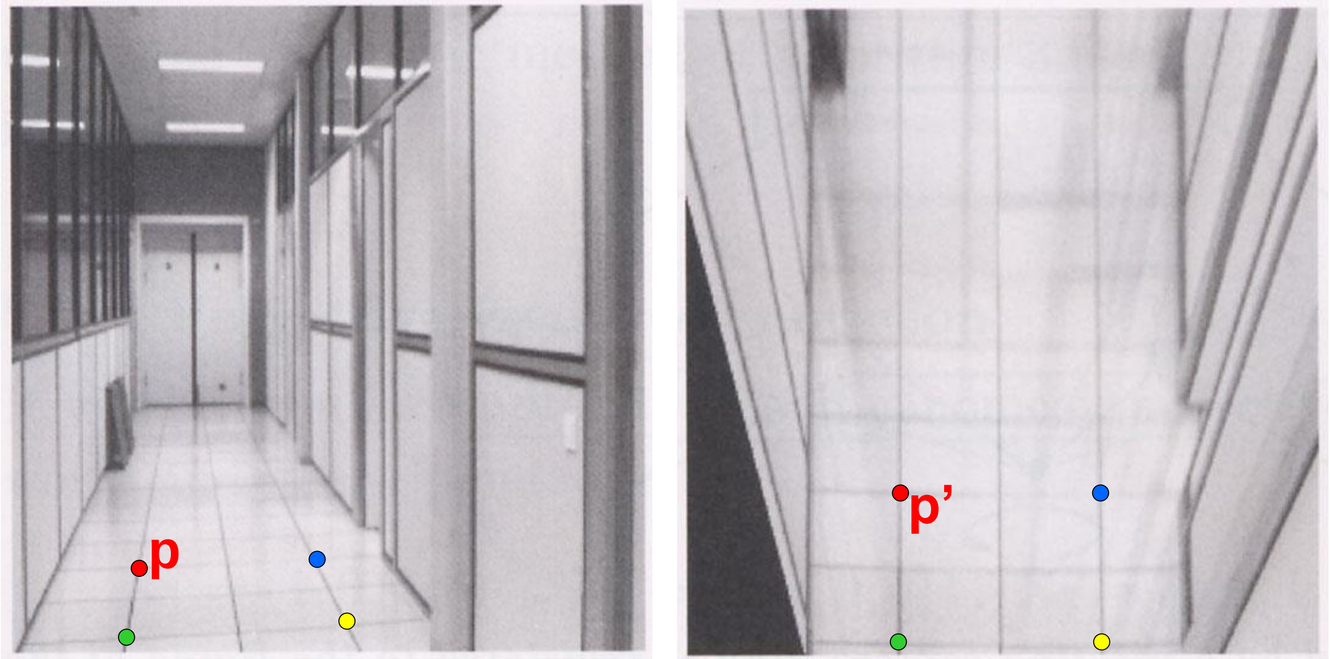


image plane in front

black area
where no pixel
maps to

Image rectification



To unwarp (rectify) an image

- Find the homography **H** given a set of **p** and **p’** pairs
- How many correspondences are needed?

Panorama Stitching

Example: two pictures taken by rotating the camera:

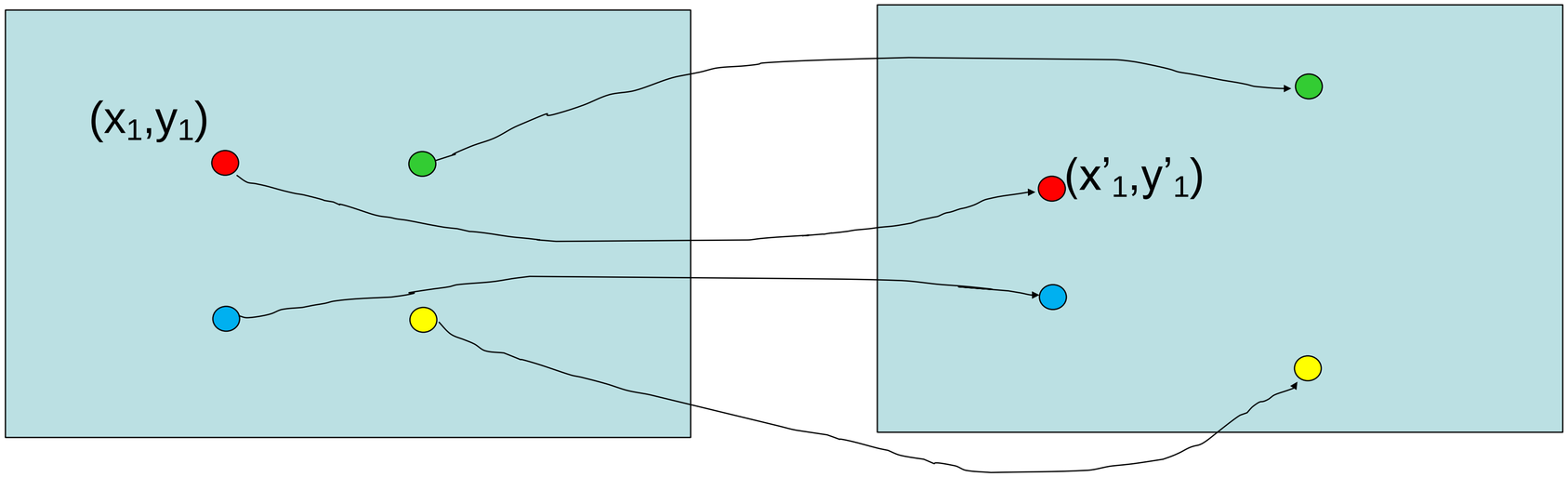
If we try to build a panorama by overlapping them:

Panorama Stitching

Example: two pictures taken by rotating the camera:

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

Homography

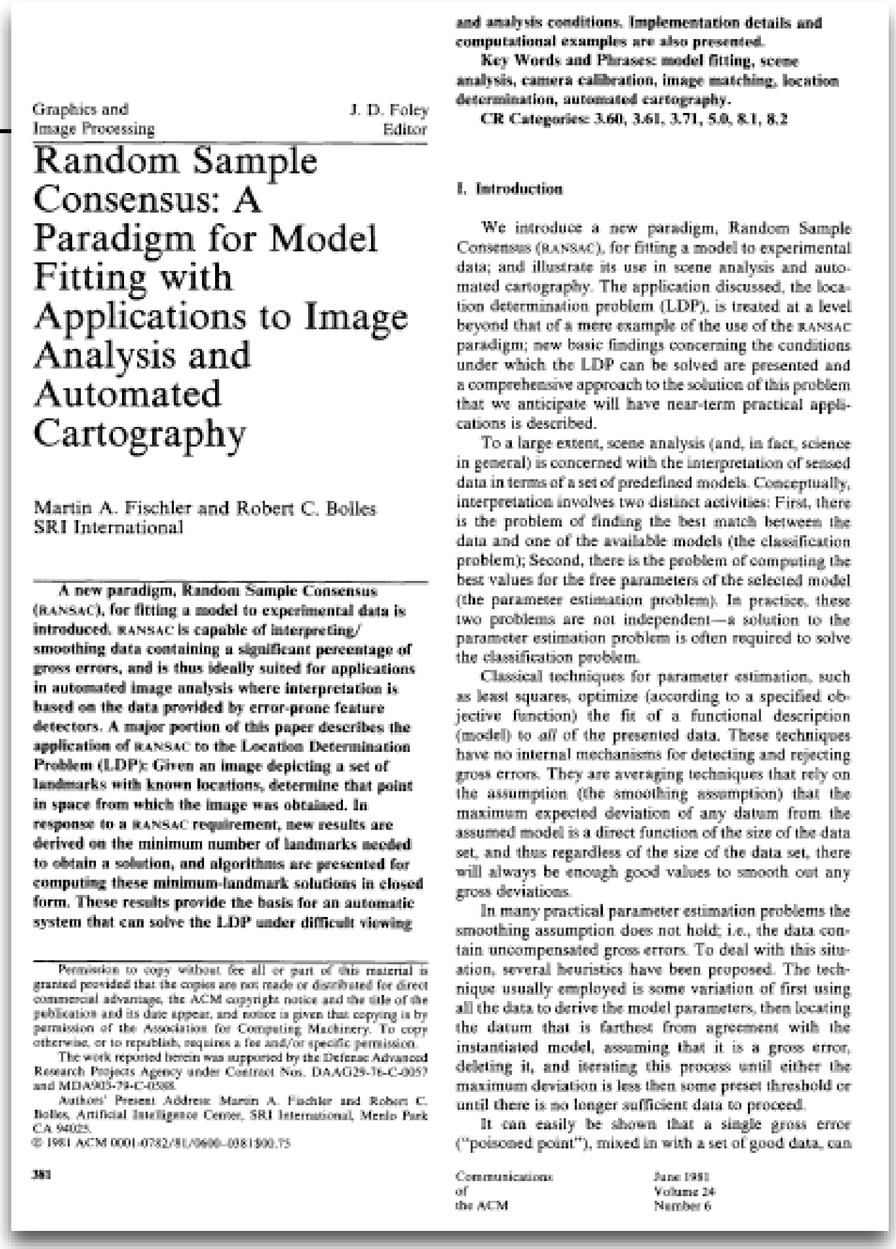


$$\begin{bmatrix} x_1' \\ y_1' \\ w_1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

(note that only 8 DOF are relevant here)

Ransac

M. A. Fischler, R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM, Vol 24, pp 381-395, 1981.



Graphics and Image Processing — J. D. Foley Editor

Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography

Martin A. Fischler and Robert C. Bolles
SRI International

A new paradigm, Random Sample Consensus (RANSAC), for fitting a model to experimental data is introduced. RANSAC is capable of interpreting/smoothing data containing a significant percentage of gross errors, and is thus ideally suited for applications in automated image analysis where interpretation is based on the data provided by error-prone feature detectors. A major portion of this paper describes the application of RANSAC to the Location Determination Problem (LDP): Given an image depicting a set of landmarks with known locations, determine that point in space from which the image was obtained. In response to a RANSAC requirement, new results are derived on the minimum number of landmarks needed to obtain a solution, and algorithms are presented for computing these minimum-landmark solutions in closed form. These results provide the basis for an automatic system that can solve the LDP under difficult viewing and analysis conditions. Implementation details and computational examples are also presented.

Key Words and Phrases: model fitting, scene analysis, camera calibration, image matching, location determination, automated cartography.
CR Categories: 3.60, 3.61, 3.71, 5.0, 8.1, 8.2

The work reported herein was supported by the Defense Advanced Research Projects Agency under Contract Nos. DAAG29-76-C-0057 and MDA903-79-C-0588.

Authors' Present Address: Martin A. Fischler and Robert C. Bolles, Artificial Intelligence Center, SRI International, Menlo Park CA 94025.

1. Introduction

We introduce a new paradigm, Random Sample Consensus (RANSAC), for fitting a model to experimental data; and illustrate its use in scene analysis and automated cartography. The application discussed, the location determination problem (LDP), is treated at a level beyond that of a mere example of the use of the RANSAC paradigm; new basic findings concerning the conditions under which the LDP can be solved are presented and a comprehensive approach to the solution of this problem that we anticipate will have near-term practical applications is described.

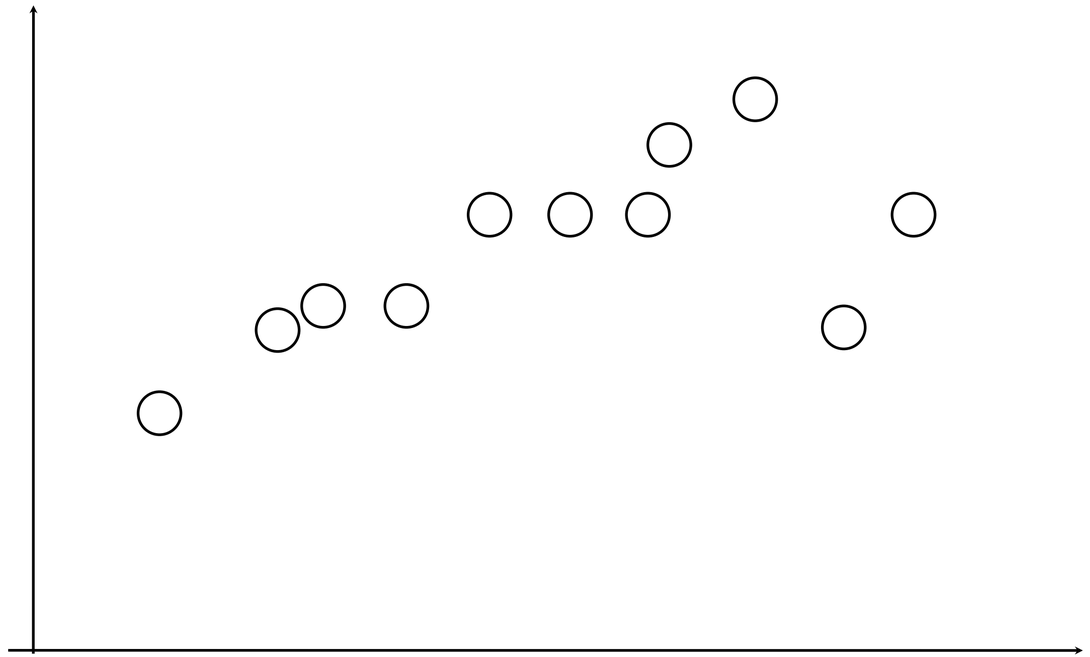
To a large extent, scene analysis (and, in fact, science in general) is concerned with the interpretation of sensed data in terms of a set of predefined models. Conceptually, interpretation involves two distinct activities: First, there is the problem of finding the best match between the data and one of the available models (the classification problem); Second, there is the problem of computing the best values for the free parameters of the selected model (the parameter estimation problem). In practice, these two problems are not independent—a solution to the parameter estimation problem is often required to solve the classification problem.

Classical techniques for parameter estimation, such as least squares, optimize (according to a specified objective function) the fit of a functional description (model) to *all* of the presented data. These techniques have no internal mechanisms for detecting and rejecting gross errors. They are averaging techniques that rely on the assumption (the smoothing assumption) that the maximum expected deviation of any datum from the assumed model is a direct function of the size of the data set, and thus regardless of the size of the data set, there will always be enough good values to smooth out any gross deviations.

In many practical parameter estimation problems the smoothing assumption does not hold; i.e., the data contain uncompensated gross errors. To deal with this situation, several heuristics have been proposed. The technique usually employed is some variation of first using all the data to derive the model parameters, then locating the datum that is farthest from agreement with the instantiated model, assuming that it is a gross error, deleting it, and iterating this process until either the maximum deviation is less then some preset threshold or until there is no longer sufficient data to proceed.

It can easily be shown that a single gross error ("poisoned point"), mixed in with a set of good data, can

Simple example: fit a line

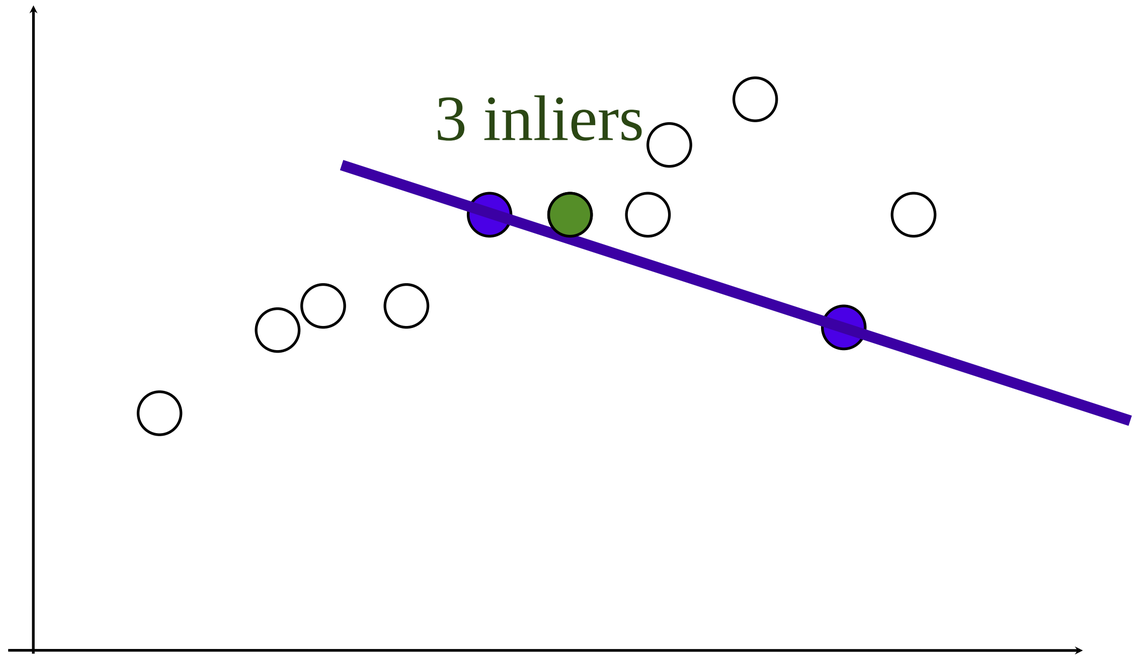


Simple example: fit a line

Pick 2 points

Fit line

Count inliers

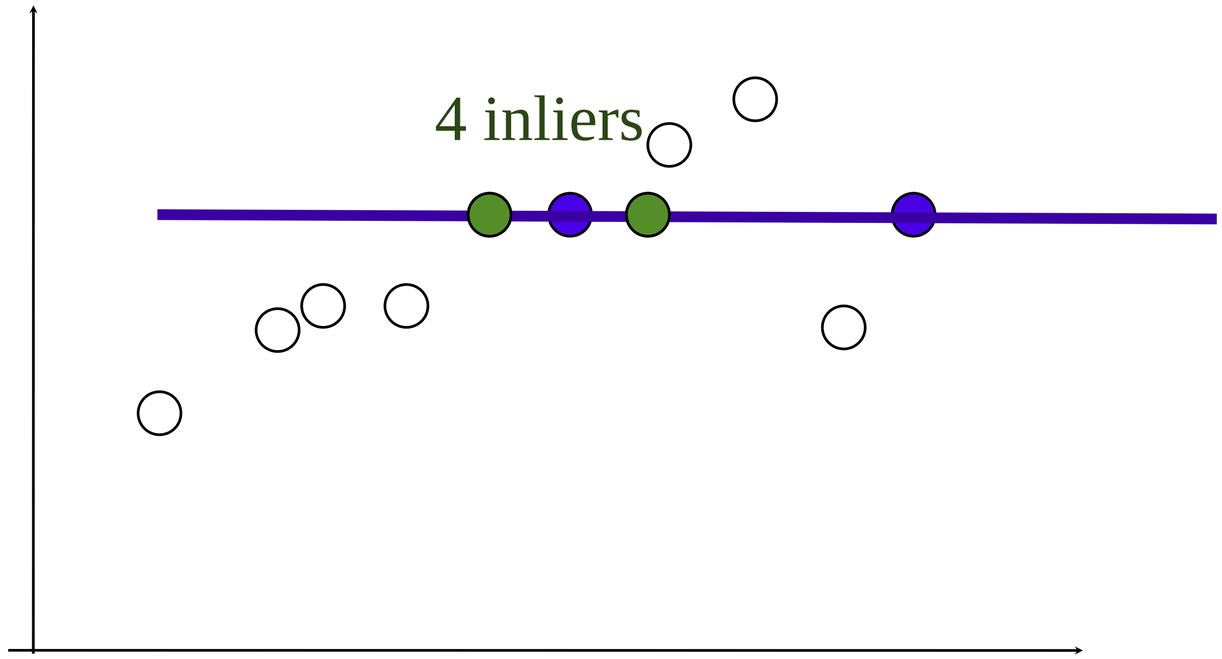


Simple example: fit a line

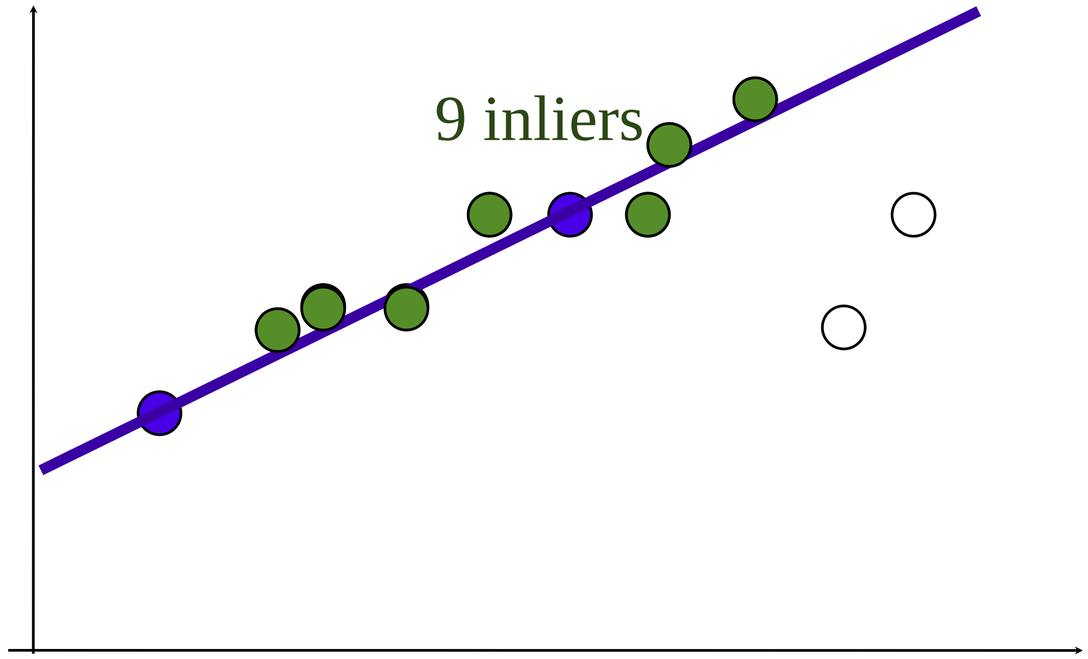
Pick 2 points

Fit line

Count inliers

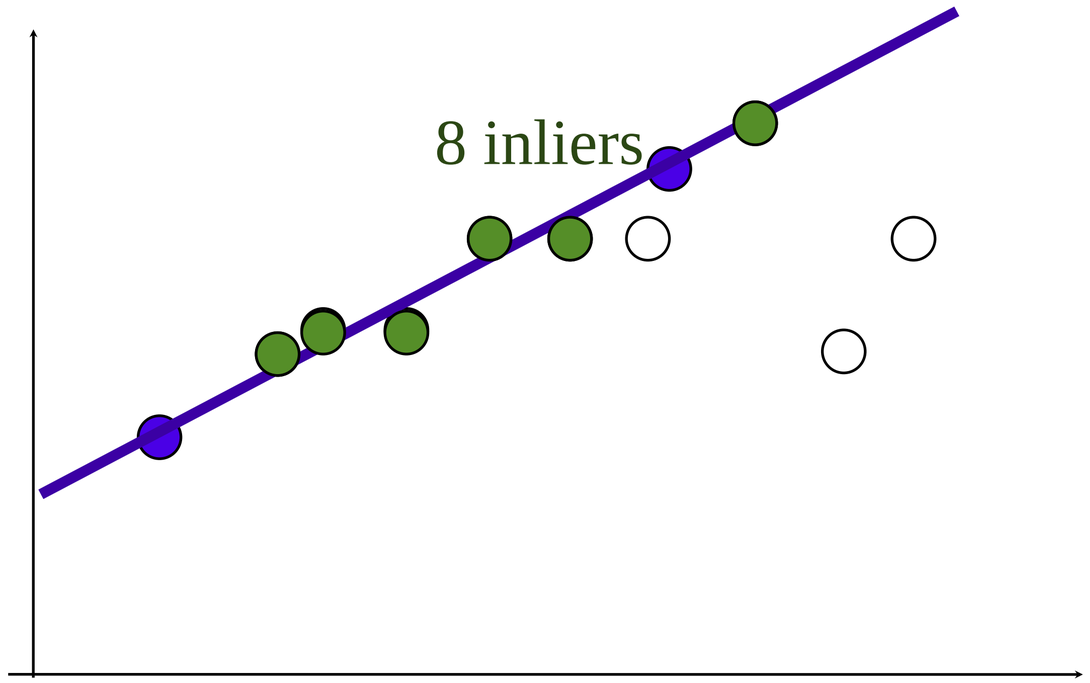


Simple example: fit a line



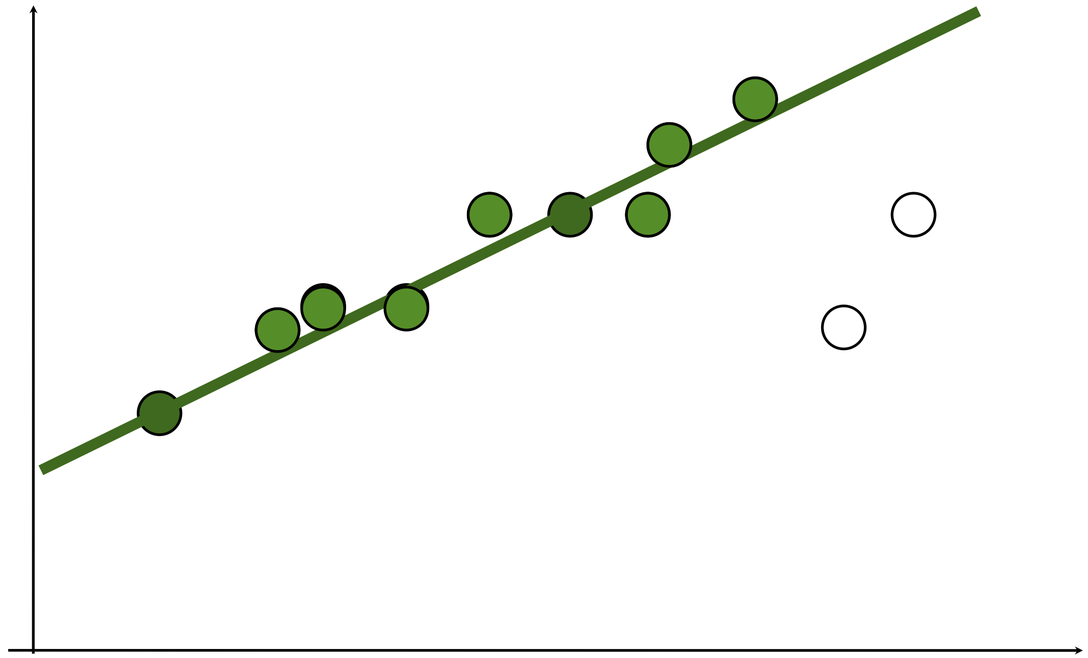
- Pick 2 points
- Fit line
- Count inliers

Simple example: fit a line



- Pick 2 points
- Fit line
- Count inliers

Simple example: fit a line



Use biggest set of inliers

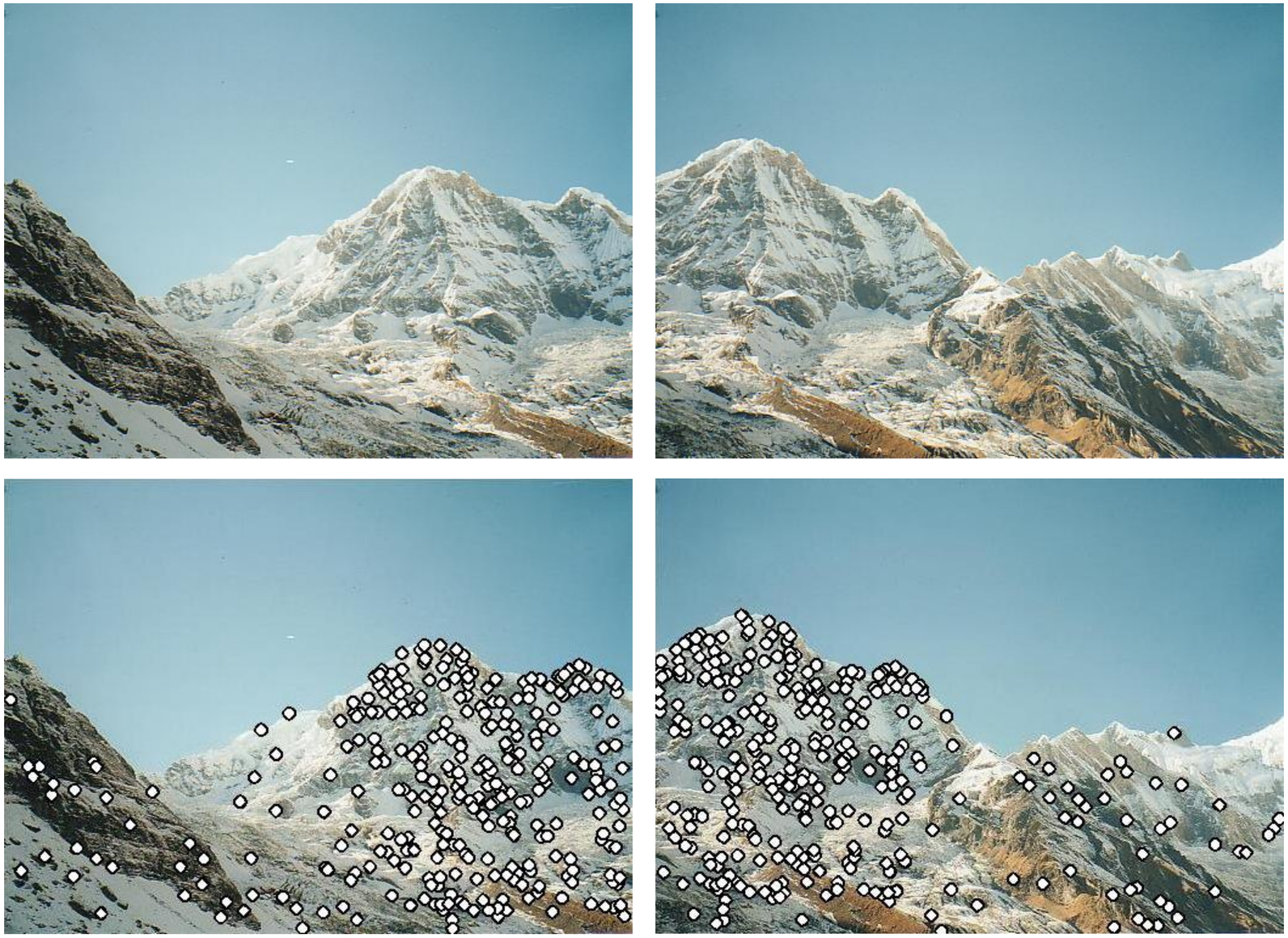
Do least-square fit

RANSAC for estimating homography

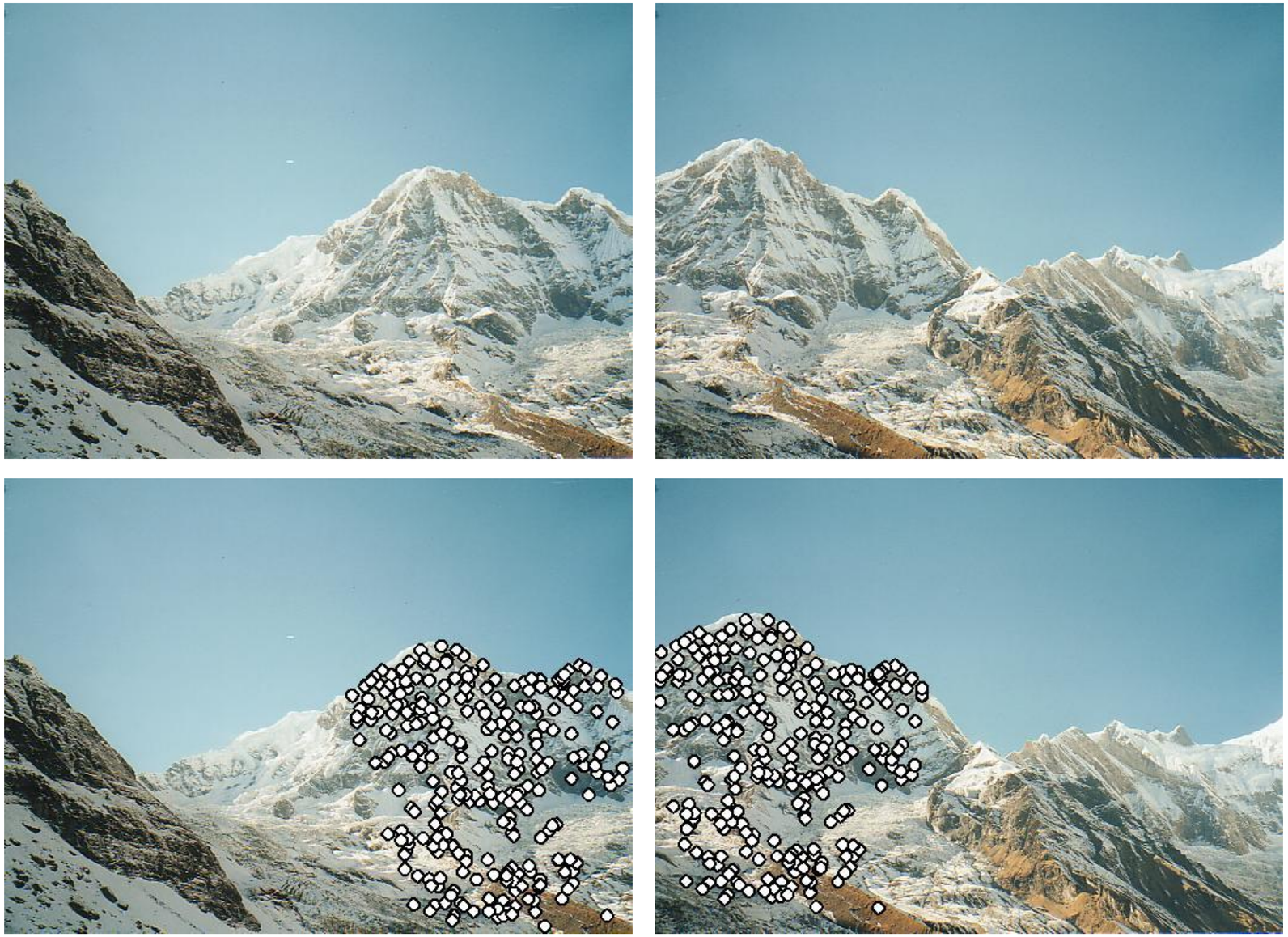
RANSAC loop:

1. Select four feature pairs (at random)
2. Compute homography H (exact)
3. Compute *inliers* where $||p_i', \boldsymbol{H}\ p_i|| < \varepsilon$
4. Keep largest set of inliers
5. Re-compute least-squares H estimate using all of the inliers

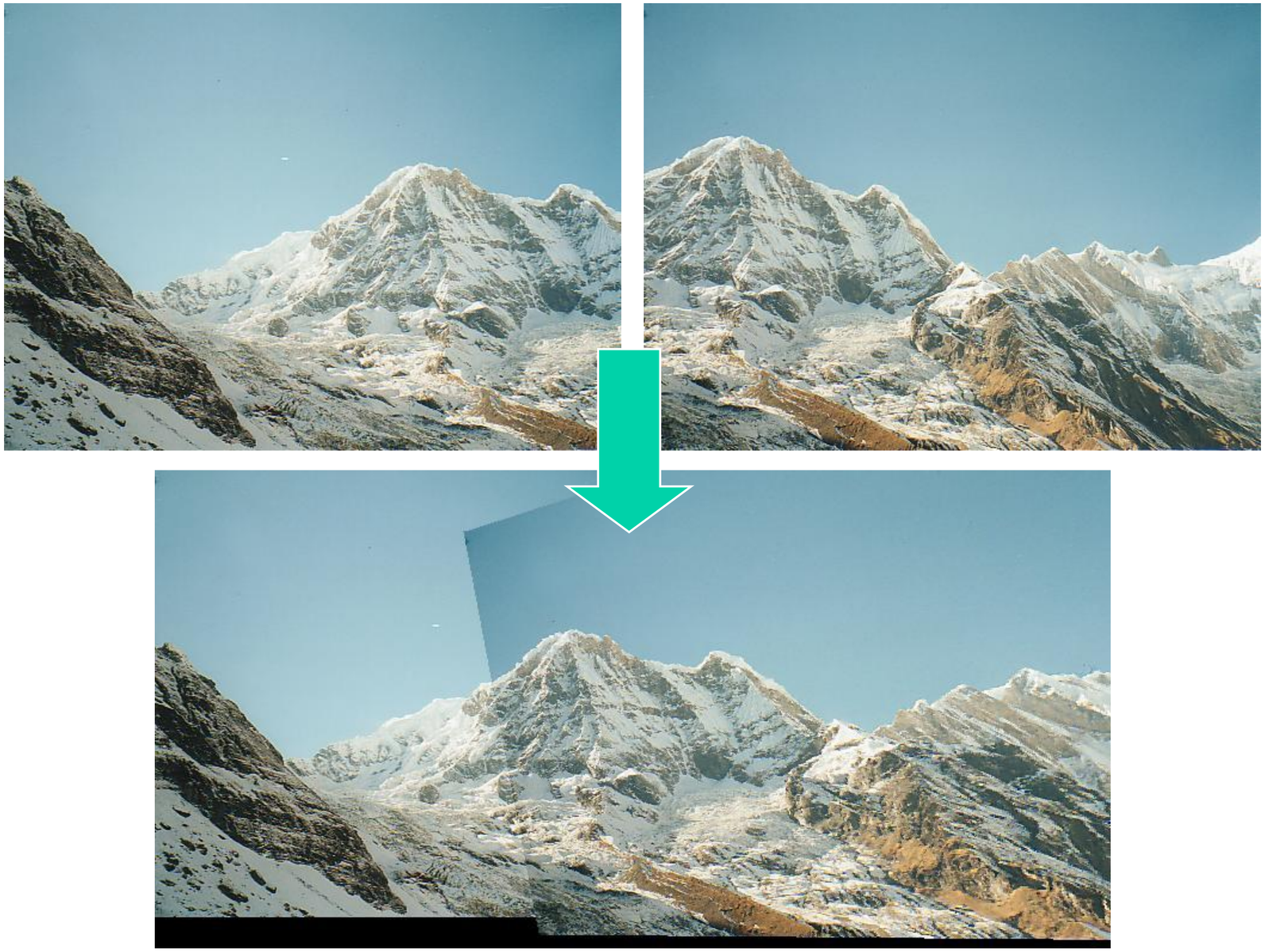
RANSAC for Homography



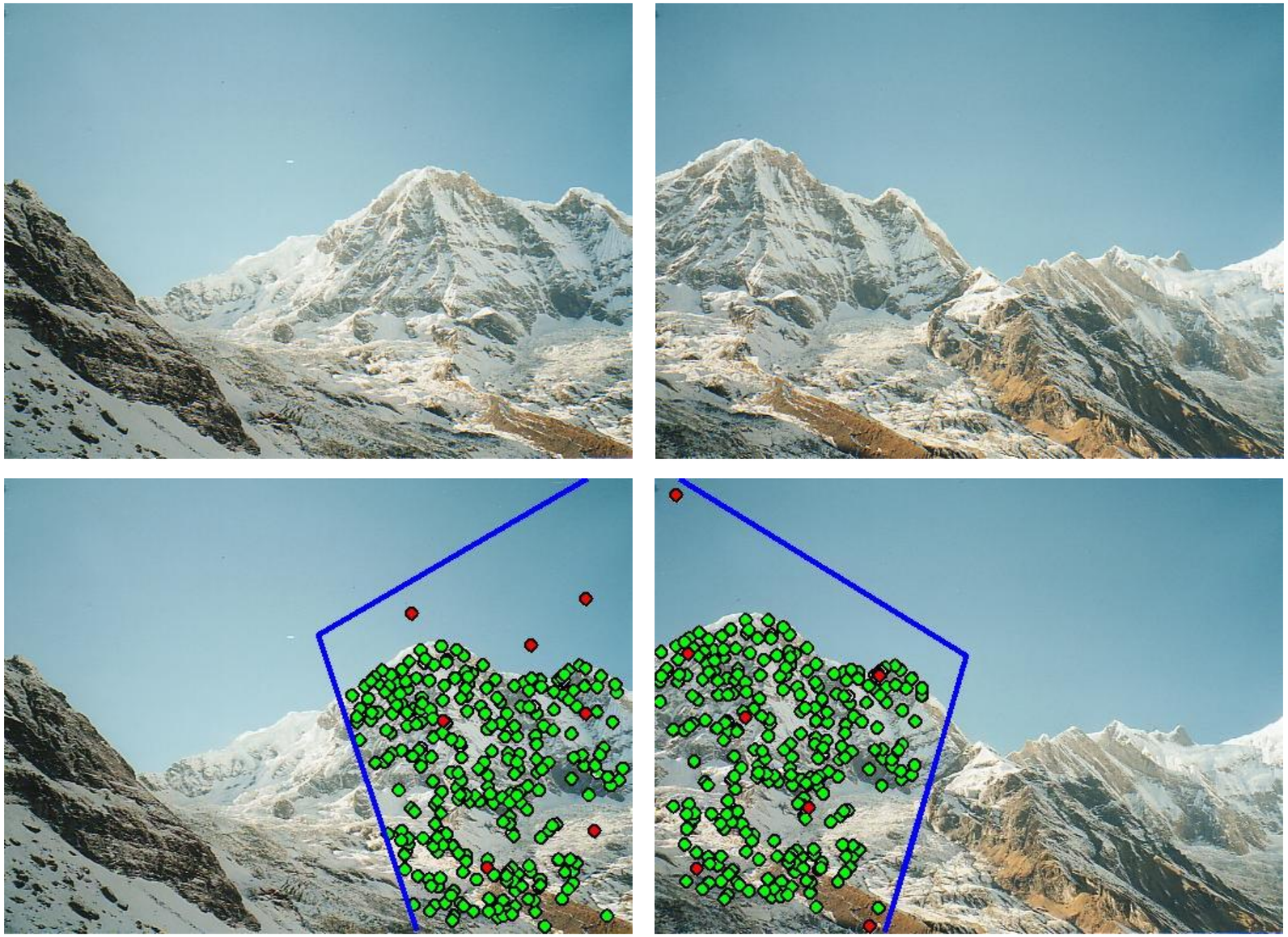
RANSAC for Homography



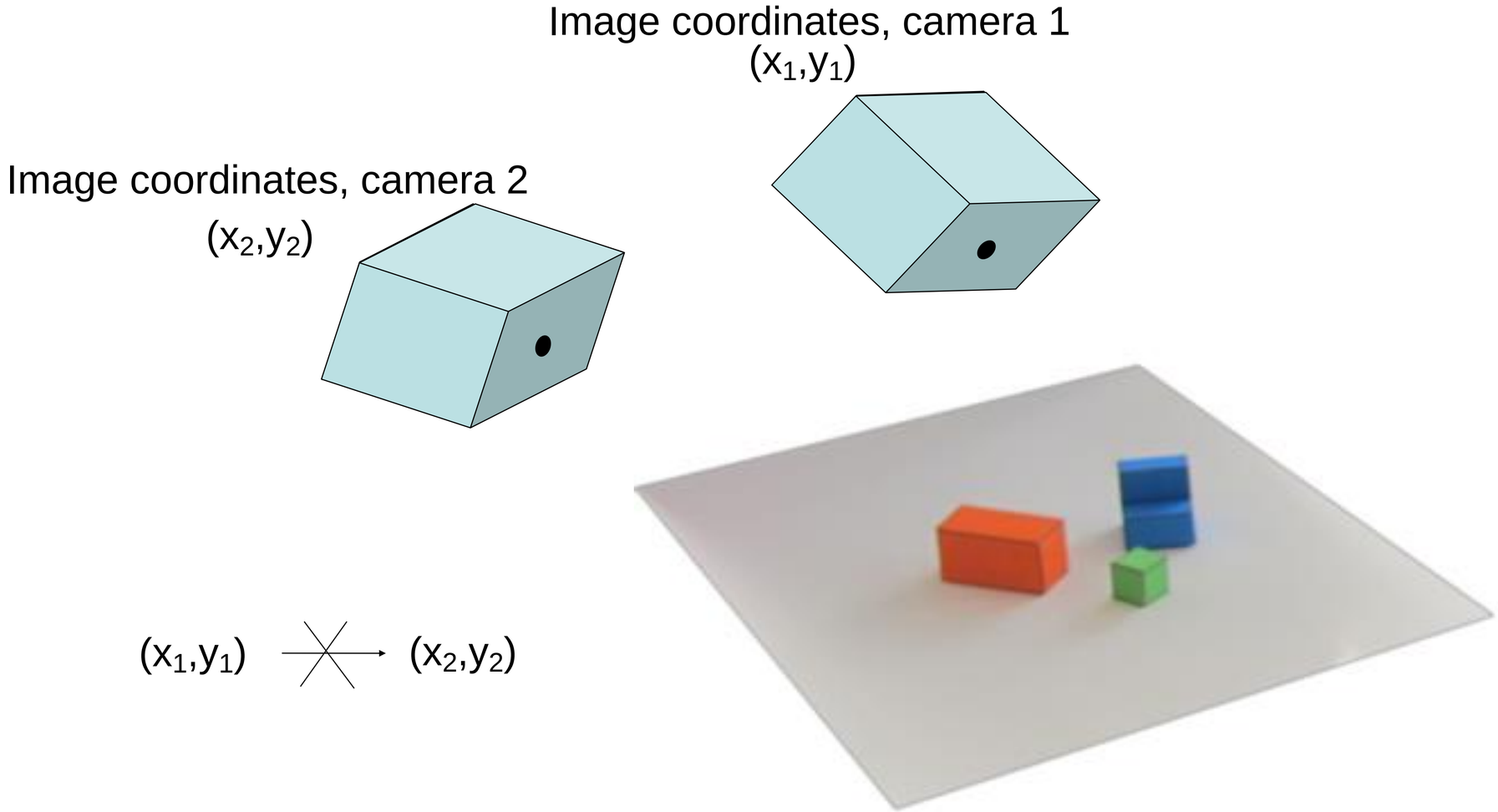
RANSAC for Homography



Probabilistic model for verification



Mapping one camera into another



In general, we can not find a transformation from x_1 to x_2. It requires knowing the 3D coordinates of each corresponding point.

(The general mapping has to depend on 3D shape, otherwise we would learn no information from the 2nd image of a stereo camera!)

Mapping one camera into another

Image coordinates, camera 1
(x_1,y_1)

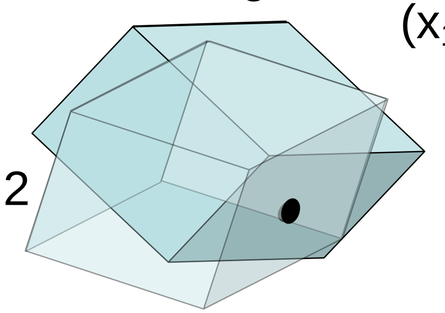
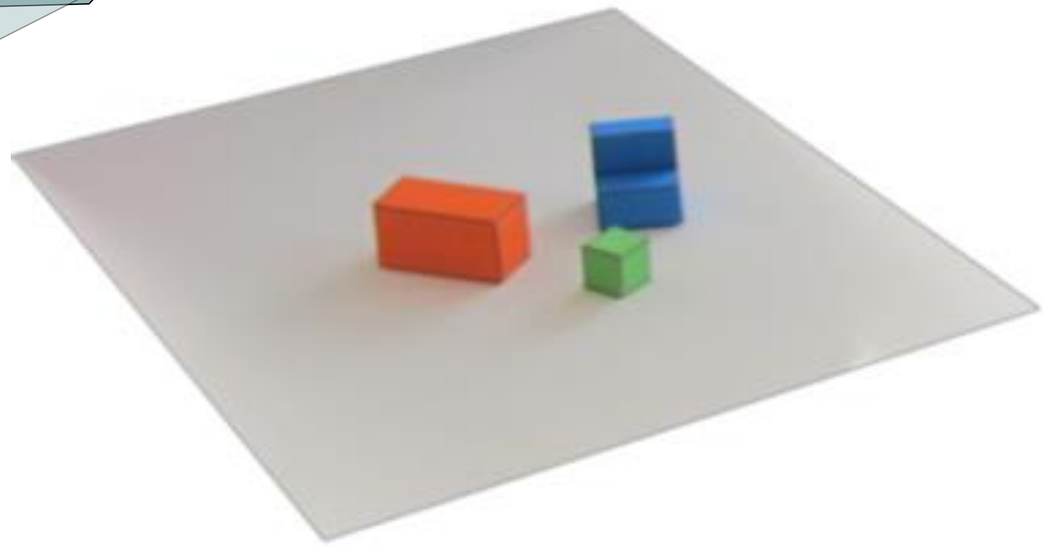


Image coordinates, camera 2
(x_2,y_2)



$(x_1,y_1) \longrightarrow (x_2,y_2)$

Homography

Mapping one camera into another

Image coordinates, camera 1

(x_1,y_1)

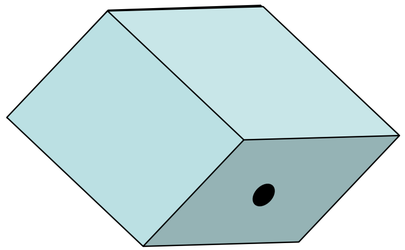
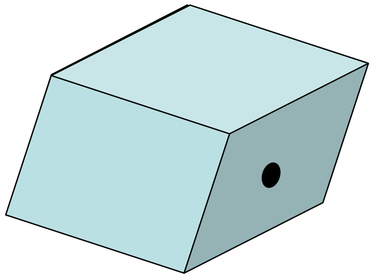
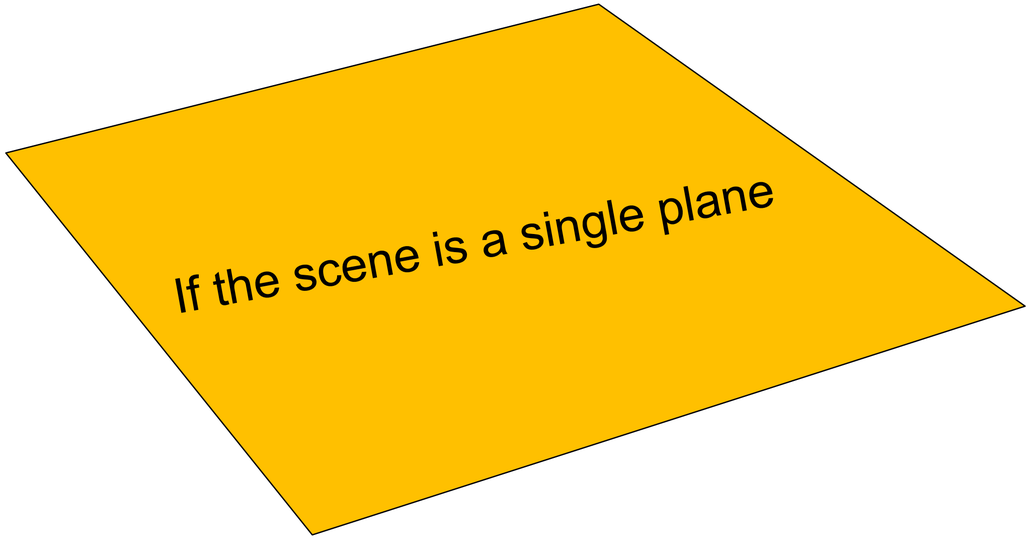


Image coordinates, camera 2

(x_2,y_2)



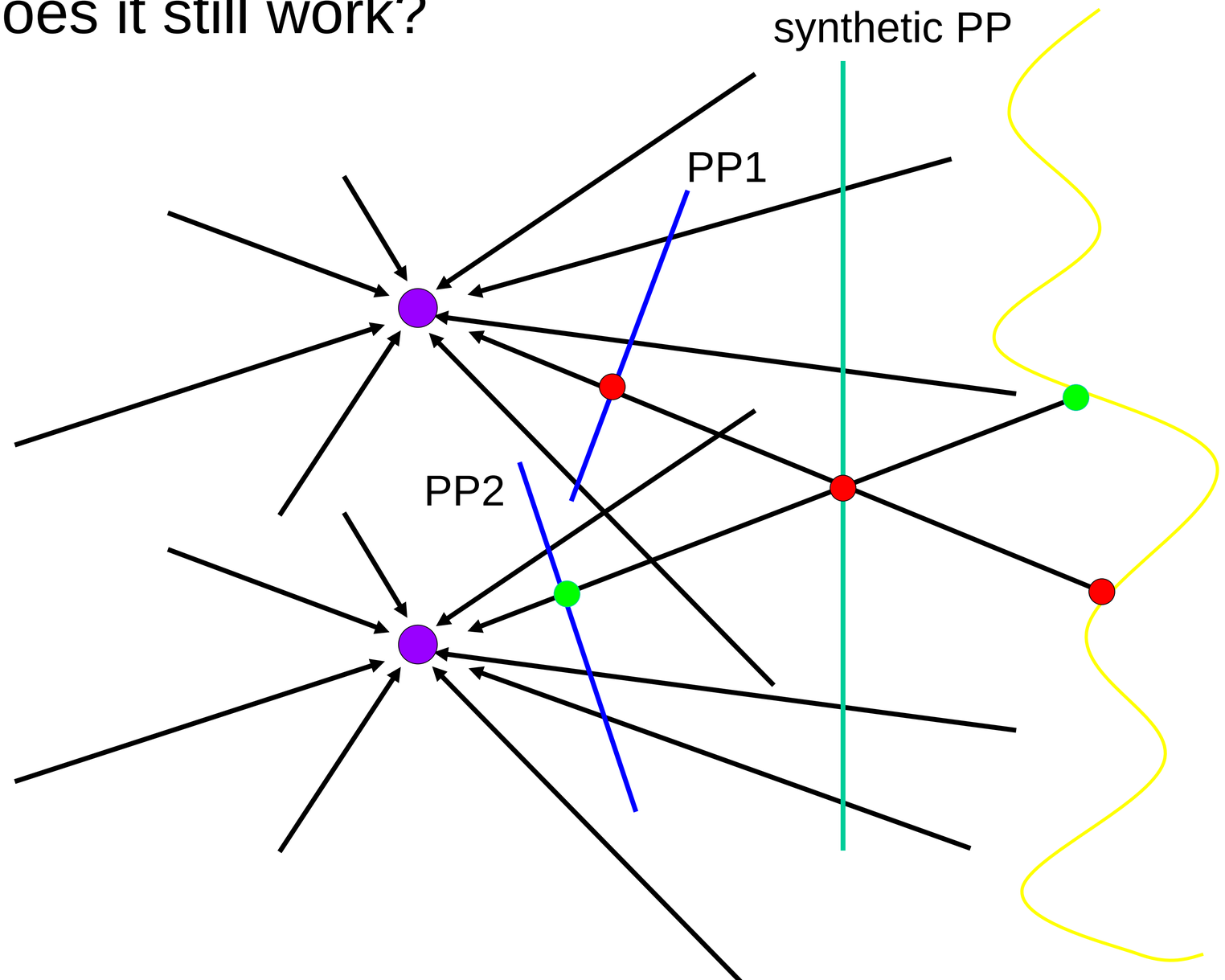


$(x_1,y_1) \longrightarrow (x_2,y_2)$

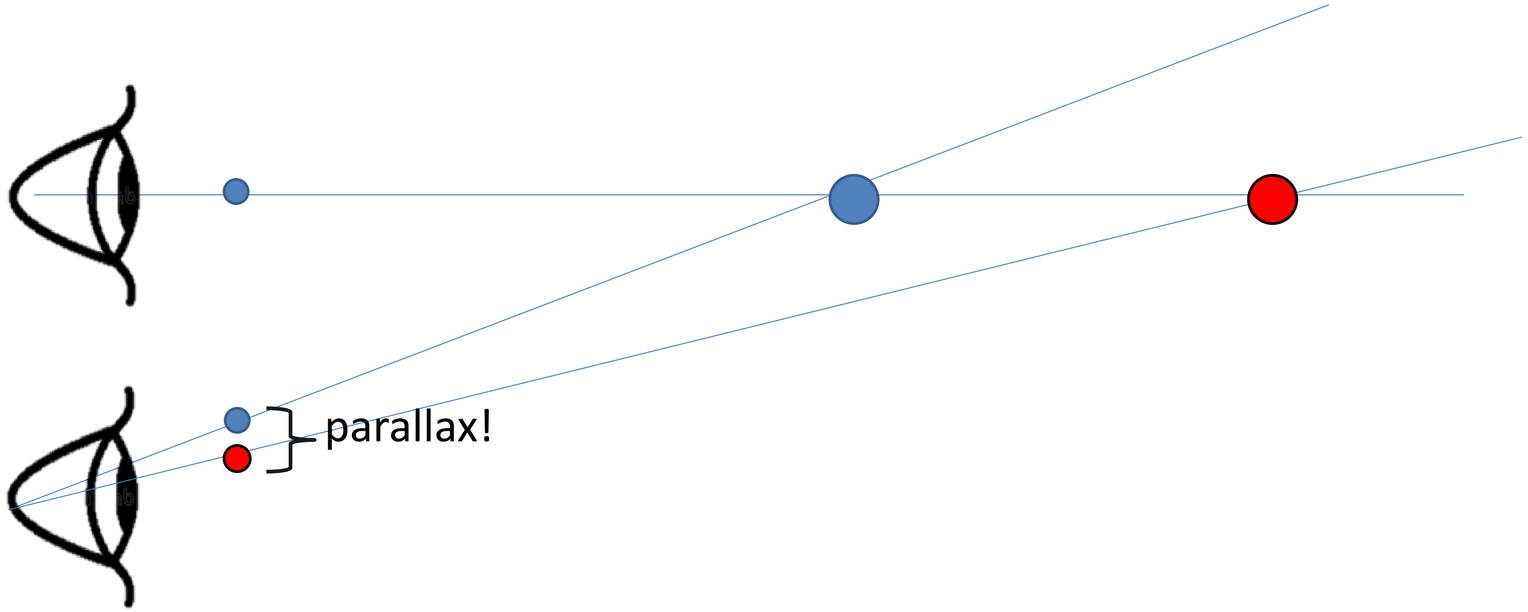
Homography

changing camera center

Does it still work?



Parallax



Parallax = *from ancient Greek parállaxis*
= *Para (*side by side) + *allássō*, (to alter)
= *Change in position from different view point*

Two eyes give you parallax, you can also move to see more parallax = “Motion Parallax”

Stereo vision

Two cameras, simultaneous views

Single moving camera and static scene

Non-parametric transformation

image I(x,y)

Disparity map D(x,y)

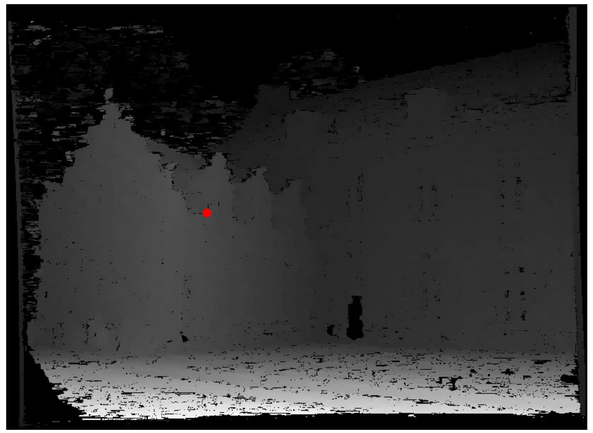
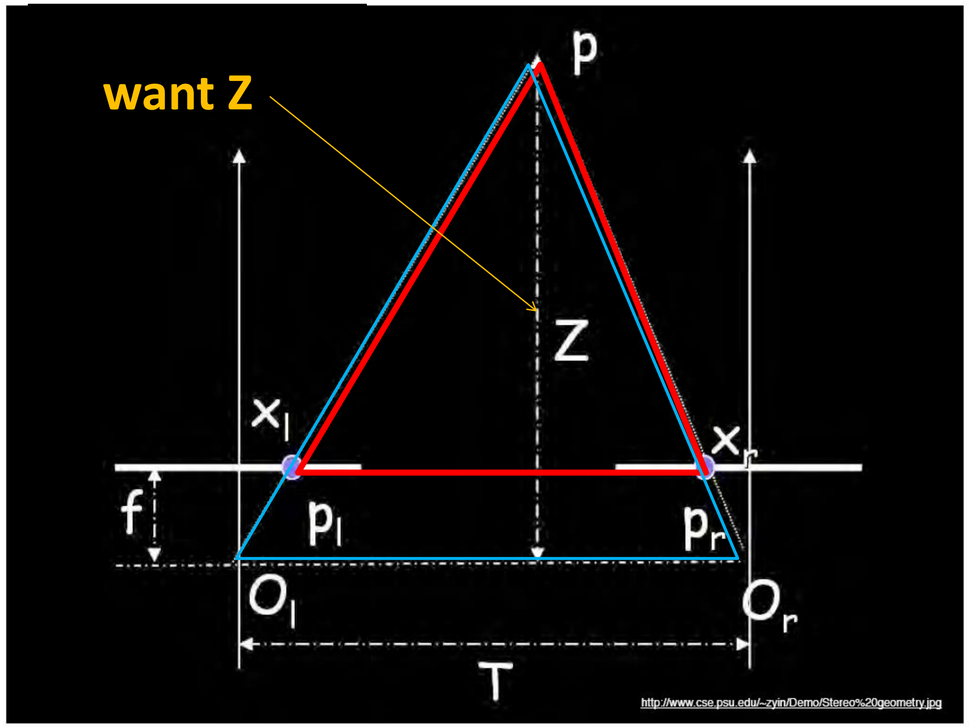


image I´(x´,y´)

$$(x´,y´)=(x+D(x,y),\ y)$$

Geometry for a simple stereo system

- Assume **parallel** optical axes, known camera parameters (i.e., calibrated cameras).



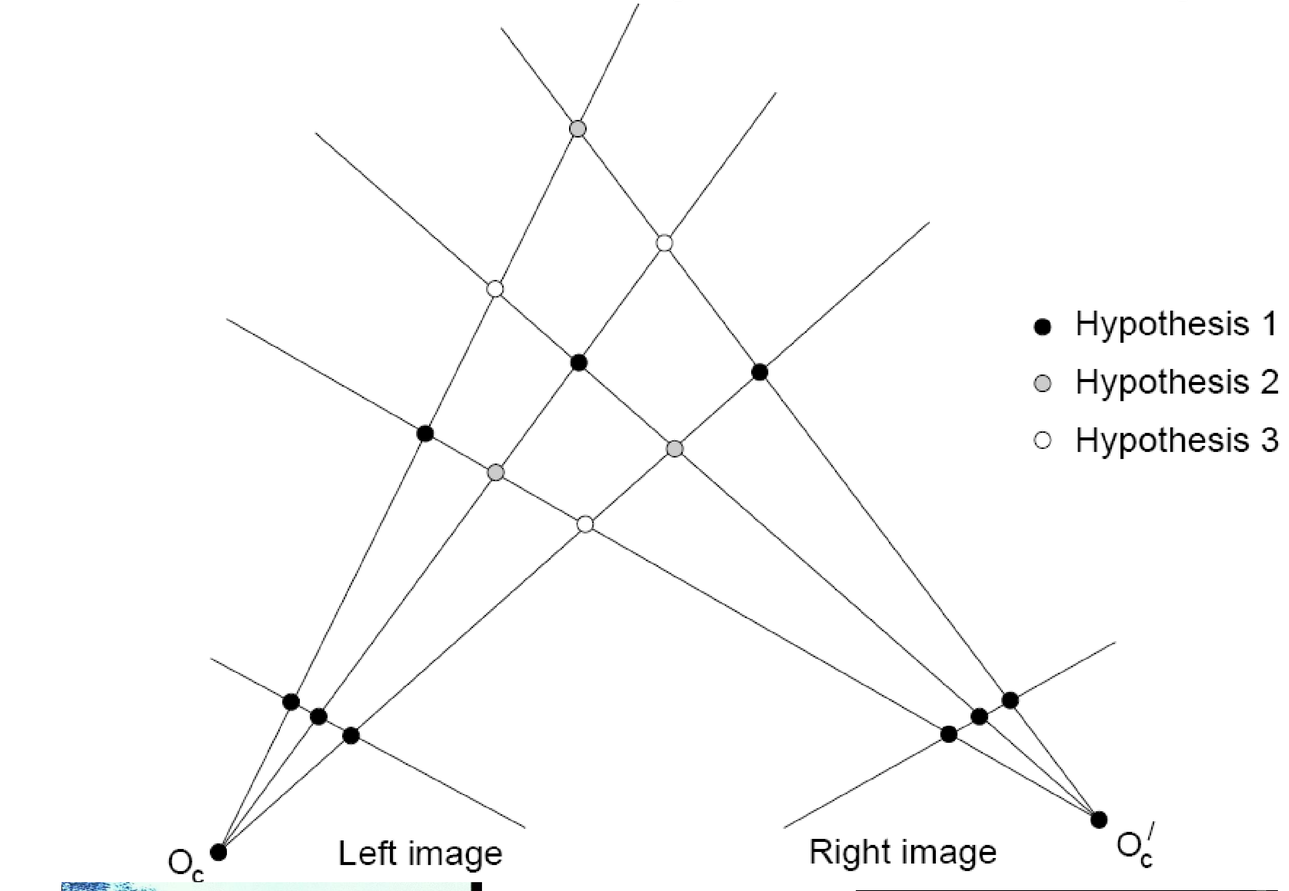
Use similar triangles (p_l, P, p_r) and (O_l, P, O_r):

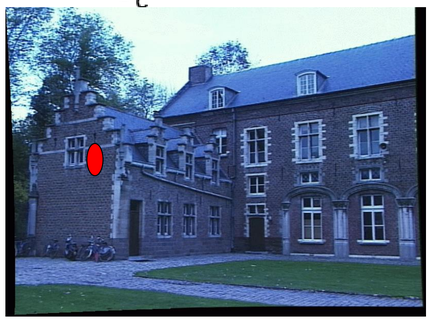
$$\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}$$

$$Z = f \frac{T}{x_r - x_l}$$

disparity

Correspondence problem





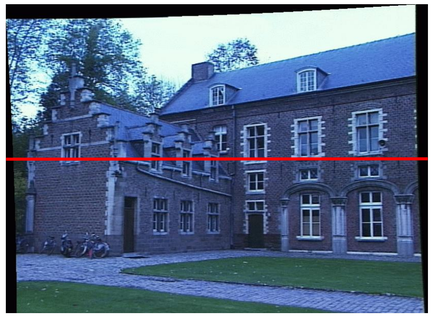
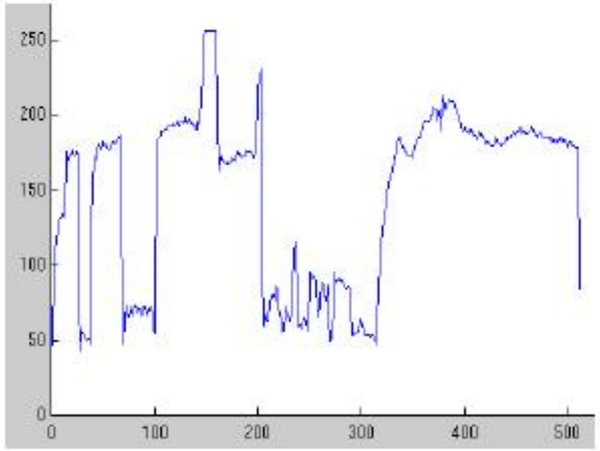
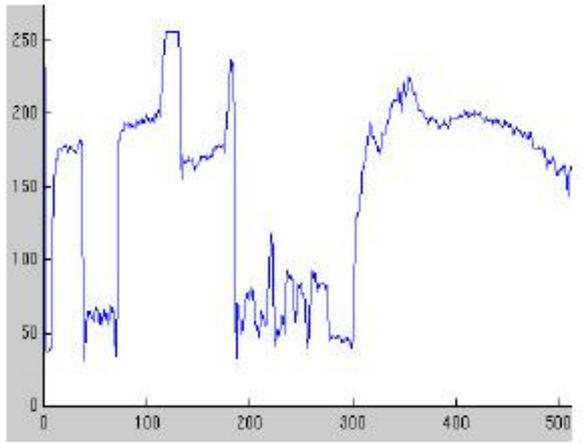


Figure from Gee & Cipolla 1999

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

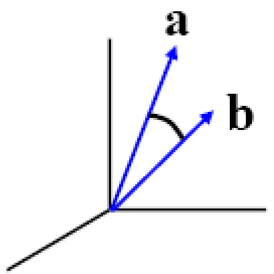
$$\text{NCC} = \frac{\sum_i \sum_j A(i,j)B(i,j)}{\sqrt{\sum_i \sum_j A(i,j)^2}\sqrt{\sum_i \sum_j B(i,j)^2}}$$

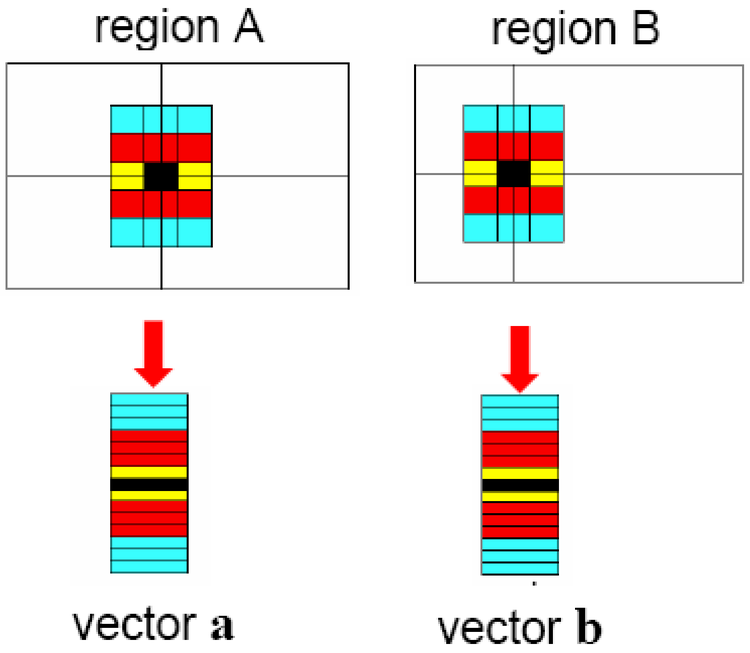
Write regions as vectors

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

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

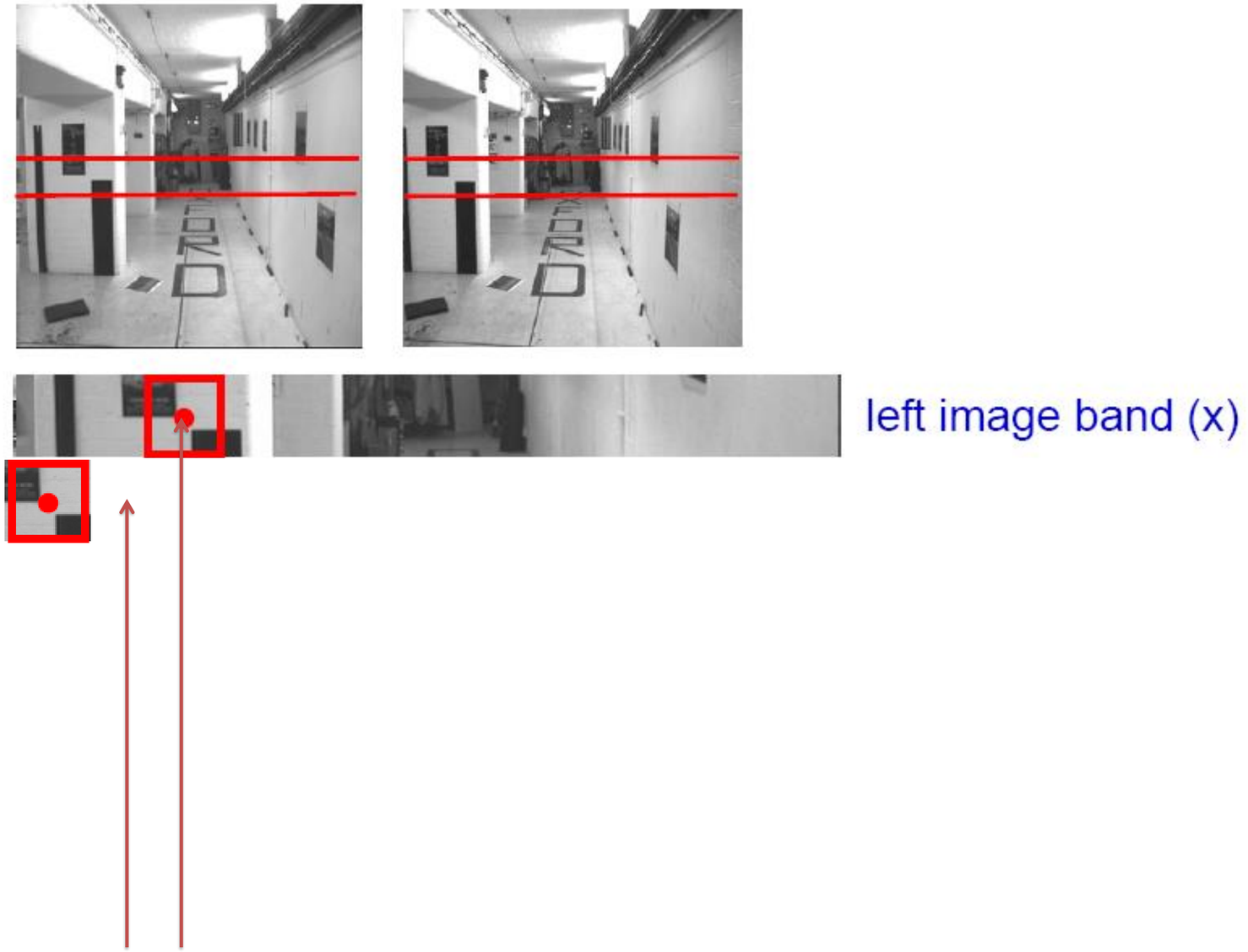
$$-1 \leq \text{NCC} \leq 1$$





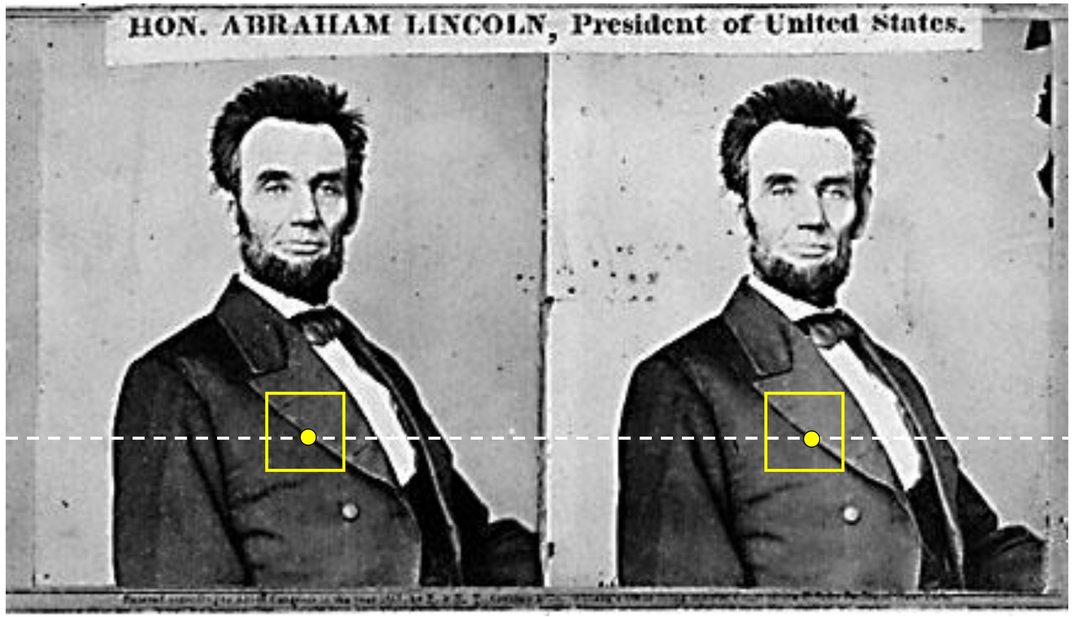
Source: Andrew Zisserman

Correlation-based window matching



Source: Andrew Zisserman

Dense correspondence search



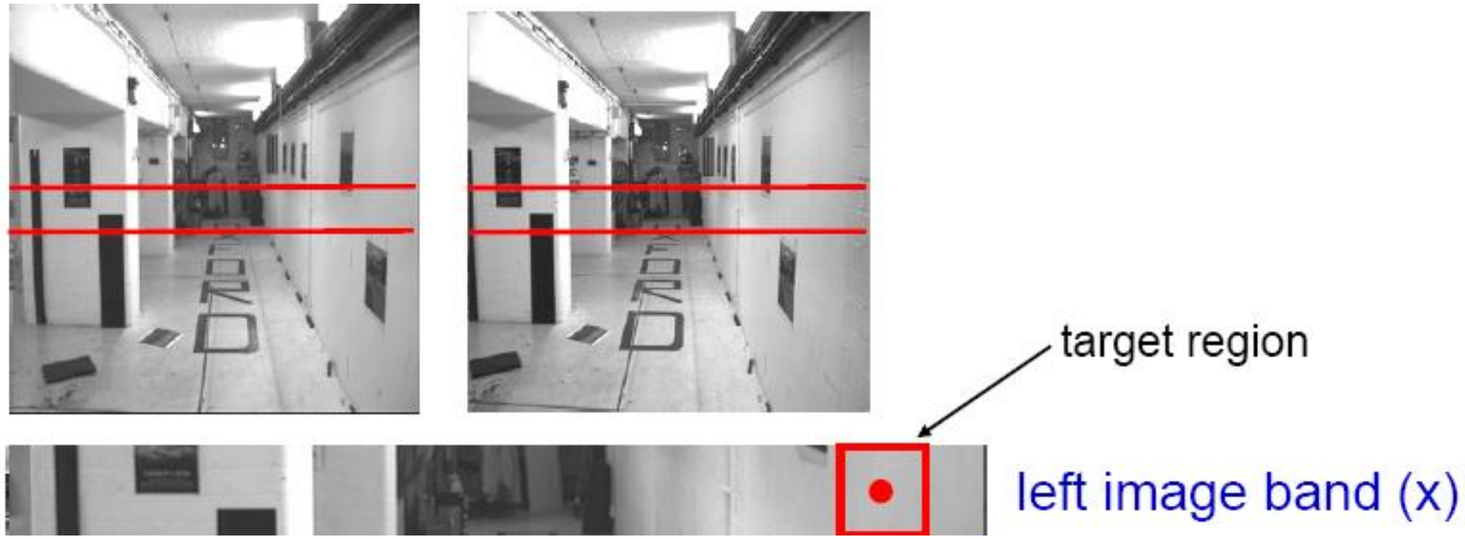
For each epipolar line

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

Failures of Correspondence Search

Repeated Patterns. Why?

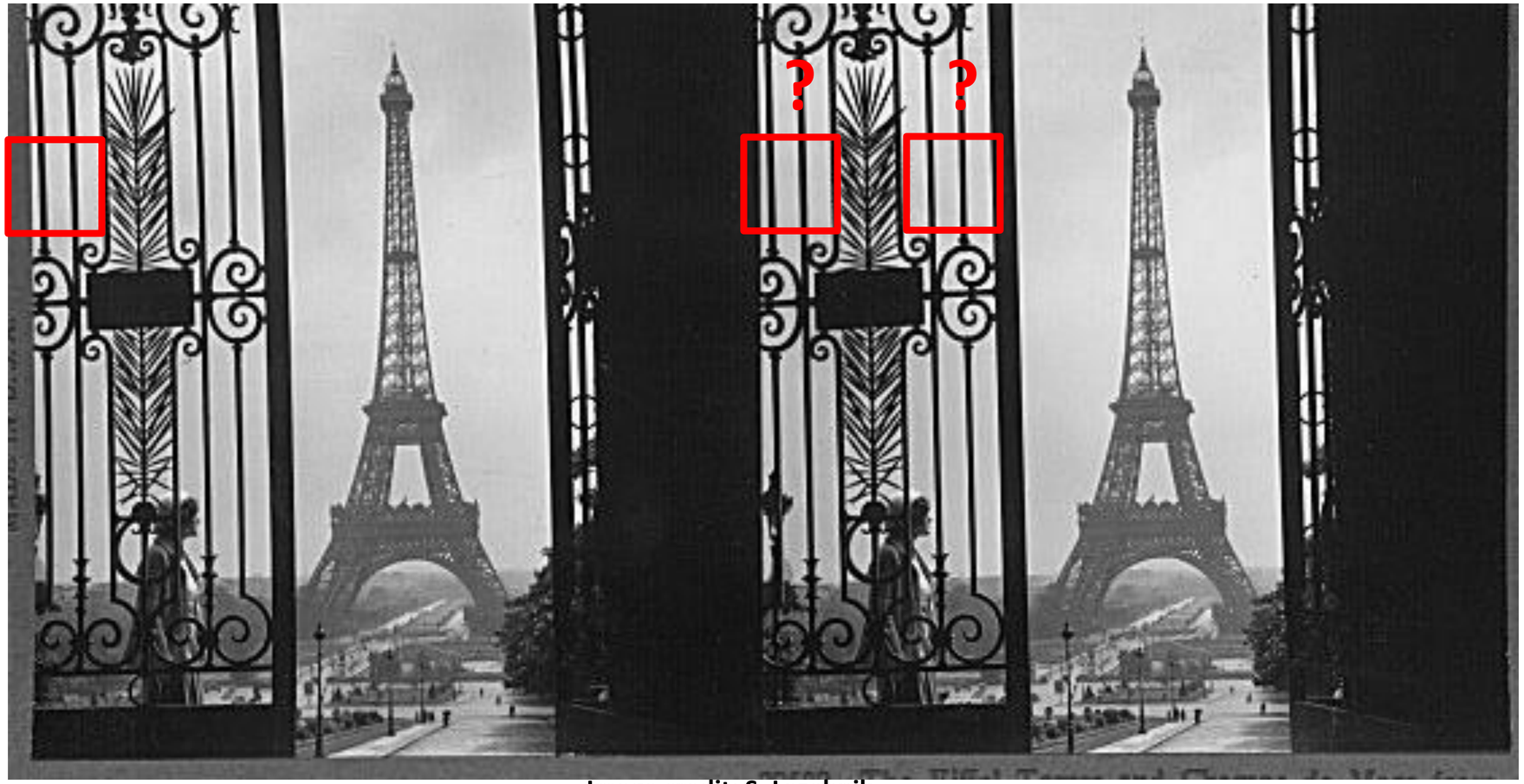
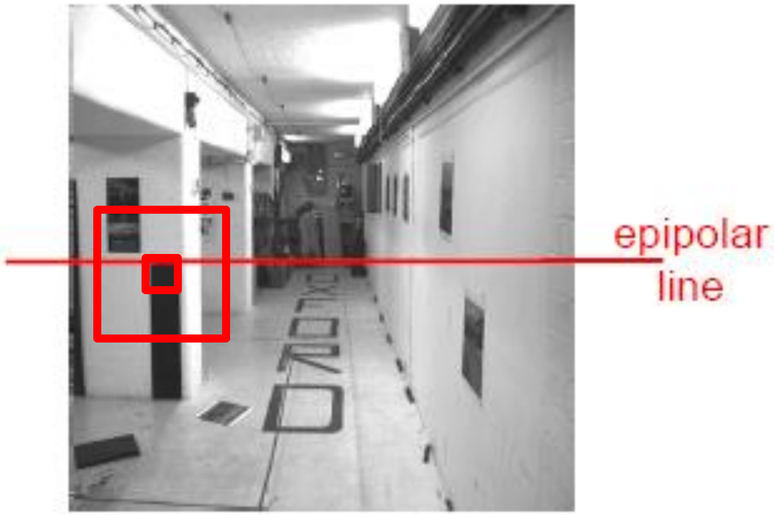


Image credit: S. Lazebnik

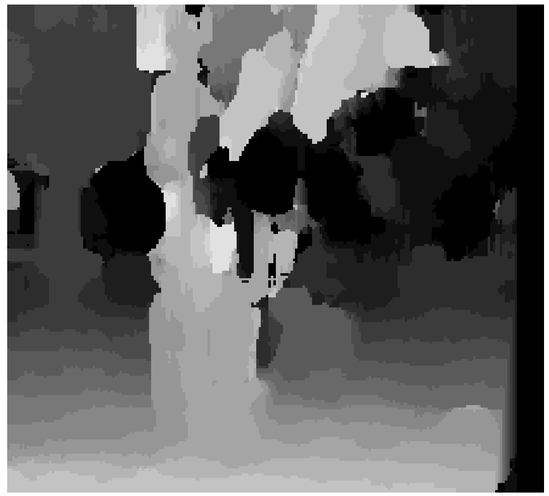
Effect of window size



Source: Andrew Zisserman

Effect of window size

W = 3



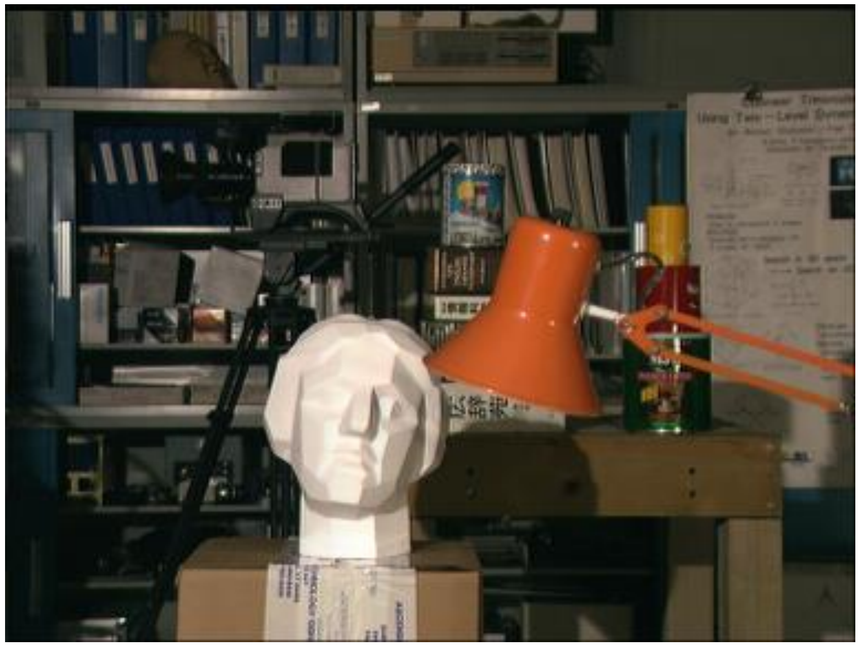
W = 20

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

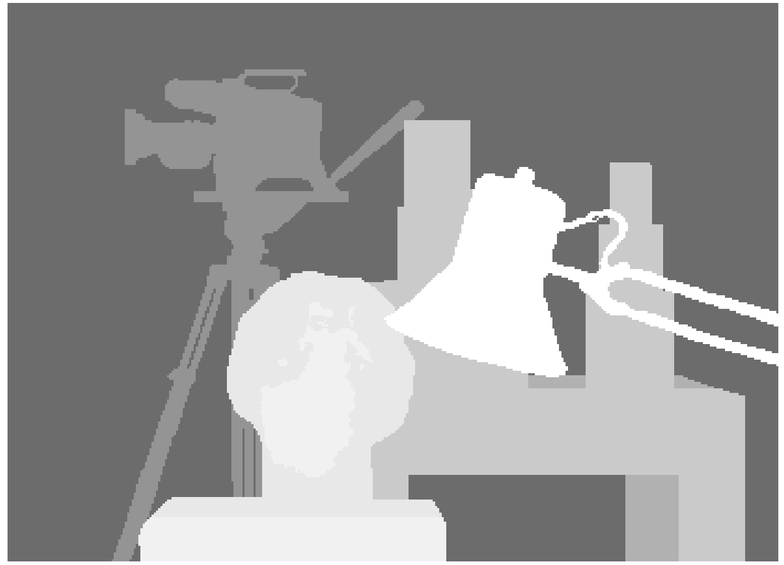
Figures from Li Zhang

Stereo Results

– Data from University of Tsukuba

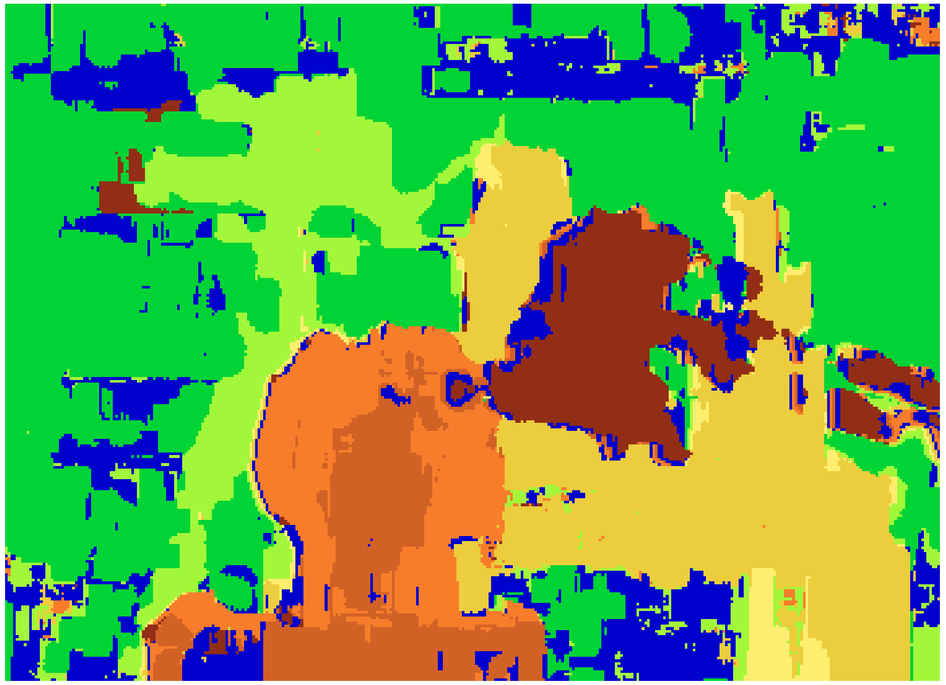


Scene



Ground truth

Results with Window Search



Window-based matching
(best window size)

Ground truth

Better methods exist...

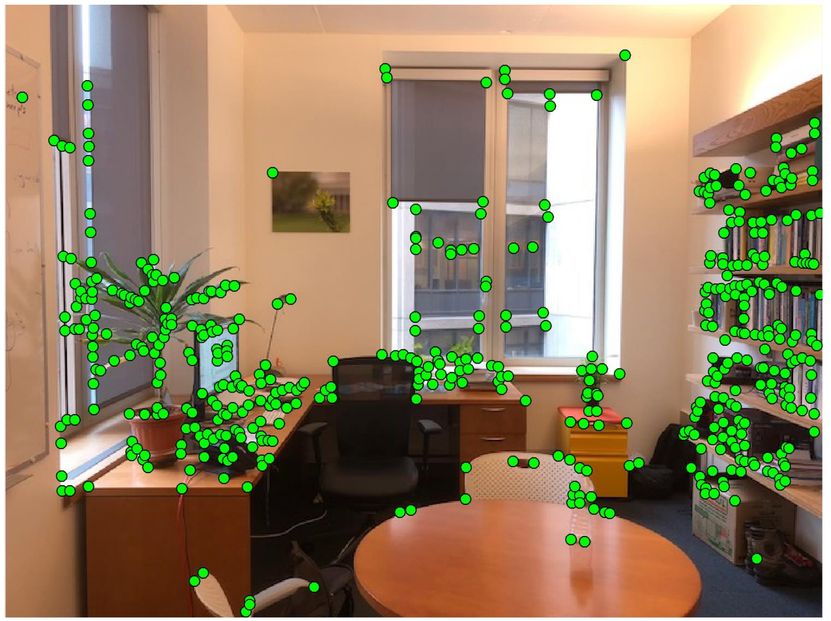
Energy Minimization

Ground truth

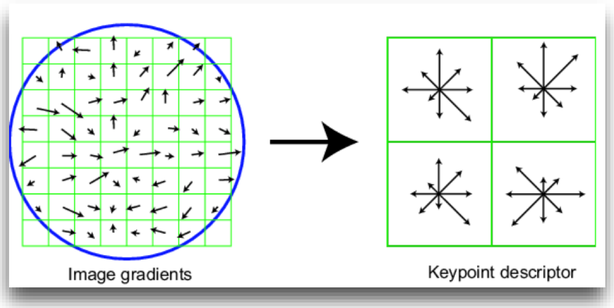
Boykov et al., Fast Approximate Energy Minimization via Graph Cuts, International Conference on Computer Vision, September 1999.

Feature correspondences

1) detect keypoints

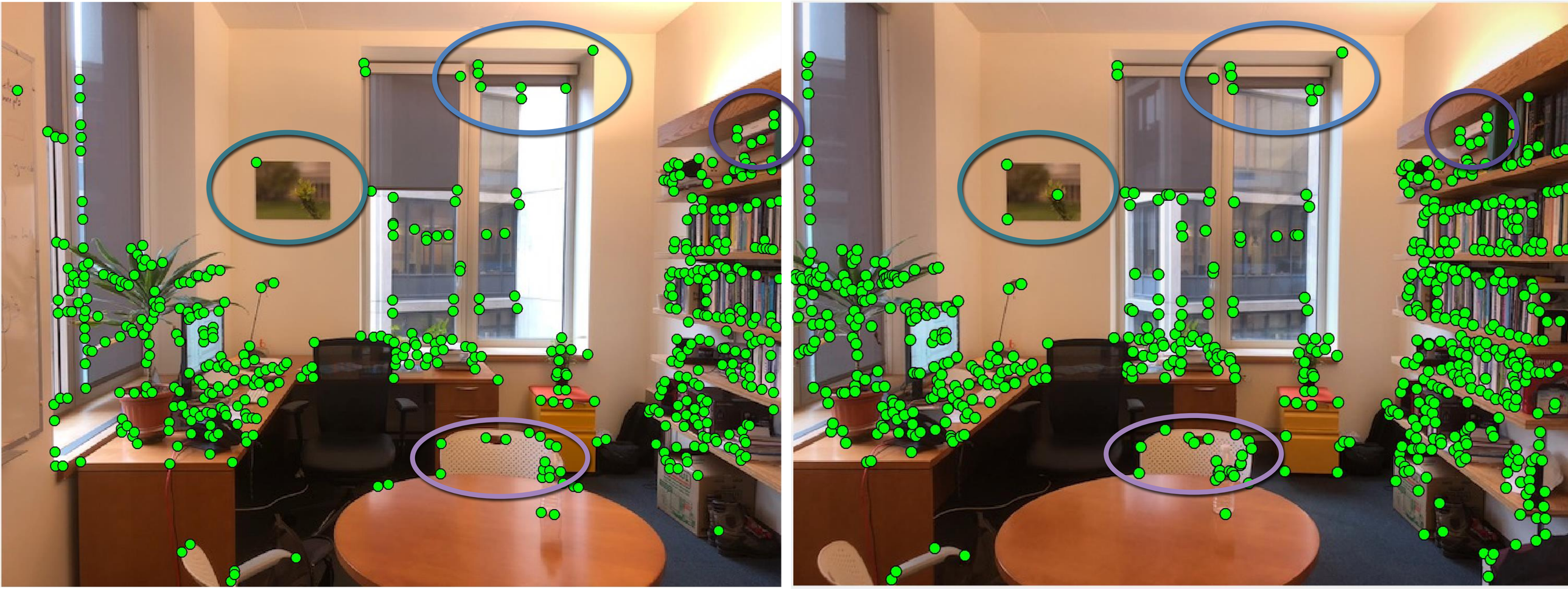


2) extract SIFT at each keypoint



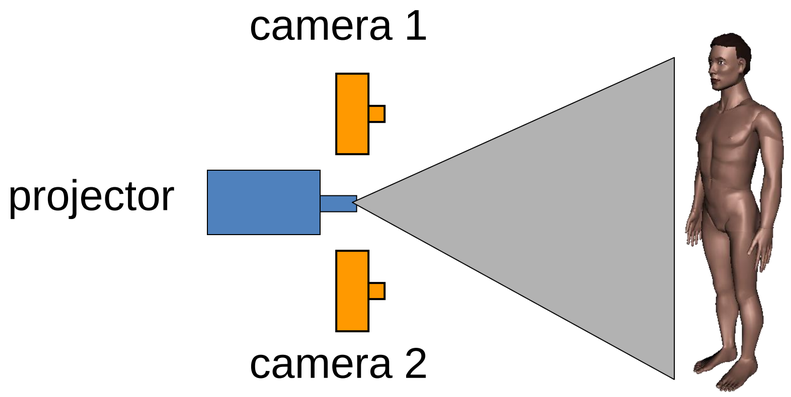
points = detectHarrisFeatures(img);

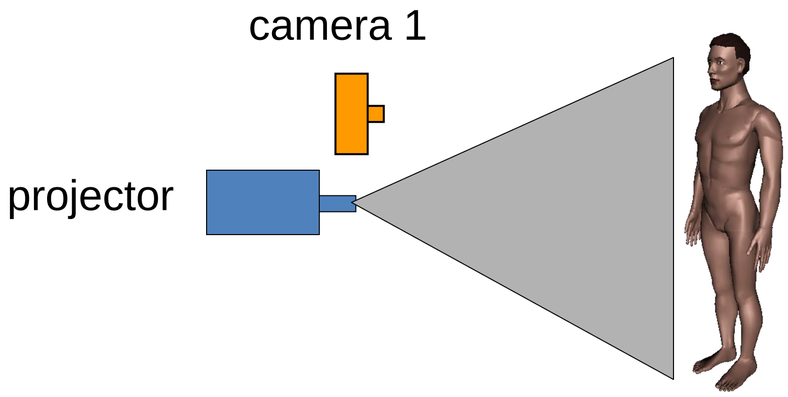
Finding correspondences (SIFT)



Active stereo with structured light

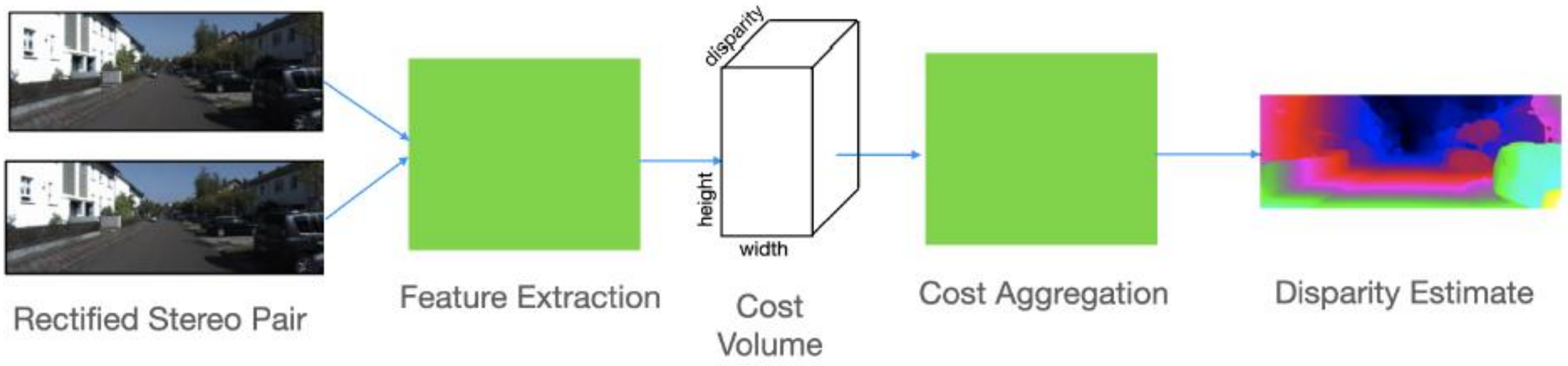
Li Zhang's one-shot stereo



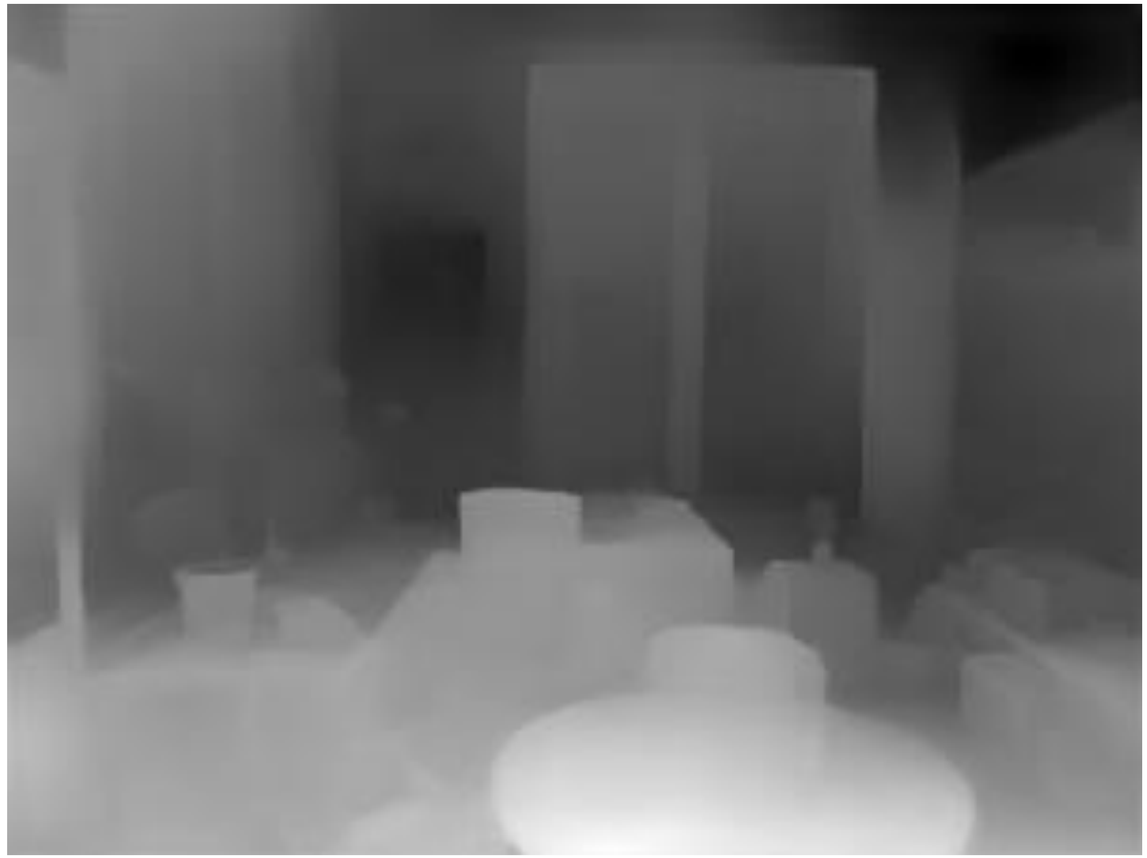


- Project "structured" light patterns onto the object
  - simplifies the correspondence problem

CNN-based Stereo Matching



Can also learn depth from a single image



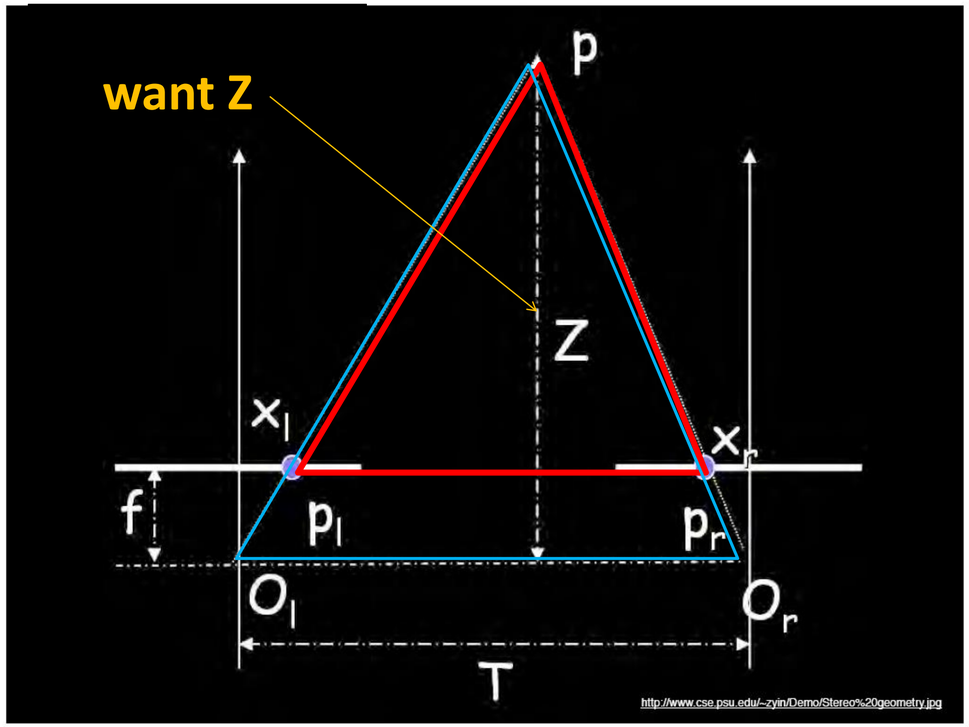
MegaDepth: Learning Single-View Depth Prediction from Internet Photos

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

Source: Torralba, Isola, Freeman

Geometry for a simple stereo system

- Assume **parallel** optical axes**, known camera parameters (i.e., calibrated cameras).**



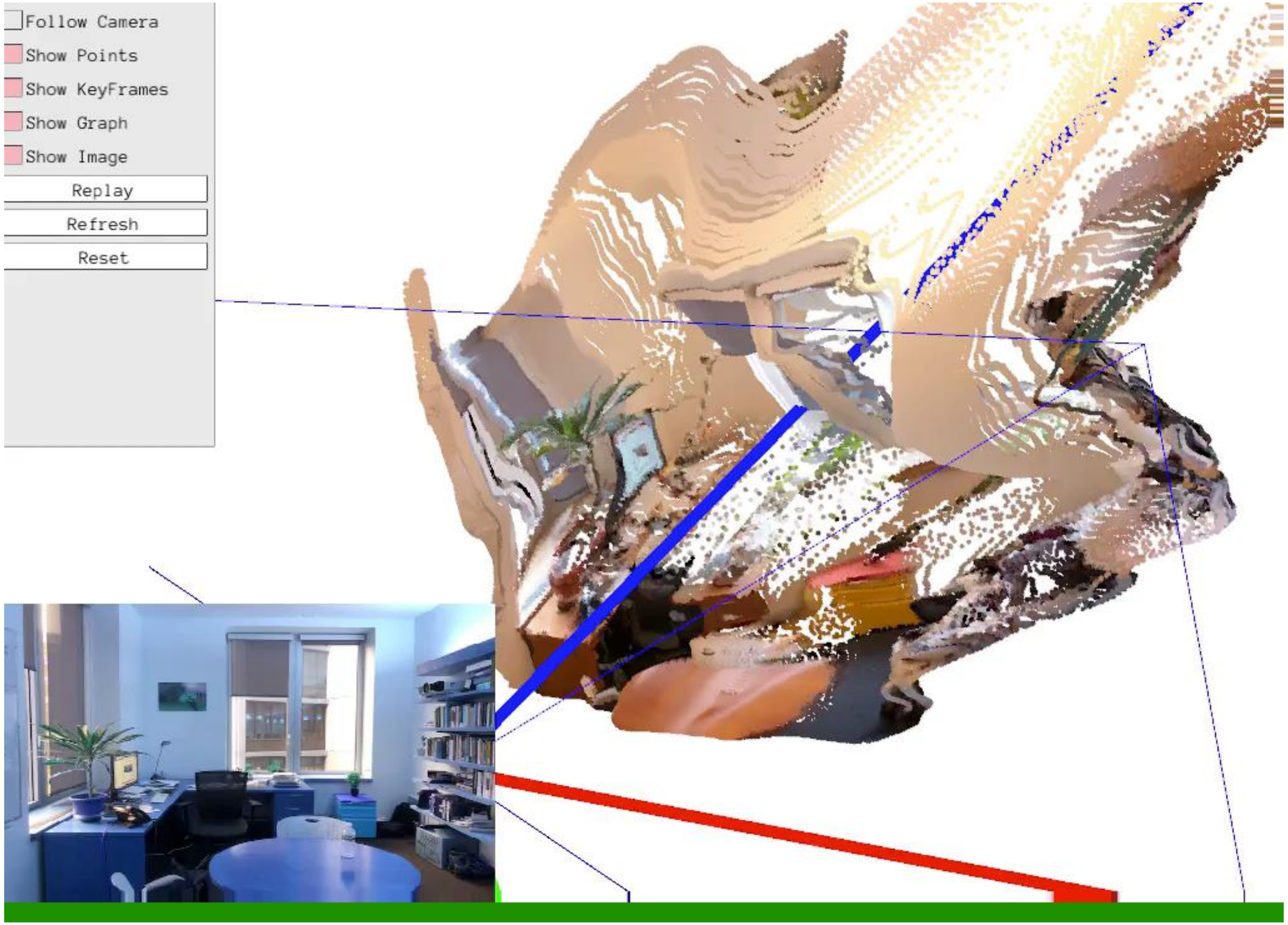
Use similar triangles (p_l, P, p_r) and (O_l, P, O_r):

$$\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}$$

$$Z = f \frac{T}{x_r - x_l}$$

disparity

Manually adjusting the focal length, to something that looks reasonable...



Camera Projection Model

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

5 6

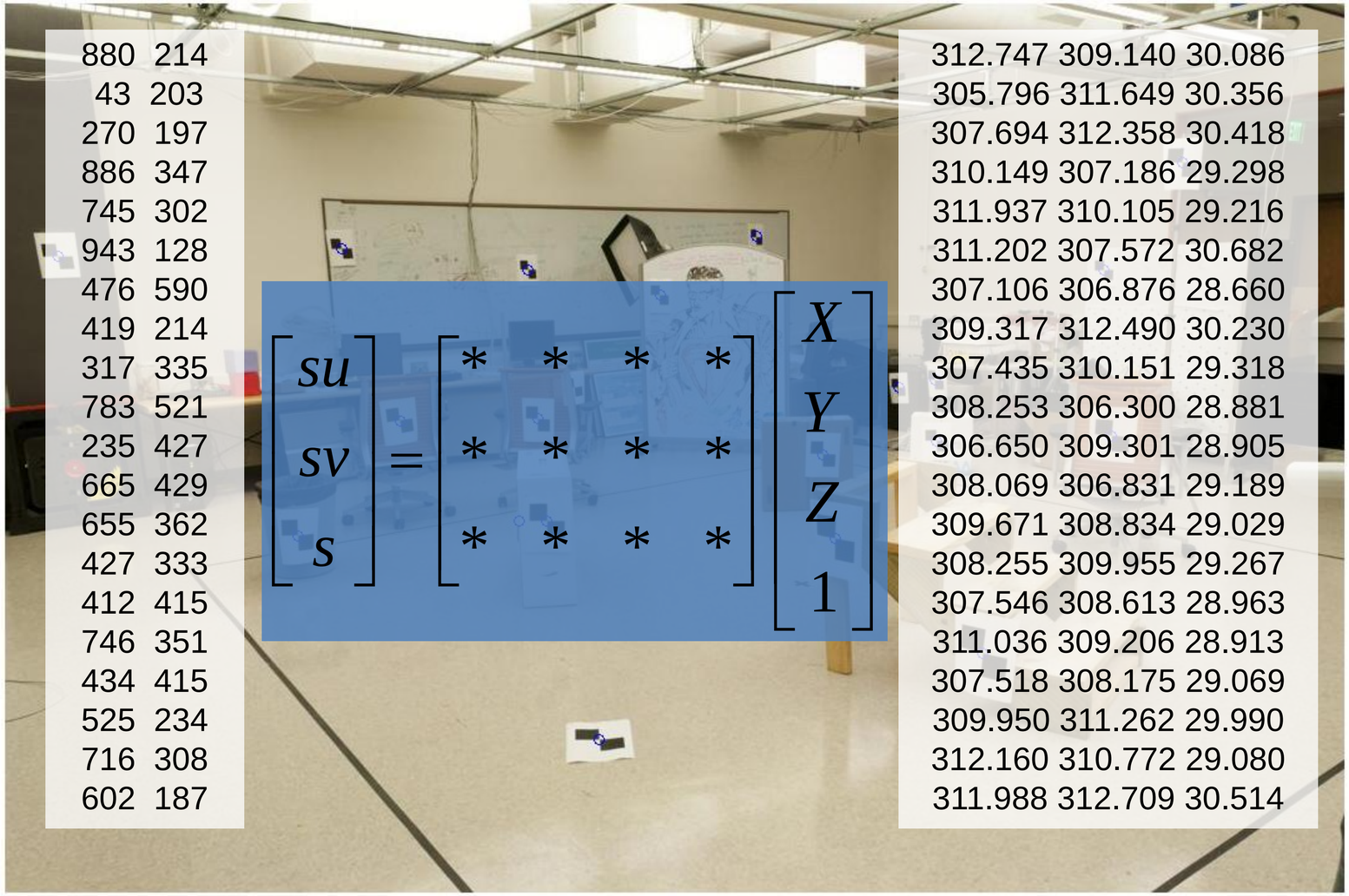
$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

How to calibrate the camera?

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

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

How do we calibrate a camera? Learning problem!



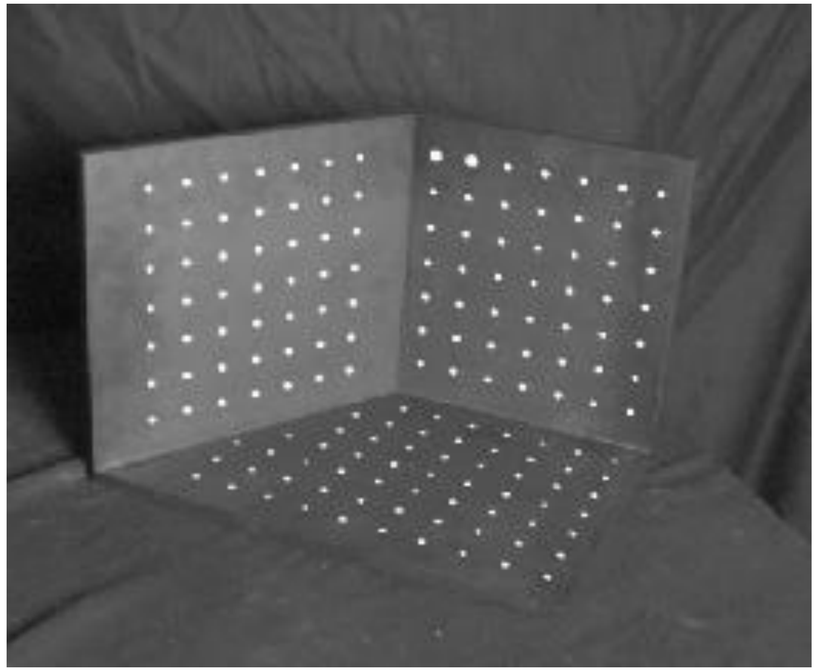
880	214
43	203
270	197
886	347
745	302
943	128
476	590
419	214
317	335
783	521
235	427
665	429
655	362
427	333
412	415
746	351
434	415
525	234
716	308
602	187

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

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

Calibration using a reference object

- Place a known object in the scene
  - identify correspondence between image and scene
  - compute mapping from scene to image



Issues

- **must know geometry very accurately**
- **must know 3D -> 2D correspondence**

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

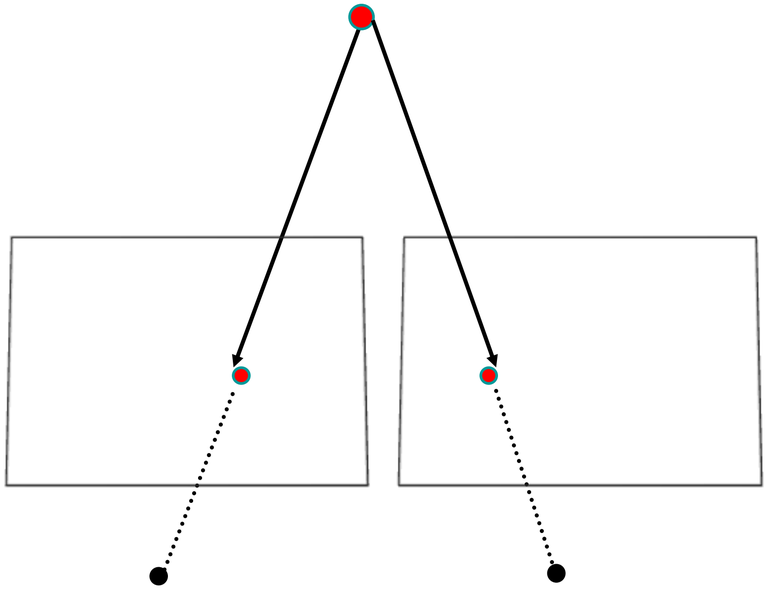
$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 & -u_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 & -v_1 \\ & & & & & & \vdots & & & & & \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -u_nX_n & -u_nY_n & -u_nZ_n & -u_n \\ 0 & 0 & 0 & 0 & X_n & Y_n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

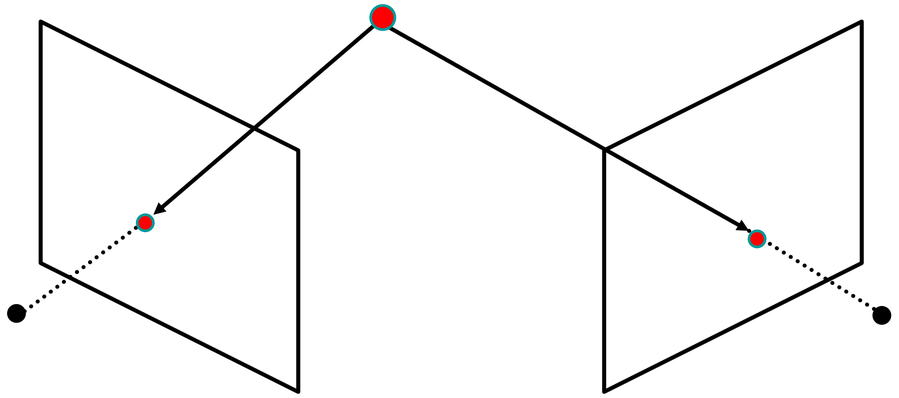
Can we factorize M back to K [R | T]?

- Yes!
- You can use *RQ* factorization (note – not the more familiar *QR* factorization). *R* (right diagonal) is K, and *Q* (orthogonal basis) is R. T, the last column of [R | T], is inv(K) * last column of M.
  - But you need to do a bit of post-processing to make sure that the matrices are valid. See http://ksimek.github.io/2012/08/14/decompose/

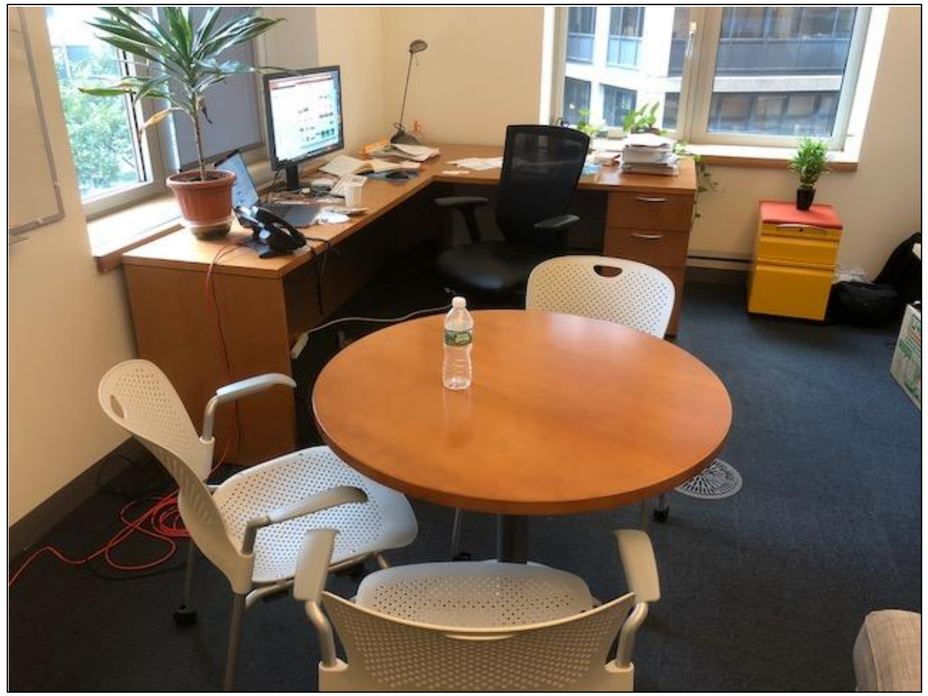
General case

- The two cameras need not have parallel optical axes.

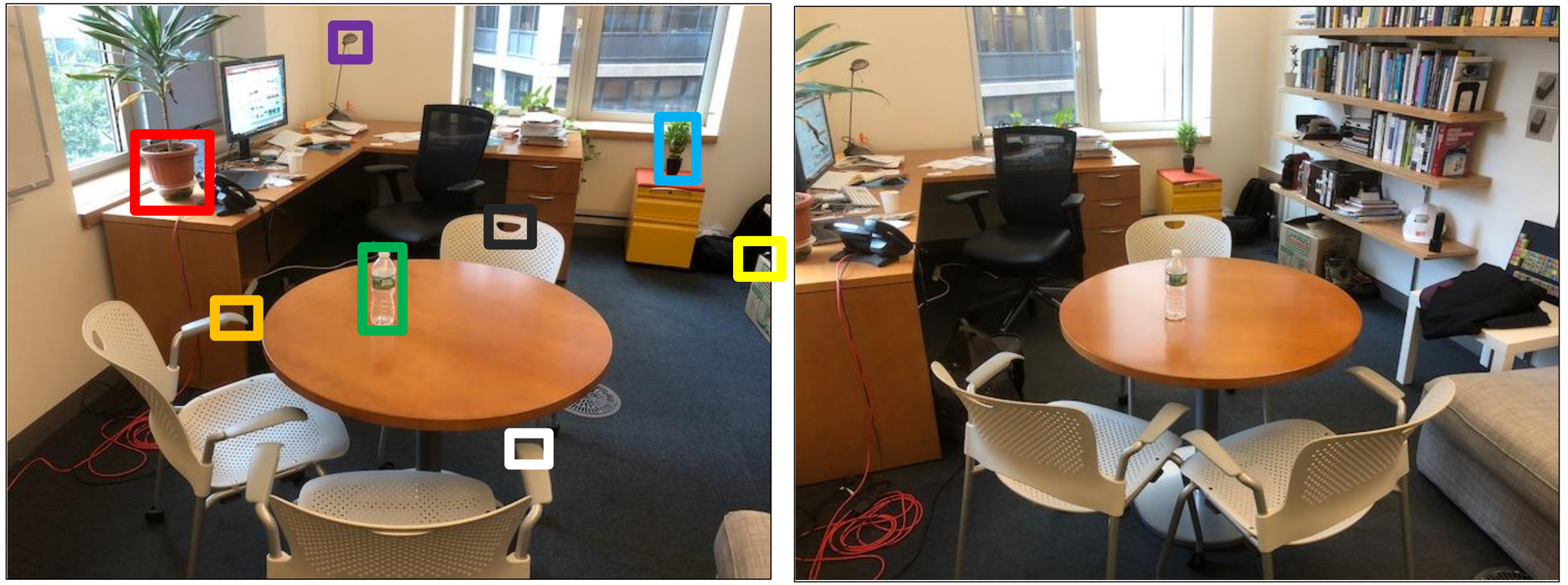




Source: Torralba, Isola, Freeman

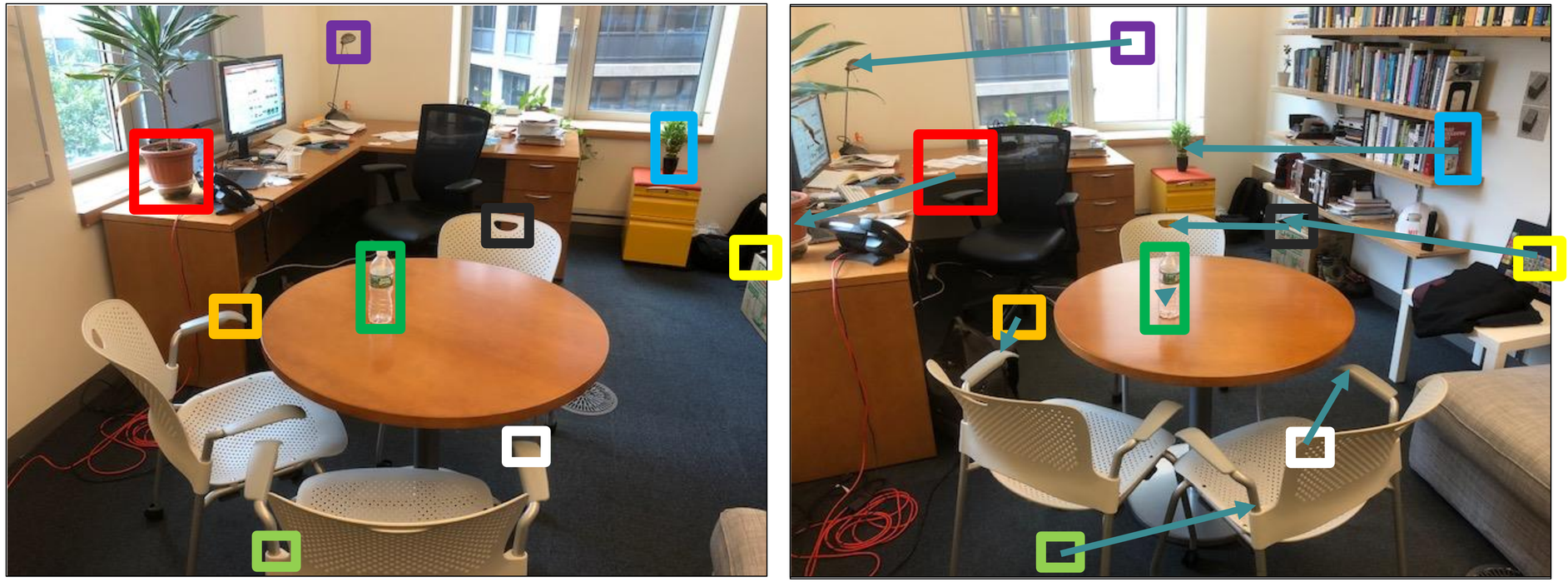


Source: Torralba, Isola, Freeman



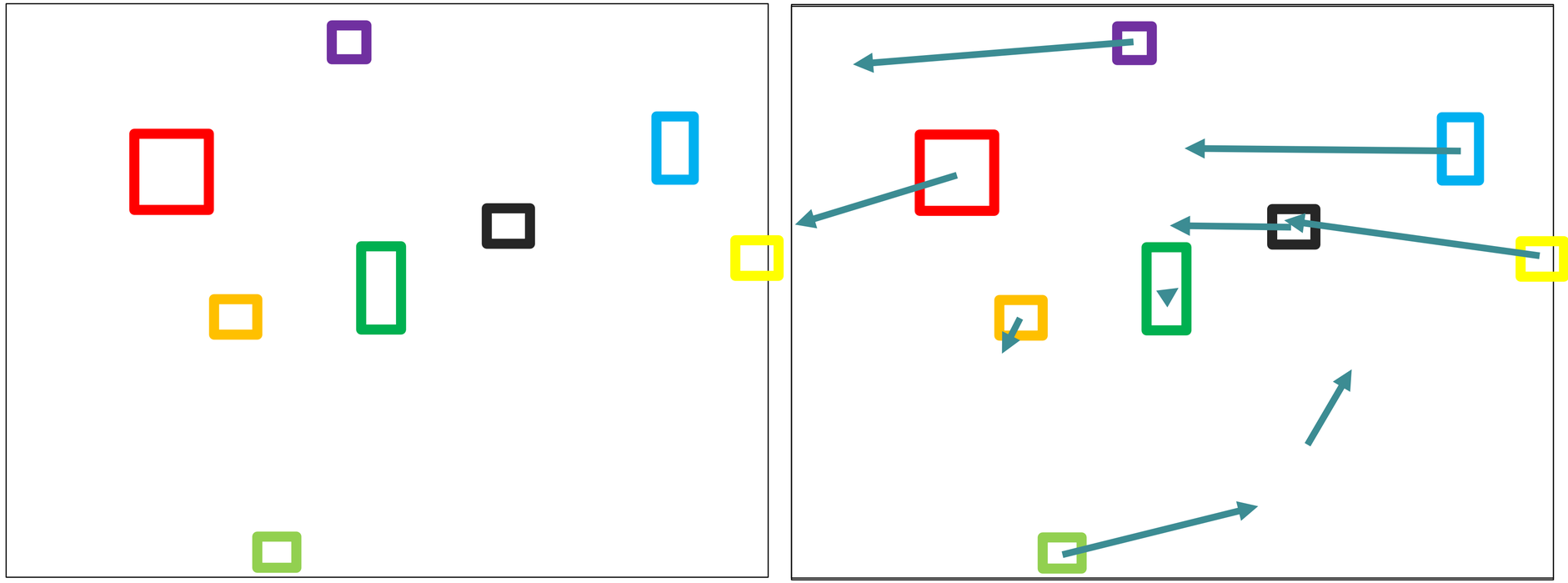
Do we need to search for matches only along horizontal lines?

Source: Torralba, Isola, Freeman



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

Source: Torralba, Isola, Freeman

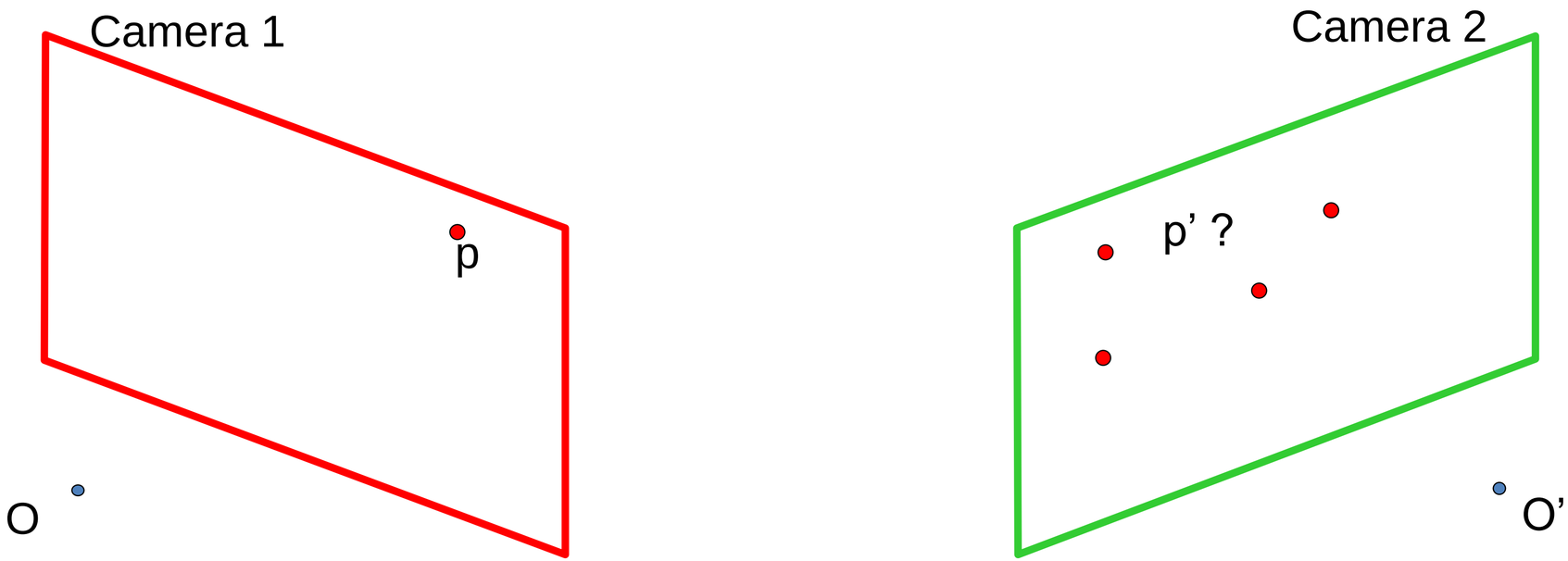


~~Do we need to search for matches only along horizontal lines?~~

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

Source: Torralba, Isola, Freeman

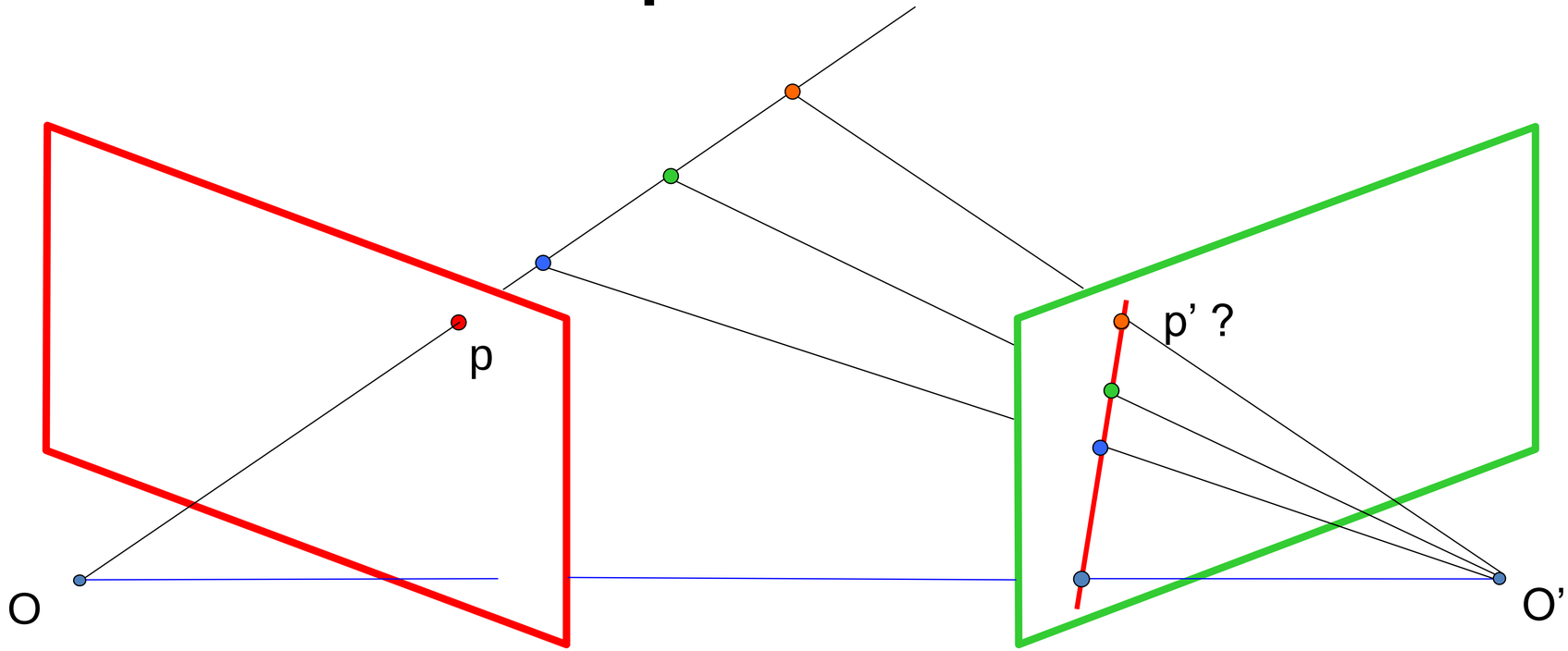
Stereo correspondence constraints



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

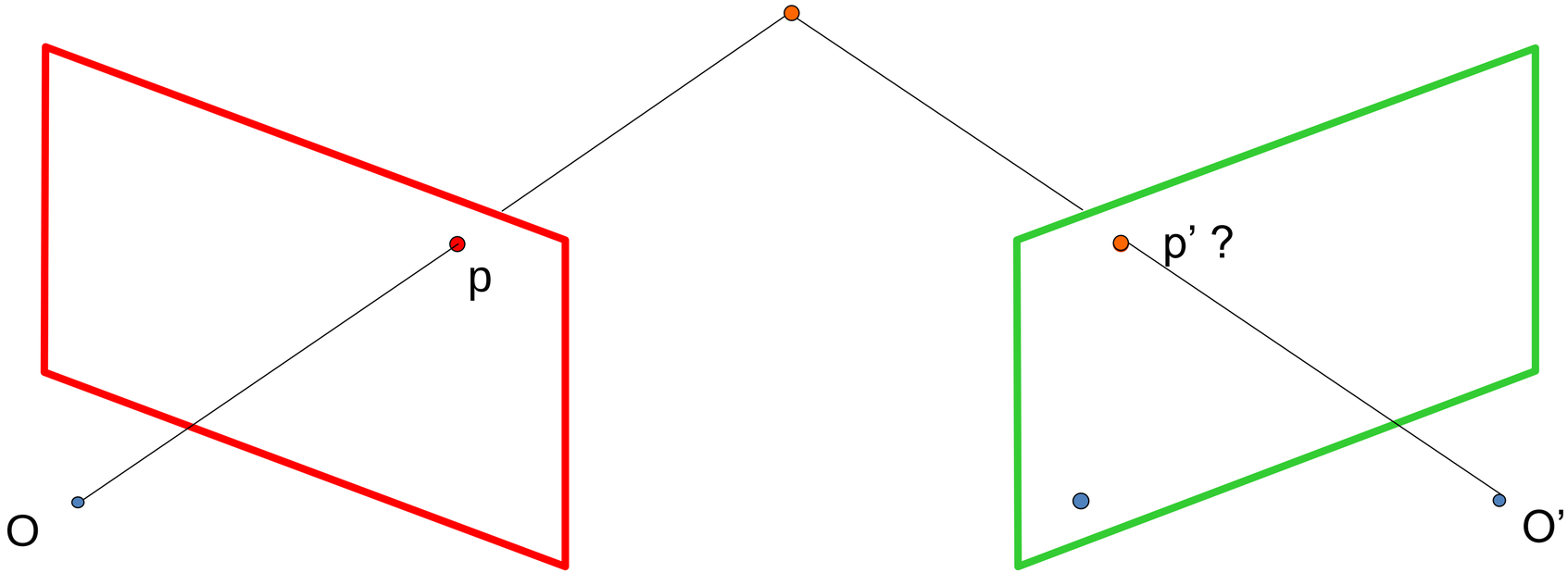
Source: Torralba, Isola, Freeman

Stereo correspondence constraints



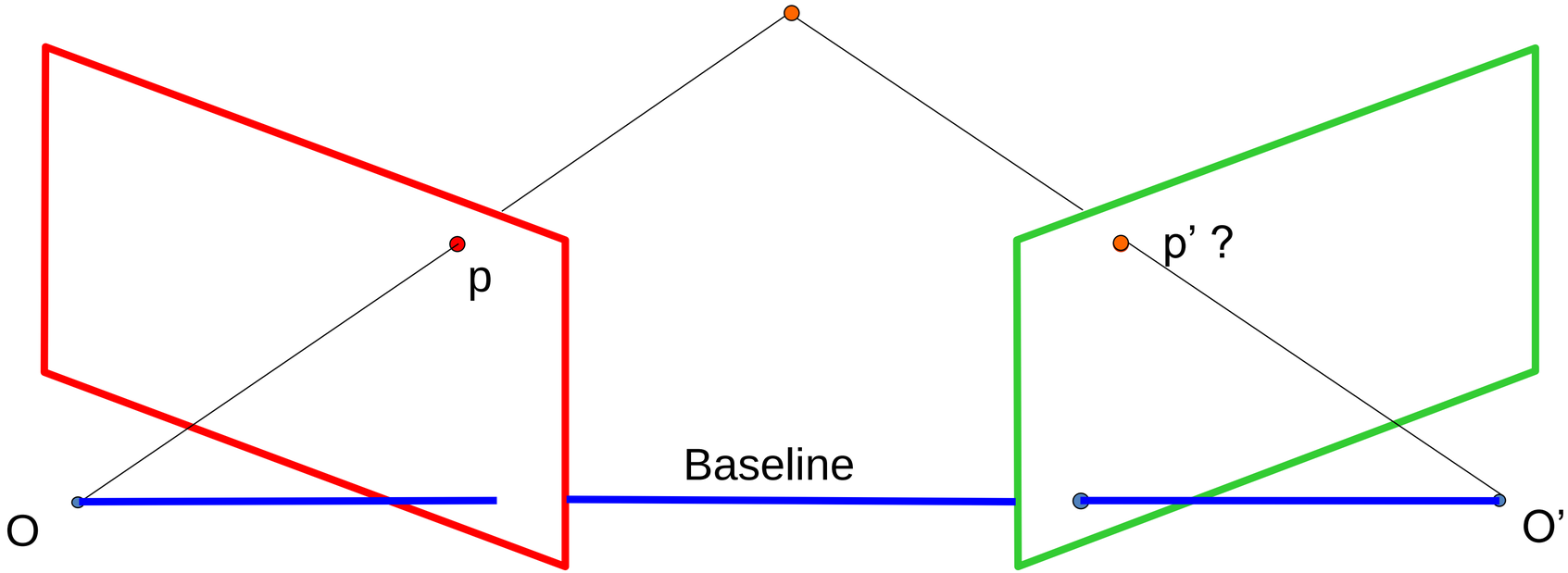
Source: Torralba, Isola, Freeman

Some terminology



Source: Torralba, Isola, Freeman

Some terminology

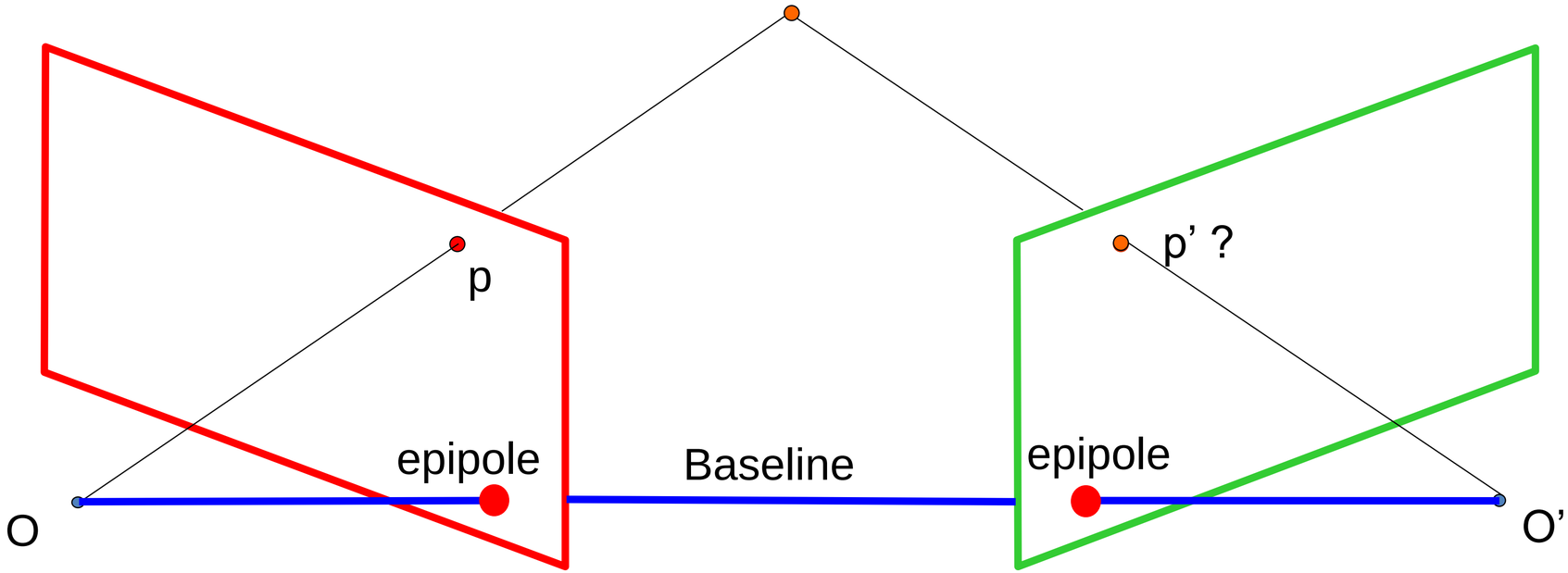


Baseline: the line connecting the two camera centers

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

Source: Torralba, Isola, Freeman

Some terminology

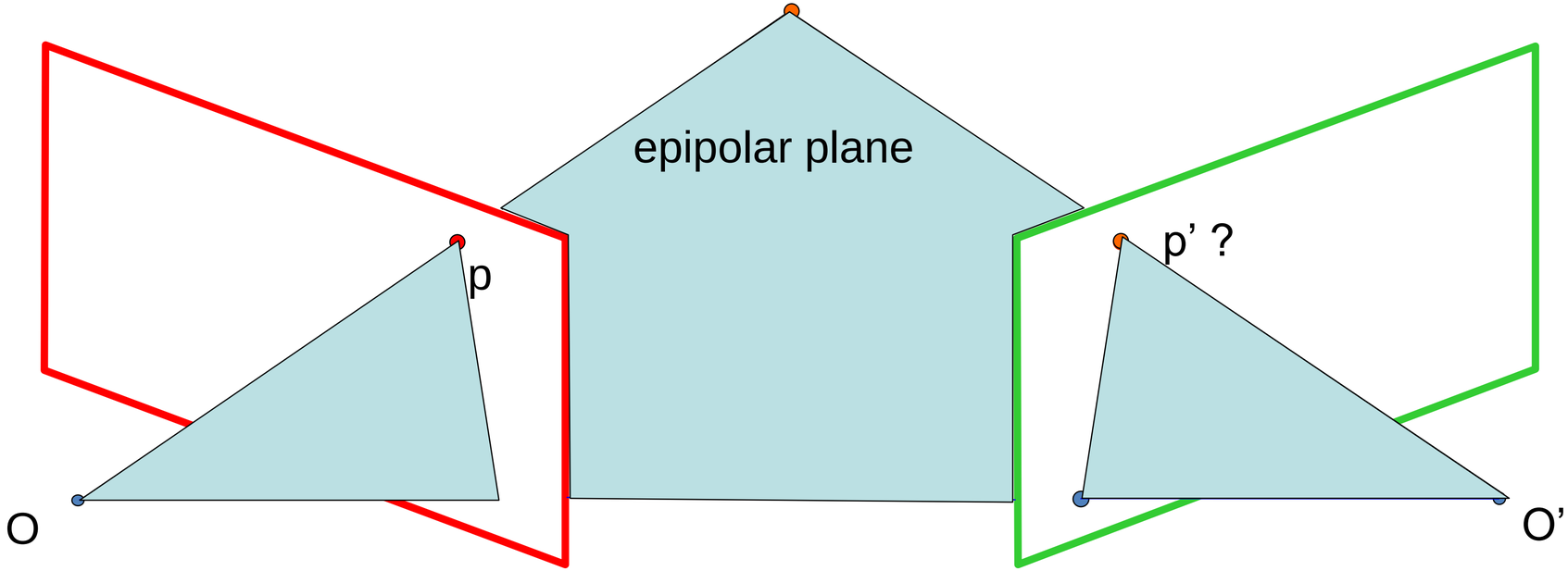


Baseline: the line connecting the two camera centers

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

Source: Torralba, Isola, Freeman

Some terminology



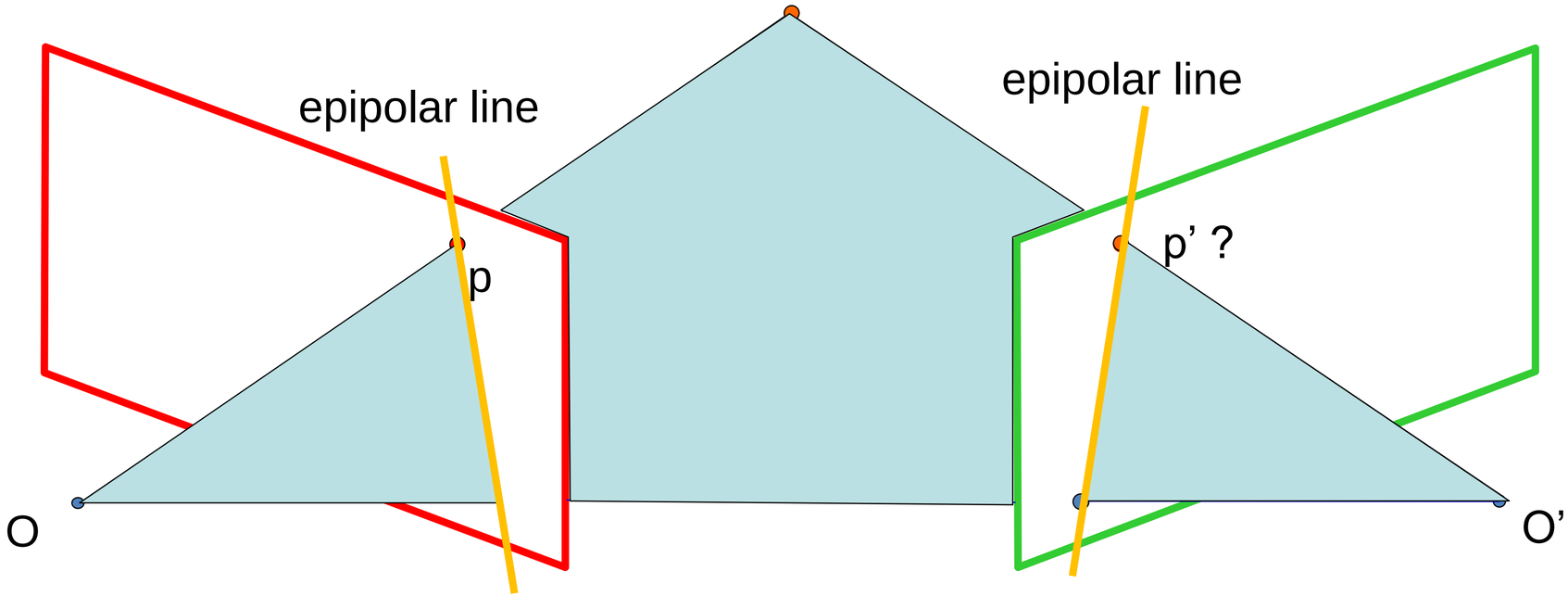
Baseline: the line connecting the two camera centers

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

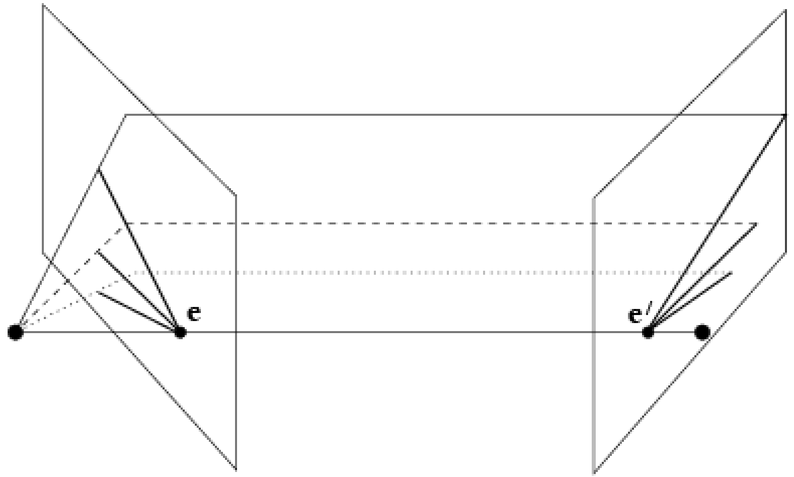
Epipole: point of intersection of *baseline* with the image plane

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

Example: converging cameras



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

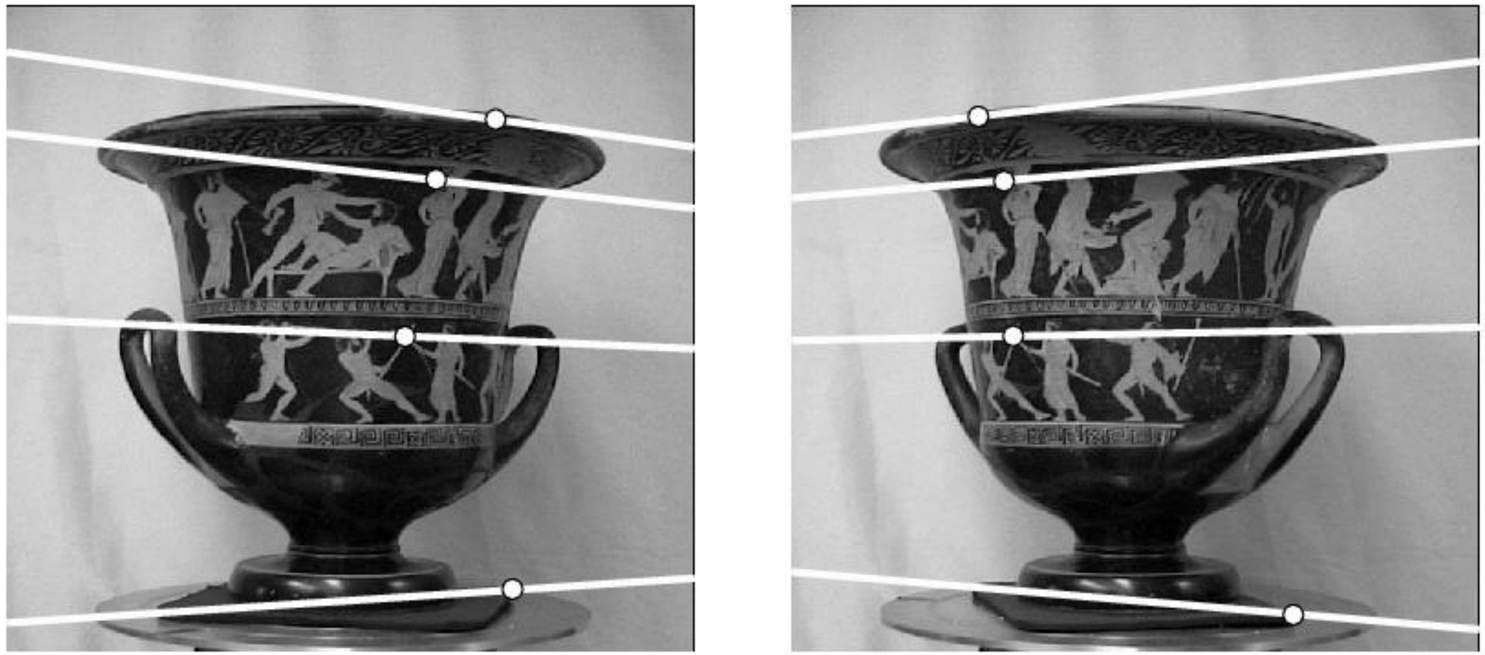
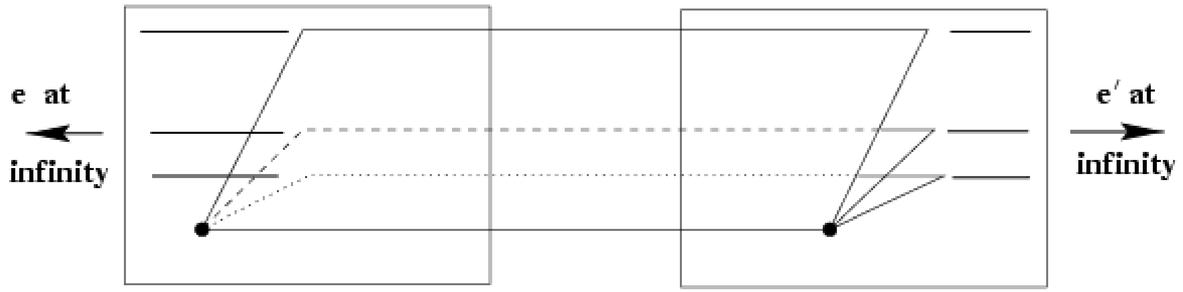


Figure from Hartley & Zisserman

Example: motion parallel with image plane



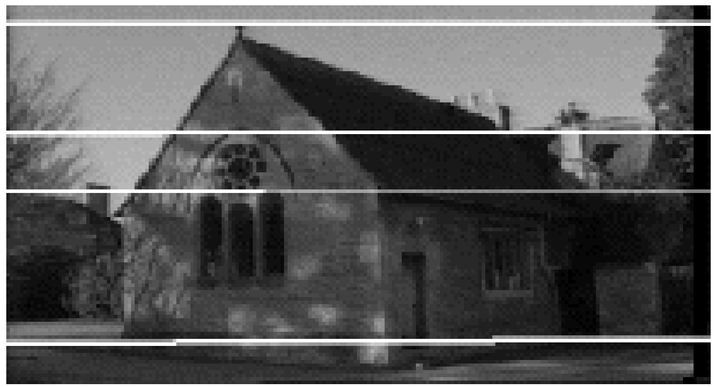
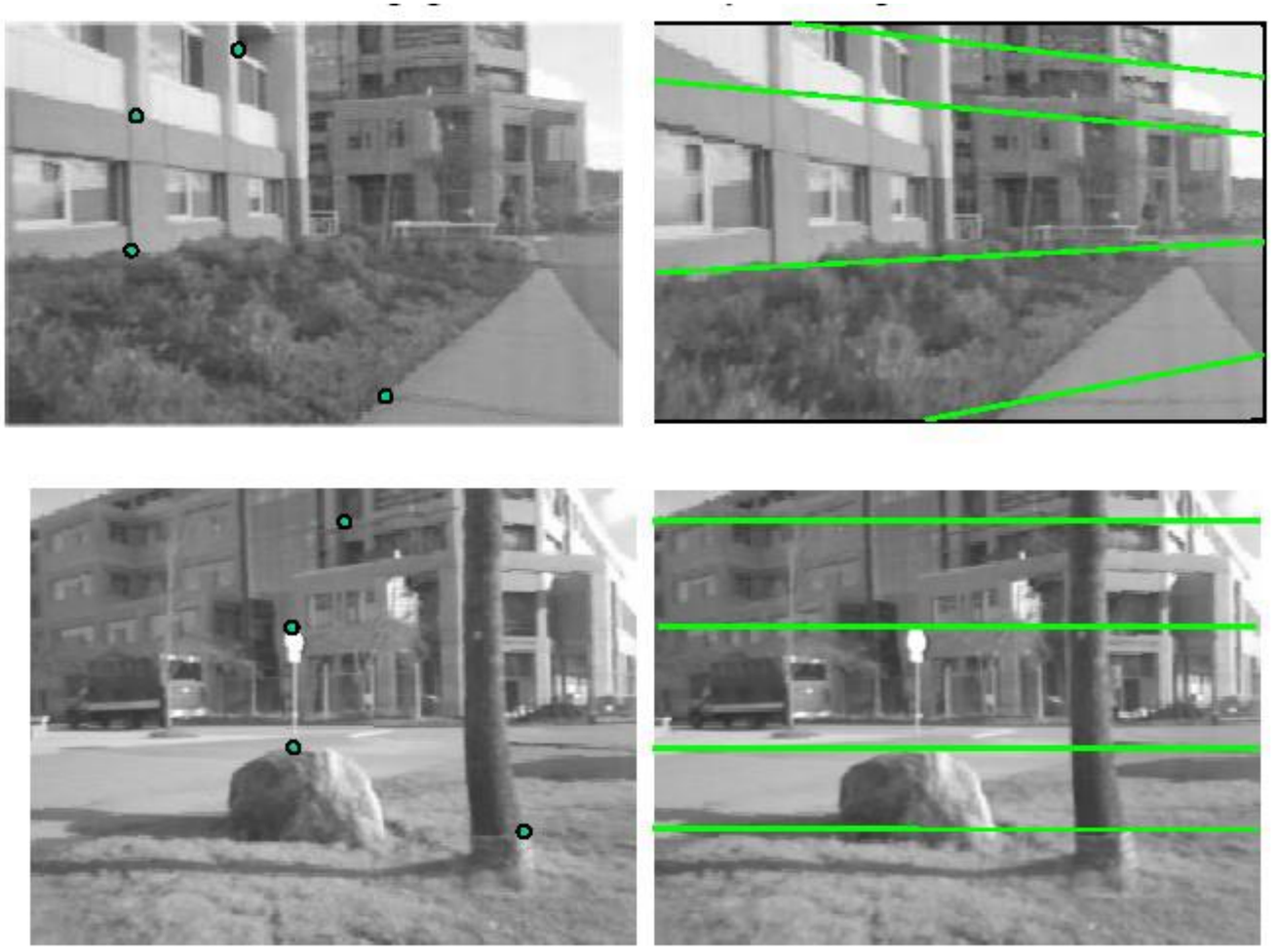
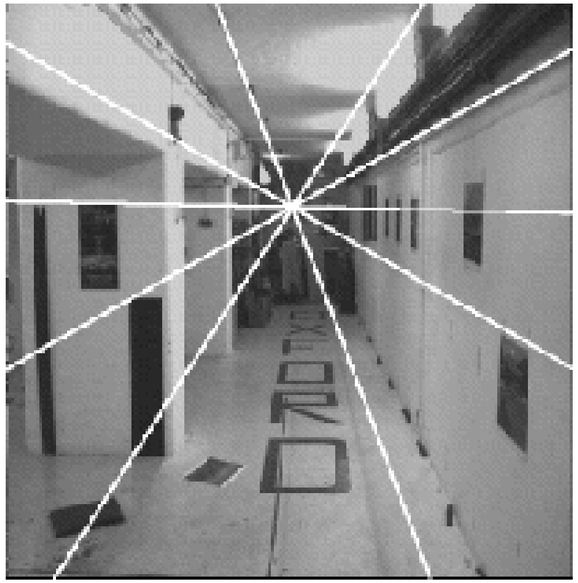


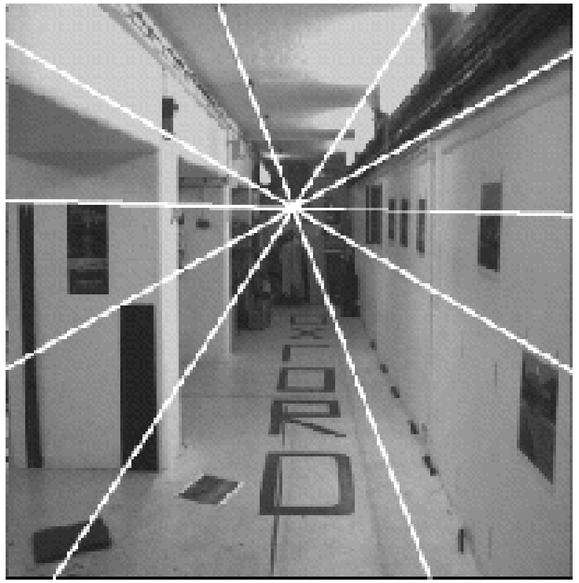
Figure from Hartley & Zisserman

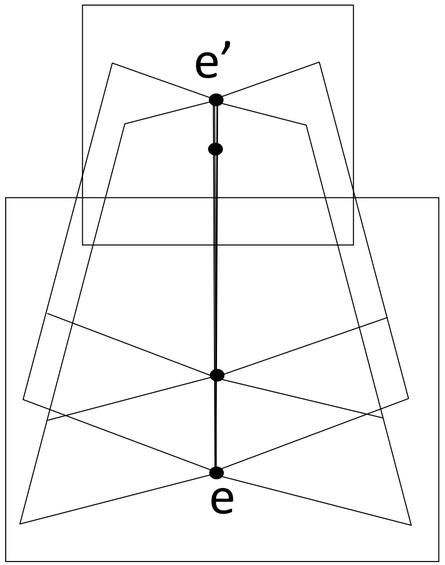
Example



Example: forward motion







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

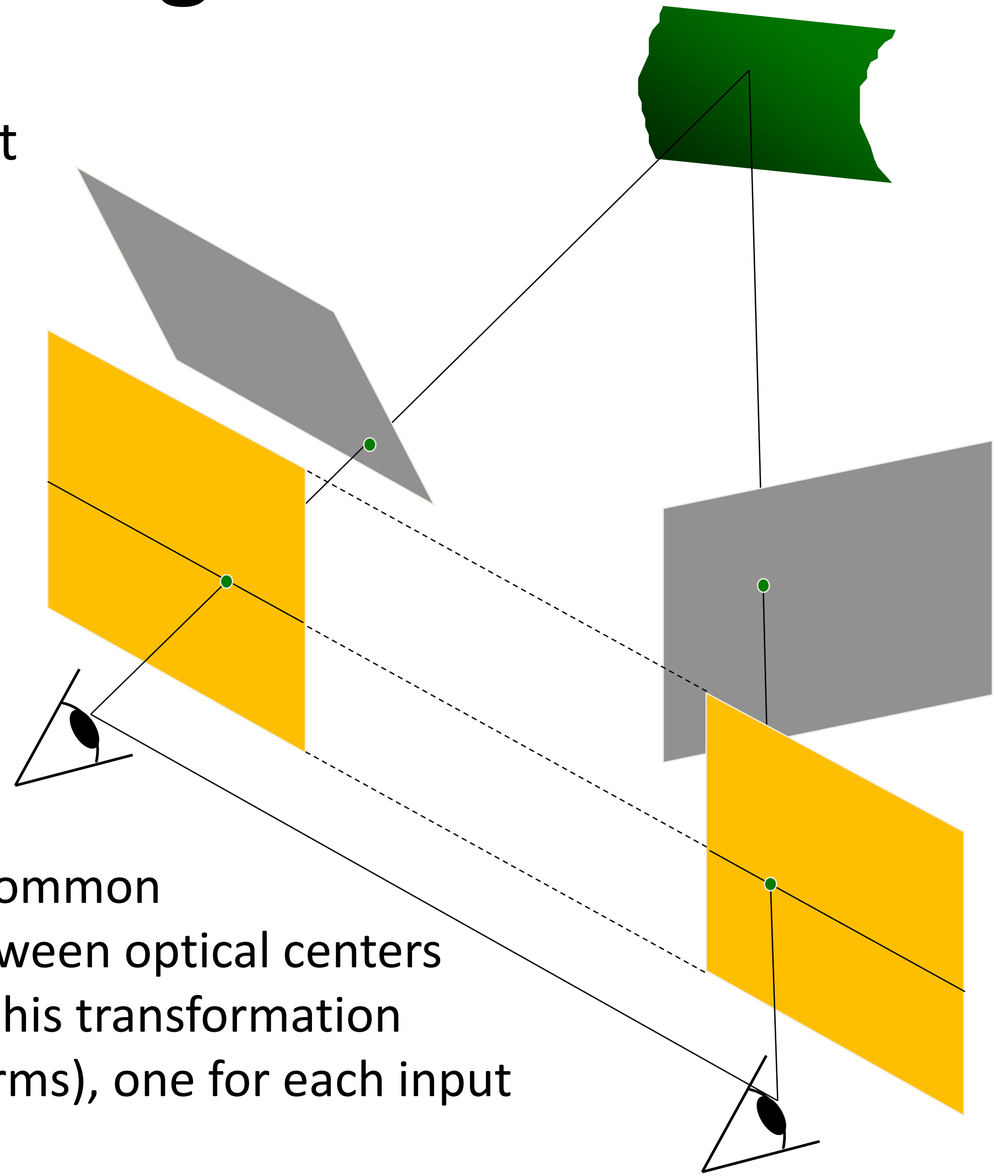
Figure from Hartley & Zisserman

The Epipole

Photo by Frank Dellaert

Stereo image rectification

In practice, it is convenient if image scanlines are the epipolar lines.



reproject image planes onto a common plane parallel to the line between optical centers

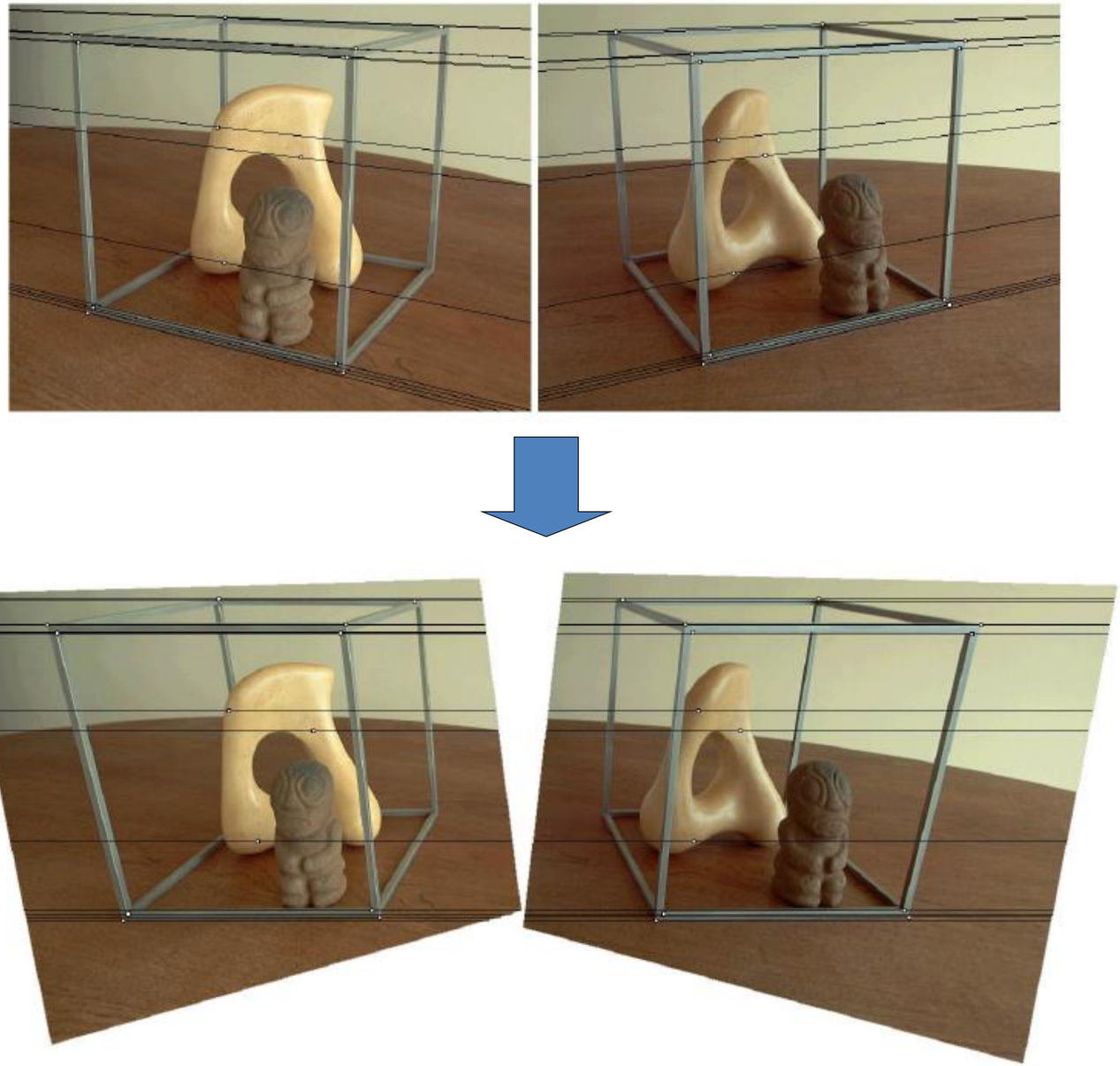
pixel motion is horizontal after this transformation

two homographies (3x3 transforms), one for each input image reprojection

Adapted from Li Zhang

C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision. CVPR 1999.

Stereo image rectification: example

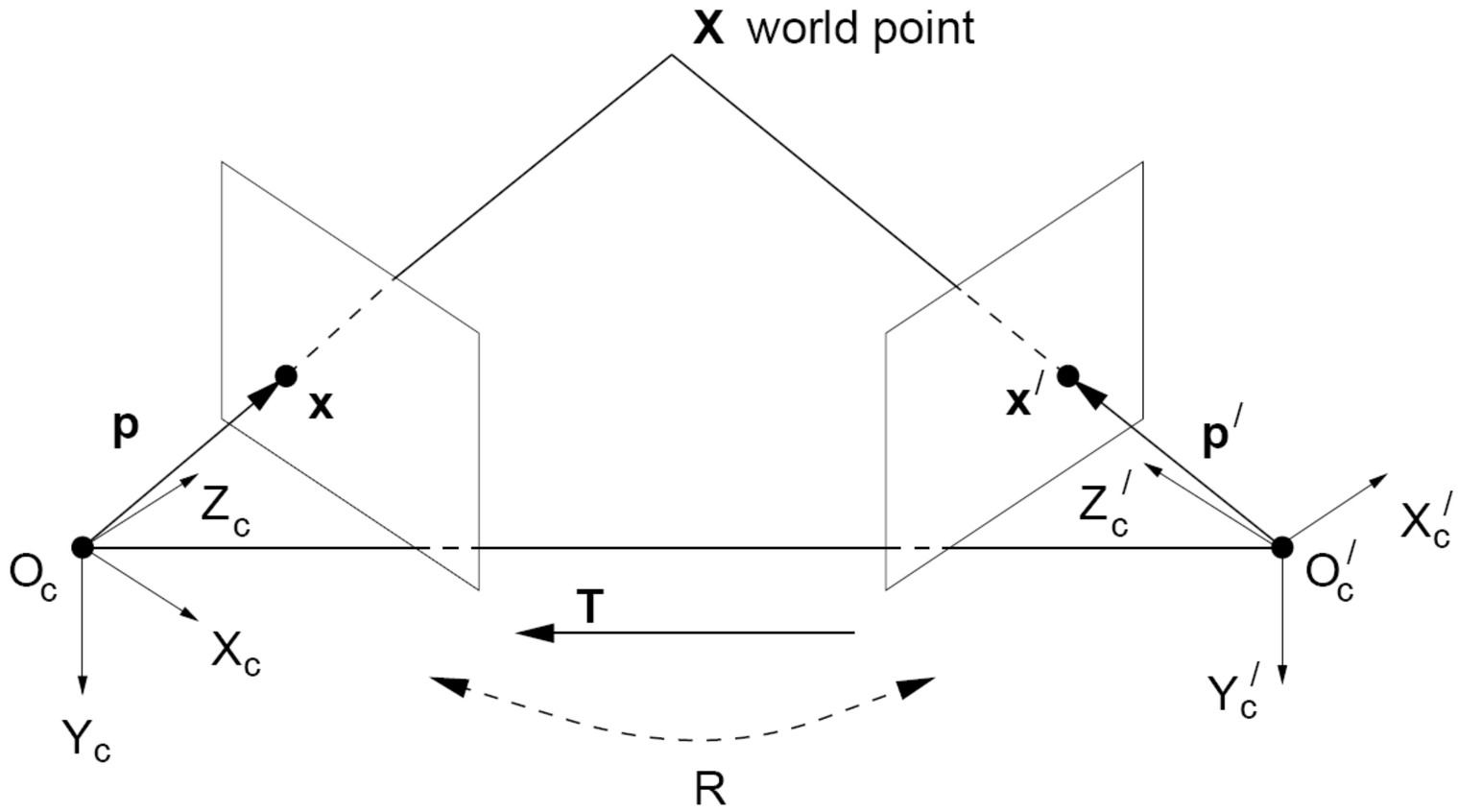


- For a given stereo rig, how do we express the epipolar constraints algebraically?

- For calibrated cameras, with **Essential Matrix**
- For uncalibrated cameras, with **Fundamental Matrix**

Deriving the Essential Matrix:

Stereo geometry, with calibrated cameras



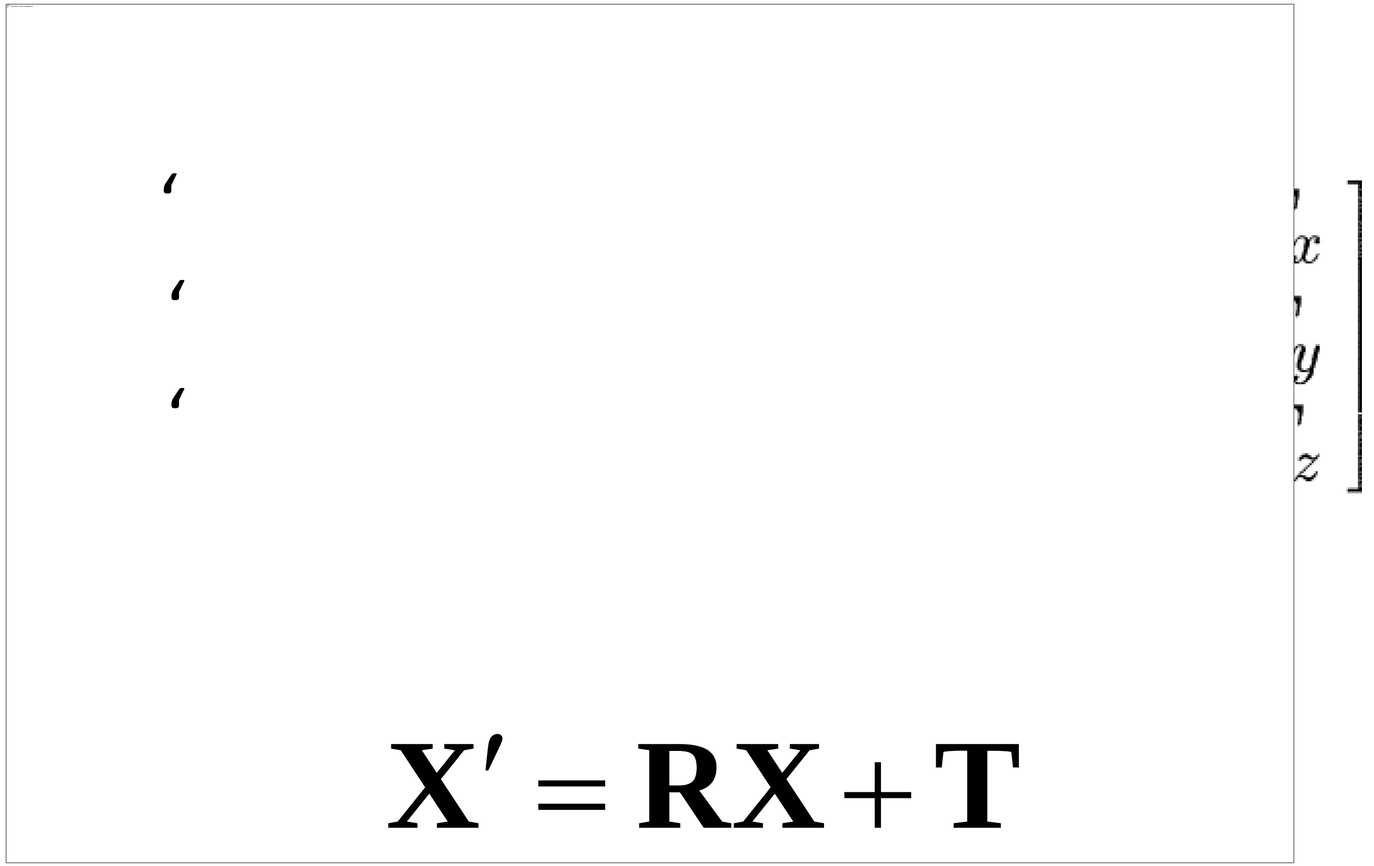
If the rig is calibrated, we know :

how to **rotate** and **translate** camera reference frame 1 to get to camera reference frame 2.

Rotation: 3 x 3 matrix; translation: 3 vector.

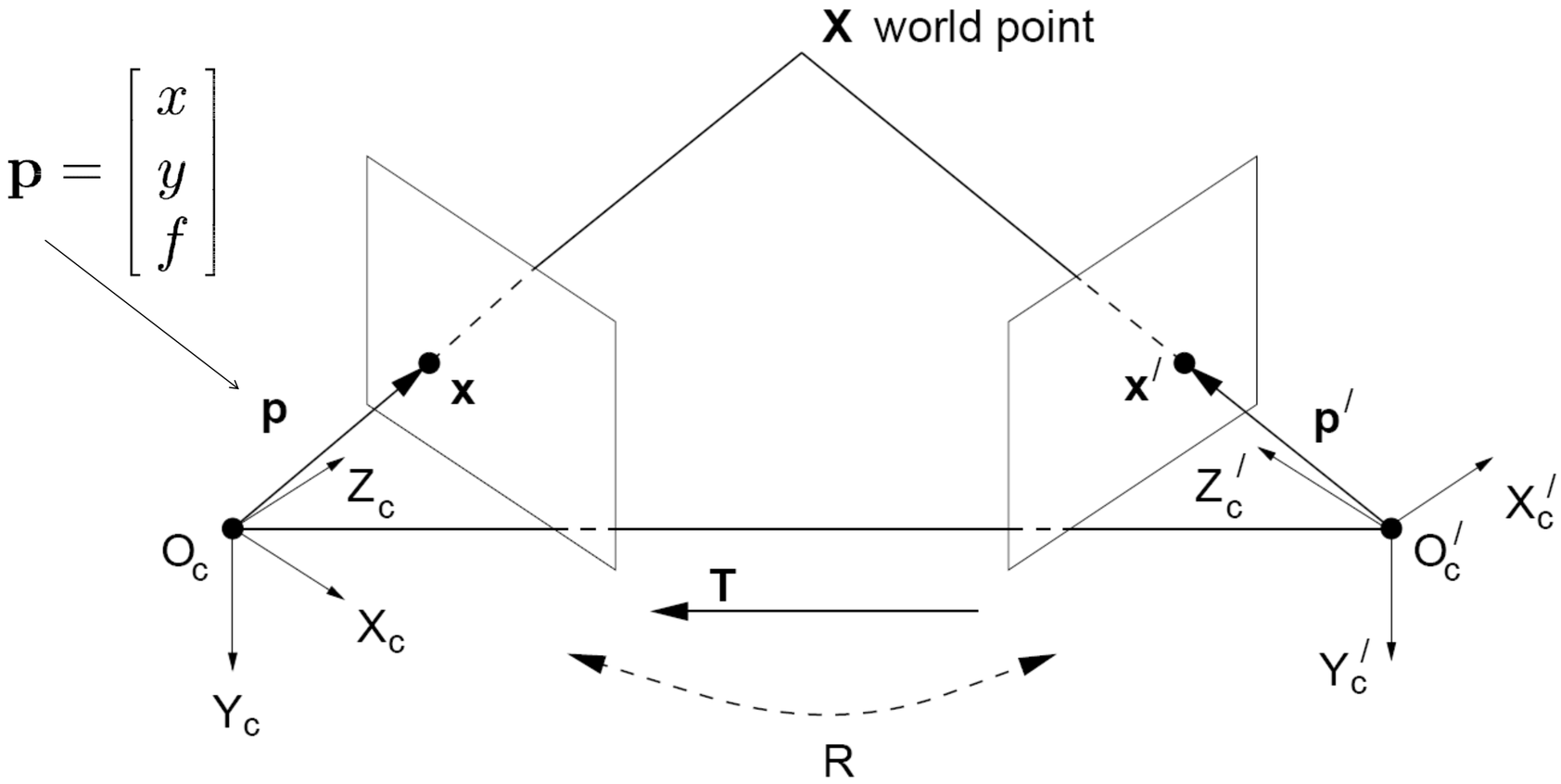
Deriving the Essential Matrix:

3d rigid transformation



Deriving the Essential Matrix:

Stereo geometry, with calibrated cameras



Camera-centered coordinate systems are related by known rotation **R** and translation **T**:

$$\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{T}$$

Review: Cross product

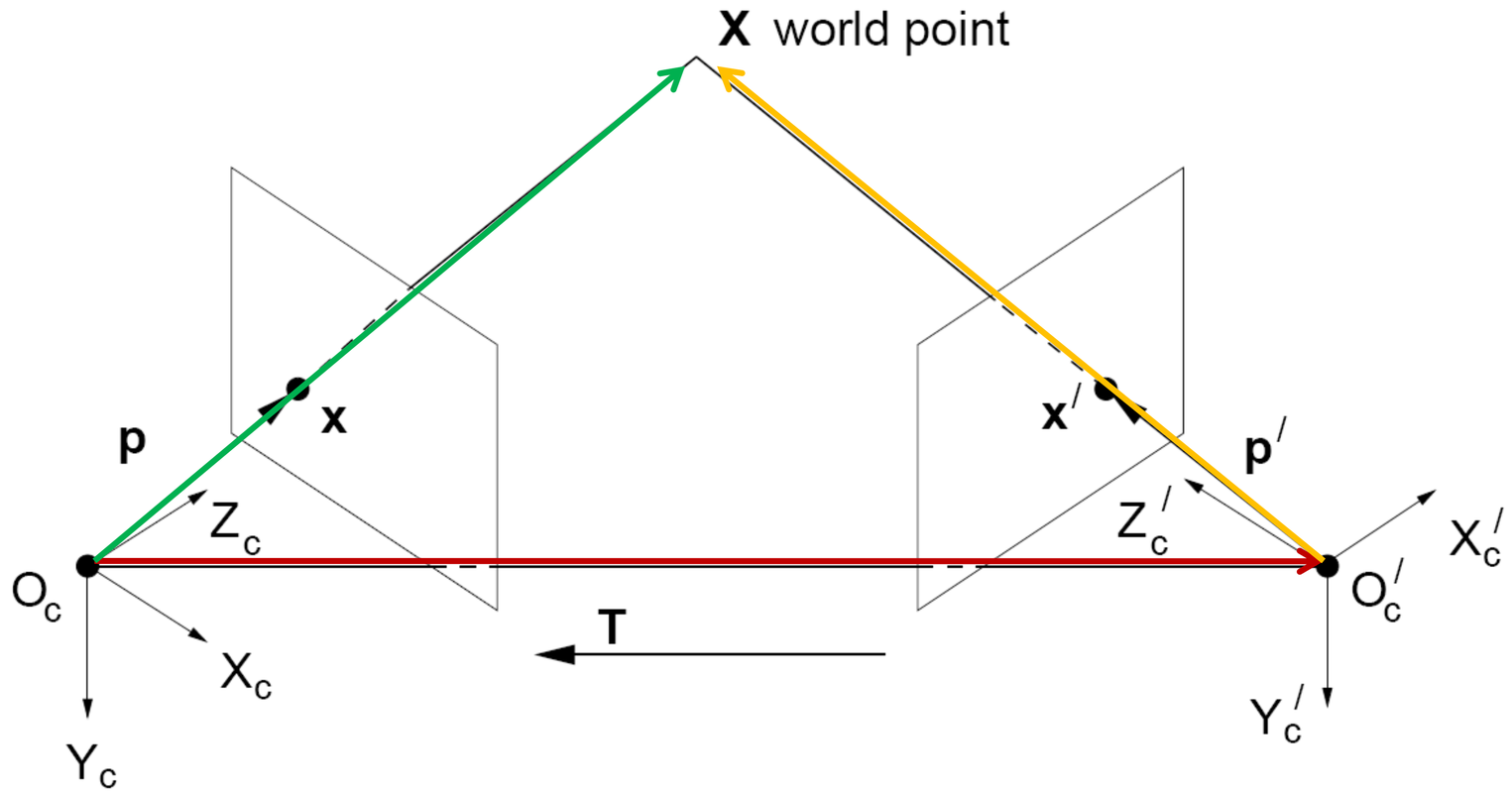
$$\vec{a} \times \vec{b} = \vec{c}$$

$$\vec{a} \cdot \vec{c} = 0$$
$$\vec{b} \cdot \vec{c} = 0$$

Vector cross product takes two vectors and returns a third vector that's perpendicular to both inputs.

So here, c is perpendicular to both a and b, which means the dot product = 0.

Deriving the Essential Matrix:
From geometry to algebra



$$\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{T}$$

$$\mathbf{T} \times \mathbf{X}' =$$

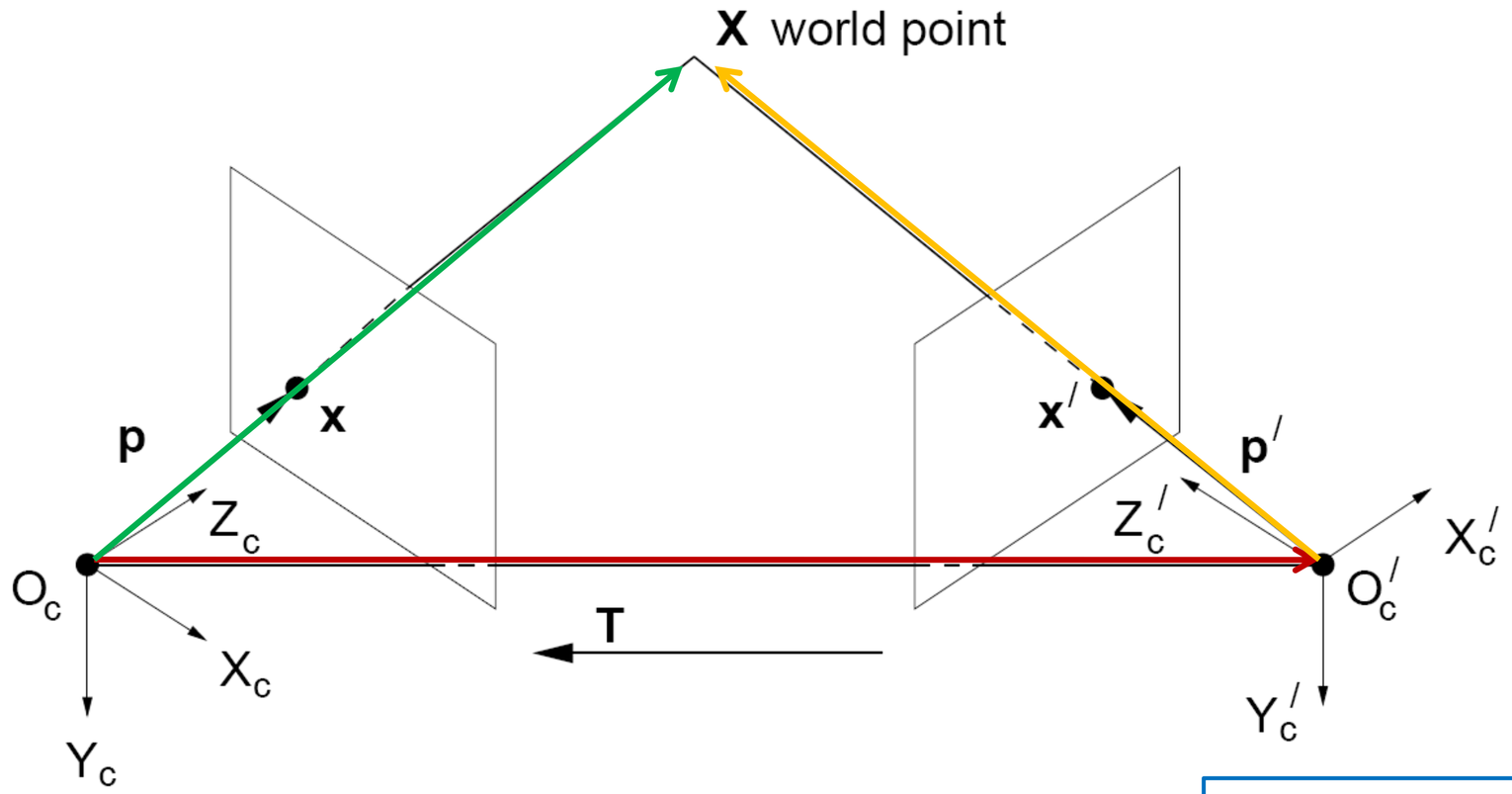
Normal to the plane

$$= \mathbf{T} \times \mathbf{R}\mathbf{X}$$

$$\mathbf{X}' \cdot (\mathbf{T} \times \mathbf{X}') = \mathbf{X}' \cdot (\mathbf{T} \times \mathbf{R}\mathbf{X})$$

$$= 0$$

Deriving the Essential Matrix:
From geometry to algebra



$$\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{T}$$

$$\mathbf{T} \times \mathbf{X}' = \mathbf{T} \times \mathbf{R}\mathbf{X} + \mathbf{T} \times \mathbf{T}$$

Normal to the plane

$$= \mathbf{T} \times \mathbf{R}\mathbf{X}$$

$$\mathbf{X}' \cdot (\mathbf{T} \times \mathbf{X}') = \mathbf{X}' \cdot (\mathbf{T} \times \mathbf{R}\mathbf{X})$$

$$= 0$$

Deriving the Essential Matrix:

Matrix form of cross product

$$\vec{a} \times \vec{b} \qquad = \vec{c} \qquad \begin{matrix} \vec{a} \cdot \vec{c} = 0 \\ \vec{b} \cdot \vec{c} = 0 \end{matrix}$$

Can be expressed as a matrix multiplication.

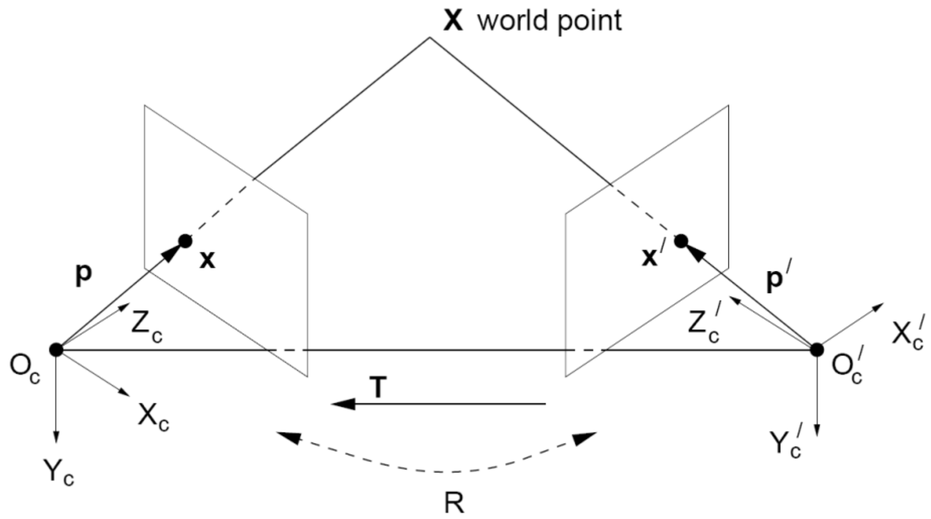
$$[a_x] = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

$$\vec{a} \times \vec{b} = [a_x]\vec{b}$$

Essential matrix

$$\mathbf{X}' \cdot \left(\mathbf{T}_x \ \ \mathbf{R}\mathbf{X} \right) = 0$$

Let $\mathbf{E} = \mathbf{T}_x \mathbf{R}$



This holds for the rays **p** and **p'** that are parallel to the camera-centered position vectors **X** and **X'**, so we have:

$$\mathbf{p}'^{T} \mathbf{E}\mathbf{p} = 0$$

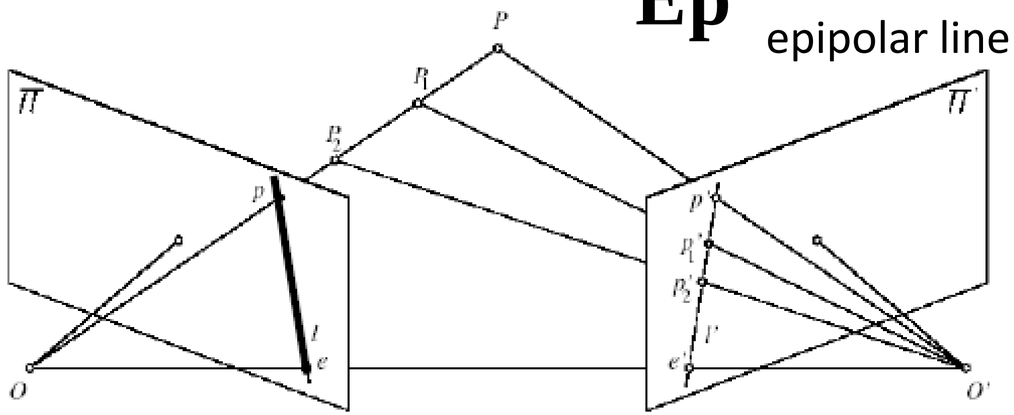
E is called the **essential matrix**, which relates corresponding image points [Longuet-Higgins 1981]

Essential matrix and epipolar lines

$$\mathbf{p'}^{\mathrm{T}}\mathbf{E}\mathbf{p} = 0$$

Epipolar constraint: if we observe point **p** in one image, then its position **p'** in second image must satisfy this equation.

$\mathbf{Ep}$ is the coordinate vector representing the epipolar line associated with point p



$\mathbf{E}^T\mathbf{p'}$ is the coordinate vector representing the epipolar line associated with point p'

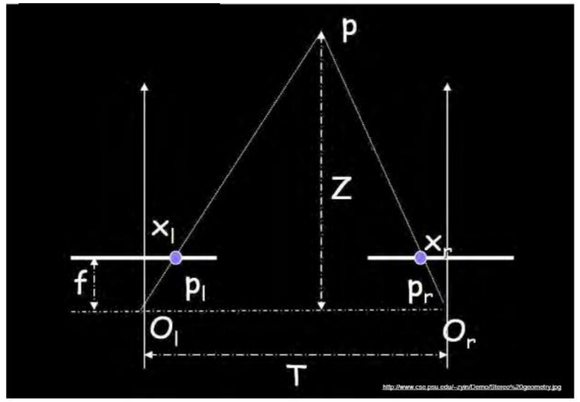
Essential matrix: properties

- Relates image of corresponding points in both cameras, given rotation and translation
- Assuming intrinsic parameters are known

$$\mathbf{E} = \mathbf{T}_x\mathbf{R}$$

- $E\,x'$ 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
  - (3 for R, 2 for t because it's up to a scale)

Essential matrix example: parallel cameras



$$\mathbf{R} =$$

$$\mathbf{T} =$$

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

$$\mathbf{p}'^{\mathrm{T}}\mathbf{E}\mathbf{p} = 0$$

For the parallel cameras, image of any point must lie on same horizontal line in each image plane.

Weak calibration

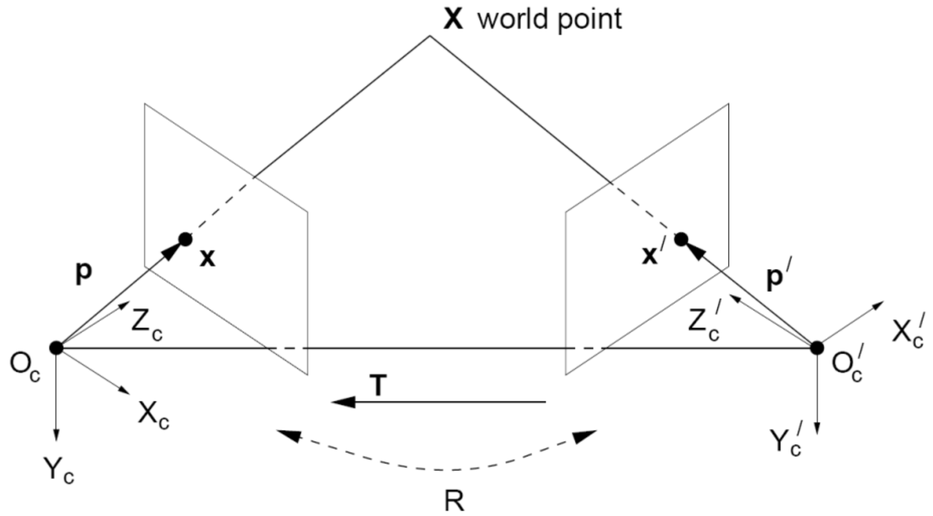
- So far, we have assumed calibrated cameras and were able to perform dense stereo estimation

- What if we want to estimate world geometry without requiring calibrated cameras?
  - Archival videos
  - Photos from multiple unrelated users
  - Dynamic camera system

Uncalibrated cameras

$$\mathbf{E} = \mathbf{T}_x\mathbf{R}$$

$$\mathbf{p}'^T\,\mathbf{E}\mathbf{p} = 0$$



- For an *uncalibrated* stereo rig, can we we express the epipolar constraints algebraically via the **Essential Matrix**?
- No, we do not know T or R
- However we can use the **Fundamental Matrix**
  - Estimate epipolar geometry from a (redundant) set of point correspondences between two uncalibrated cameras

Uncalibrated case

For a given camera:

$$\overline{\mathbf{p}} = \mathbf{M}_{\text{int}}\mathbf{p}$$

Camera coordinates

So, for two cameras (left and right):

$$\mathbf{p}_{(left)} = \mathbf{M}^{-1}_{left,\text{int}}\overline{\mathbf{p}}_{(left)}$$

$$\mathbf{p}_{(right)} = \mathbf{M}^{-1}_{right,\text{int}}\overline{\mathbf{p}}_{(right)}$$

Camera coordinates

Image pixel coordinates

Internal calibration matrices, one per camera

$$\mathbf{p}_{(left)} = \mathbf{M}_{left,\text{int}}^{-1}\,\overline{\mathbf{p}}_{(left)}$$

$$\mathbf{p}_{(right)} = \mathbf{M}_{right,\text{int}}^{-1}\,\overline{\mathbf{p}}_{(right)}$$

Uncalibrated case:
Fundamental matrix

$$\boxed{{}^{c}\mathbf{p}_{(right)}{}^{\mathrm{T}}\mathbf{E}\mathbf{p}_{(left)} = 0}$$

From before, the **essential** matrix **E.**

$$\left(\mathbf{M}_{right,\text{int}}^{-1}\,\overline{\mathbf{p}}_{right}\right)^{\mathrm{T}}\mathbf{E}\left(\mathbf{M}_{left,\text{int}}^{-1}\,\overline{\mathbf{p}}_{left}\right) = 0$$

$$\overline{\mathbf{p}}_{right}^{\mathrm{T}}\underbrace{\left(\mathbf{M}_{right,\text{int}}^{-\mathrm{T}}\mathbf{E}\mathbf{M}_{left,\text{int}}^{-1}\right)}\overline{\mathbf{p}}_{left} = 0$$

$$\overline{\mathbf{p}}_{right}^{\mathrm{T}}\boxed{\mathbf{F}}\overline{\mathbf{p}}_{left} = 0$$

Fundamental matrix

Fundamental matrix

- Relates pixel coordinates in the two views
- More general form than essential matrix: we remove need to know intrinsic parameters
- If we estimate fundamental matrix from correspondences in pixel coordinates, can reconstruct epipolar geometry without intrinsic or extrinsic parameters

Computing F from correspondences

$$\mathbf{F} = \left(\mathbf{M}_{right,\text{int}}^{-\text{T}} \mathbf{E} \mathbf{M}_{left.\text{int}}^{-1} \right)$$

$$\overline{\mathbf{p}}_{right}^{\text{T}} \mathbf{F} \overline{\mathbf{p}}_{left} = 0$$

- Cameras are uncalibrated: we don't know **E** or left or right $\mathbf{M}_{\text{int}}$ matrices
- Estimate F from 8+ point correspondences.

Computing F from correspondences

Each point correspondence generates one constraint on F

$$\overline{\mathbf{p}}_{right}^{\mathrm{T}}\mathbf{F}\overline{\mathbf{p}}_{left} = 0$$

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

Collect n of these constraints

$$\begin{bmatrix} u'_1u_1 & u'_1v_1 & u'_1 & v'_1u_1 & v'_1v_1 & v'_1 & u_1 & v_1 & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = \mathbf{0}$$

Solve for f , vector of parameters.

Rank constraint

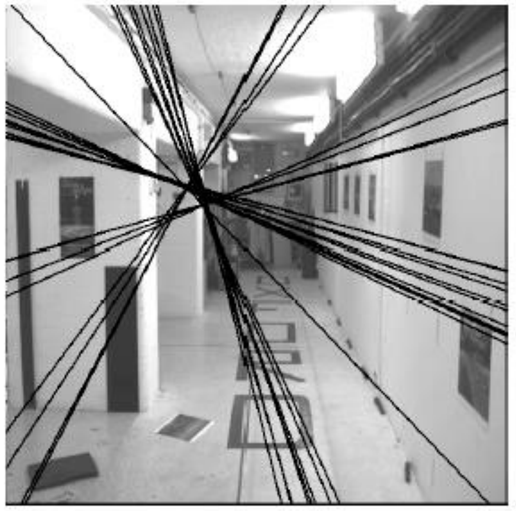
$$\boldsymbol{x} = (u, v, 1)^T, \quad \boldsymbol{x}' = (u', v', 1)$$

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

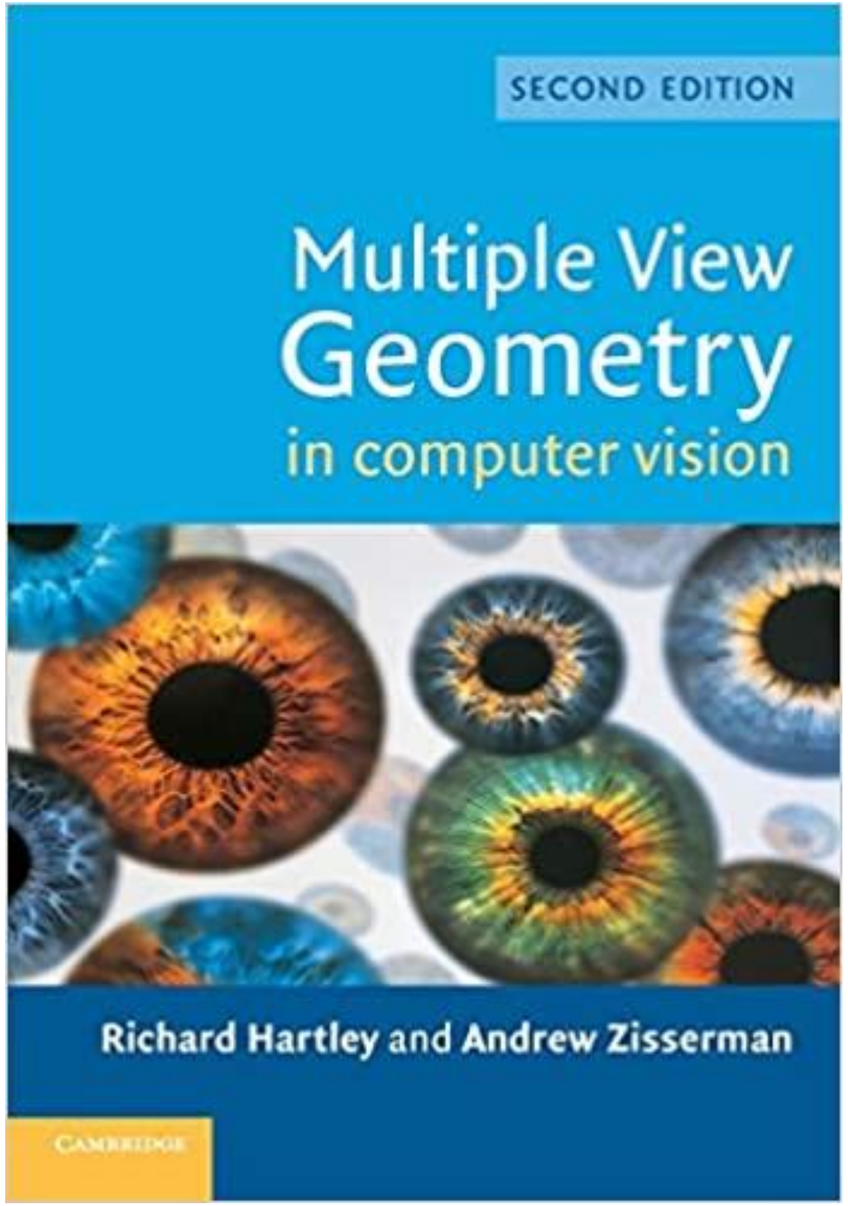
$$\begin{bmatrix} u'u & u'v & u' & v'u & v'v & v' & u & v & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

Solve homogeneous linear system using eight or more matches

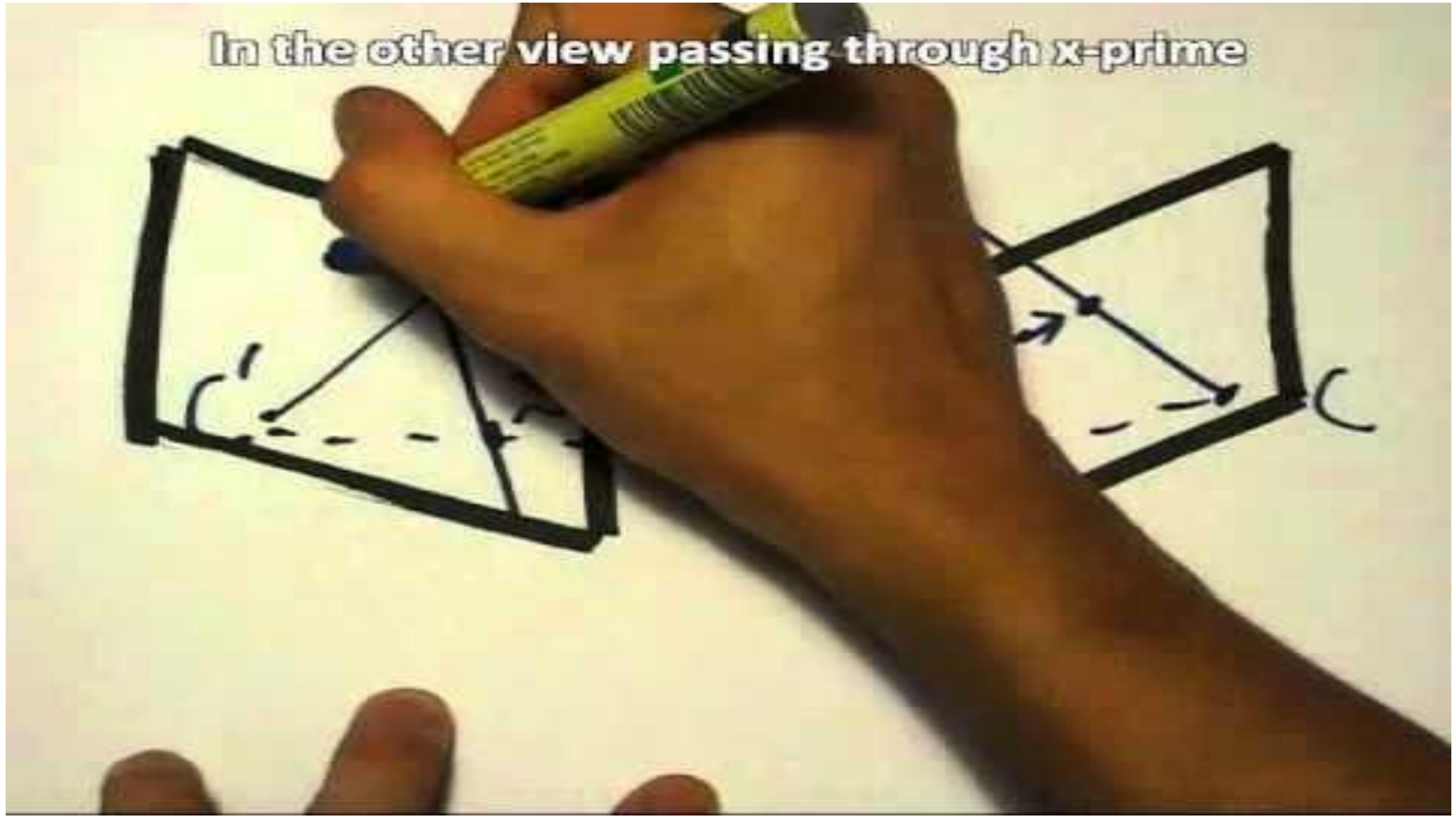
Enforce rank-2 constraint (take SVD of ***F*** and throw out the smallest singular value)



The Bible by Hartley & Zisserman



The Fundamental Matrix Song



http://danielwedge.com/fmatrix/
https://www.youtube.com/watch?time_continue=8&v=DgGV3l82NTk&feature=emb_title