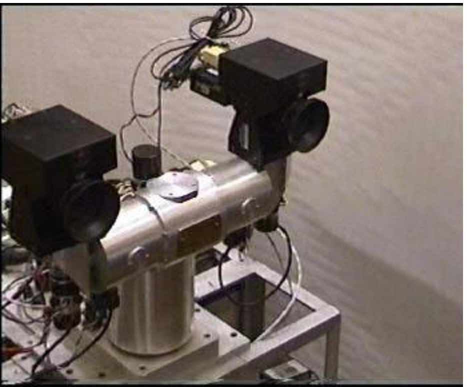
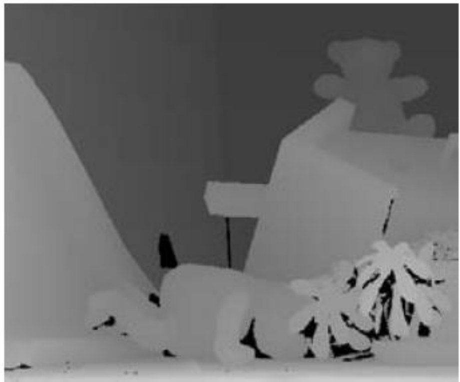
计算机视觉

Computer Vision

Lecture 10 Stereopsis

张 超

信息科学技术学院 智能科学系

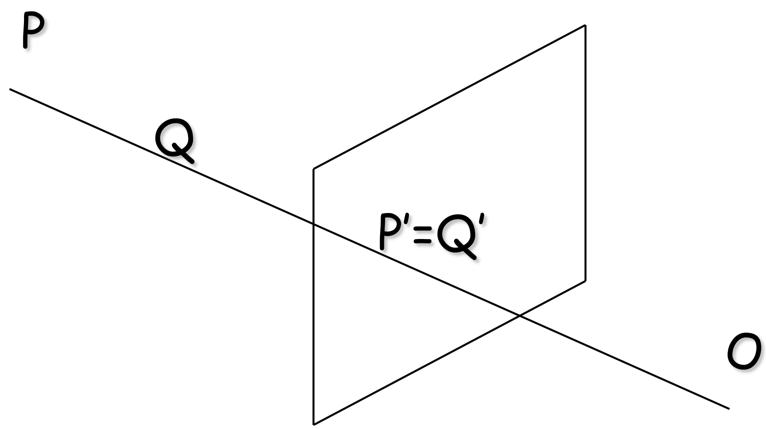
   

Stereopsis

- Triangulation
- Epipolar Geometry
  - The Essential Matrix
  - The Fundamental Matrix
- Stereo correspondence
  - Local methods for binocular fusion
  - Global methods for binocular fusion
- Multi-view stereo

Why Stereo Vision?

- **2D images project 3D points into 2D:**



- **3D Points on the same viewing line have the same 2D image:**
  - **2D imaging results in depth information loss**

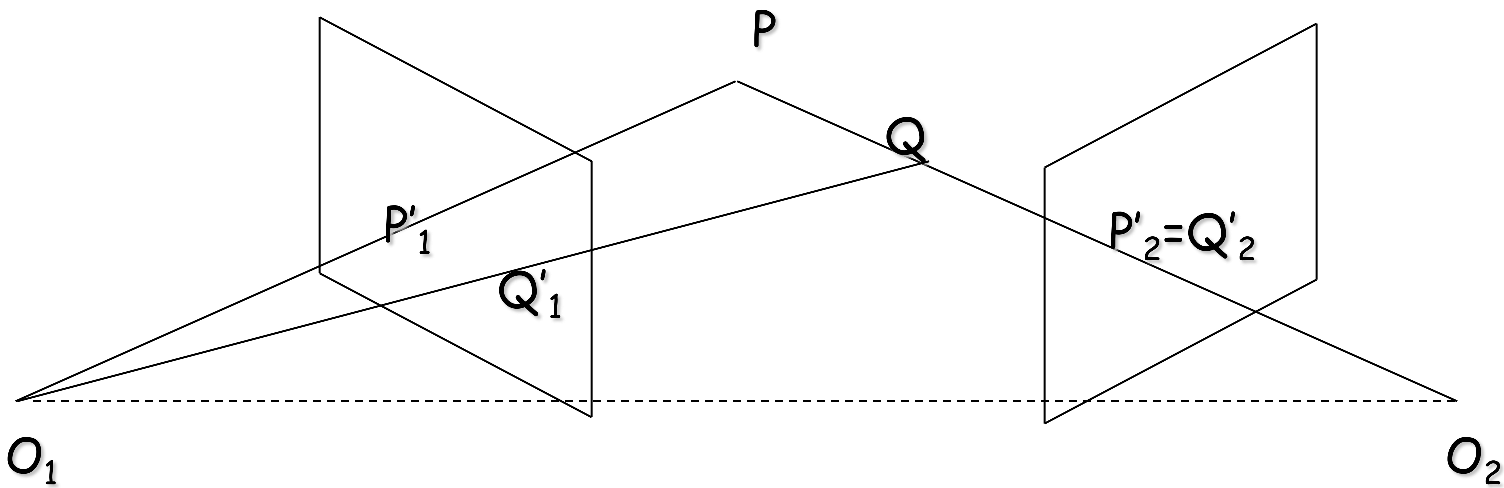
Stereo Vision

- **Refers to the ability of:**

  The ability to infer information on the 3D structure and distance of a scene from two or more images taken from different viewpoints.

Triangulation

Recovering Depth Information:

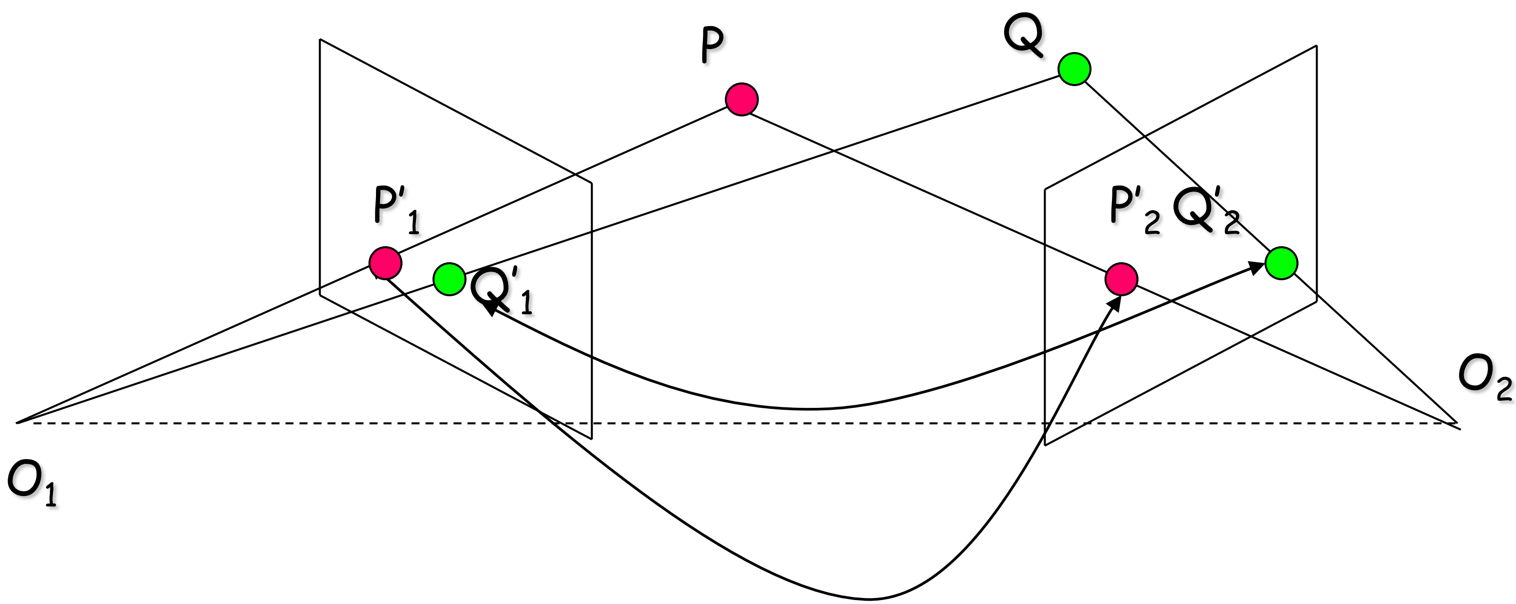


Depth can be recovered with two images and triangulation.

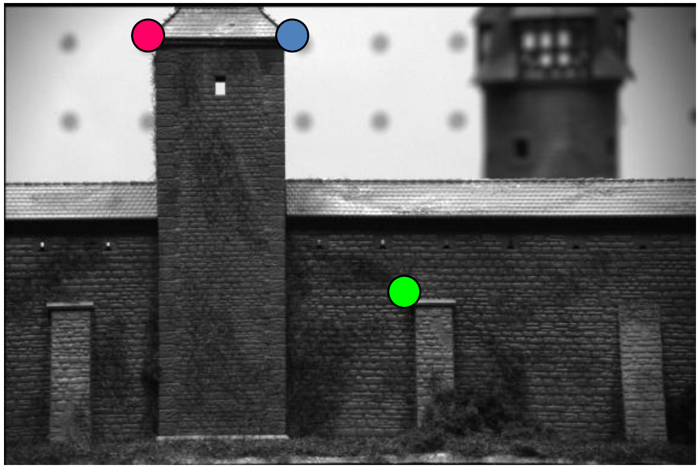
Stereo Vision Problems:

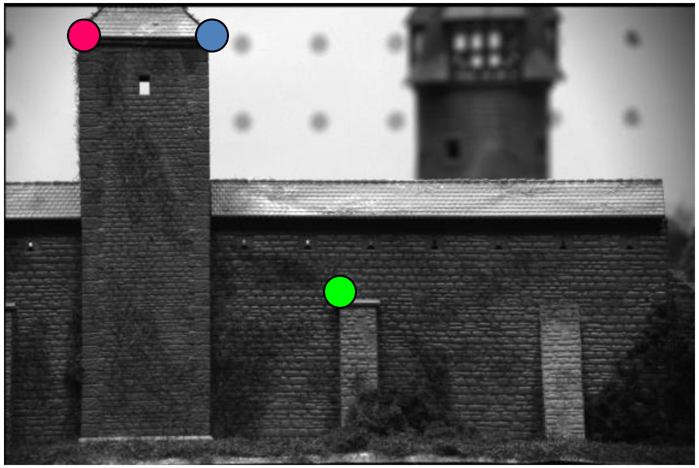
- Correspondence Problem:
  - Determining which pixel on the left corresponds to which pixel on the right.
- Reconstruction Problem:
  - Given a number of correspondence pairs and camera geometry information, find location and 3D structure of the observed objects.

Finding Correspondences:

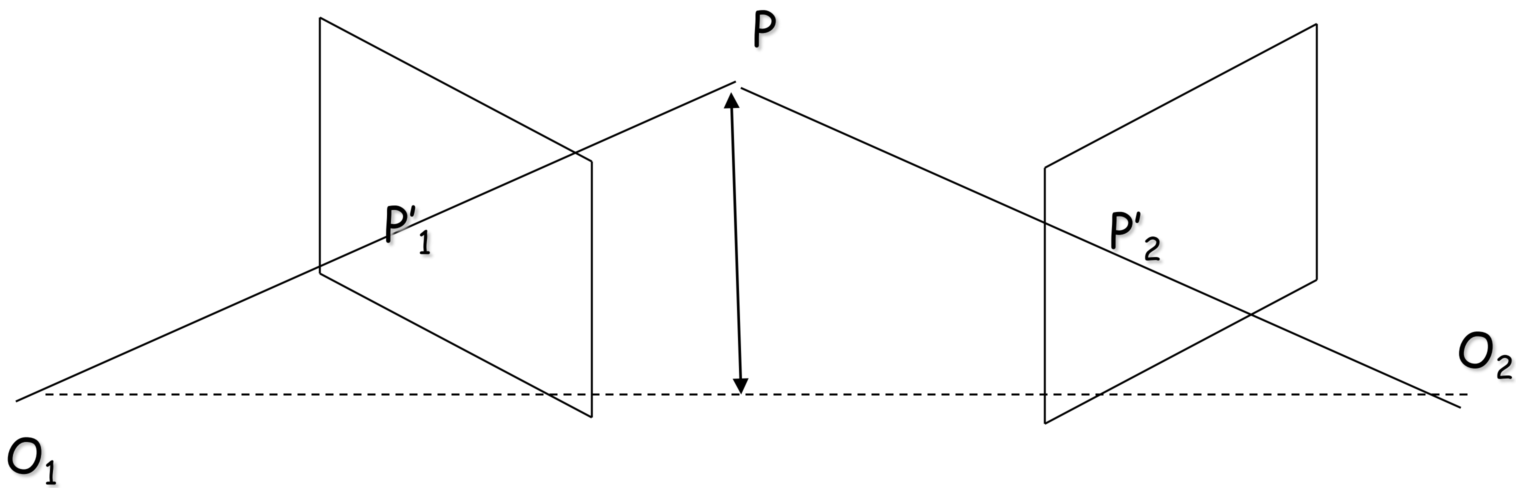


Finding Correspondences:



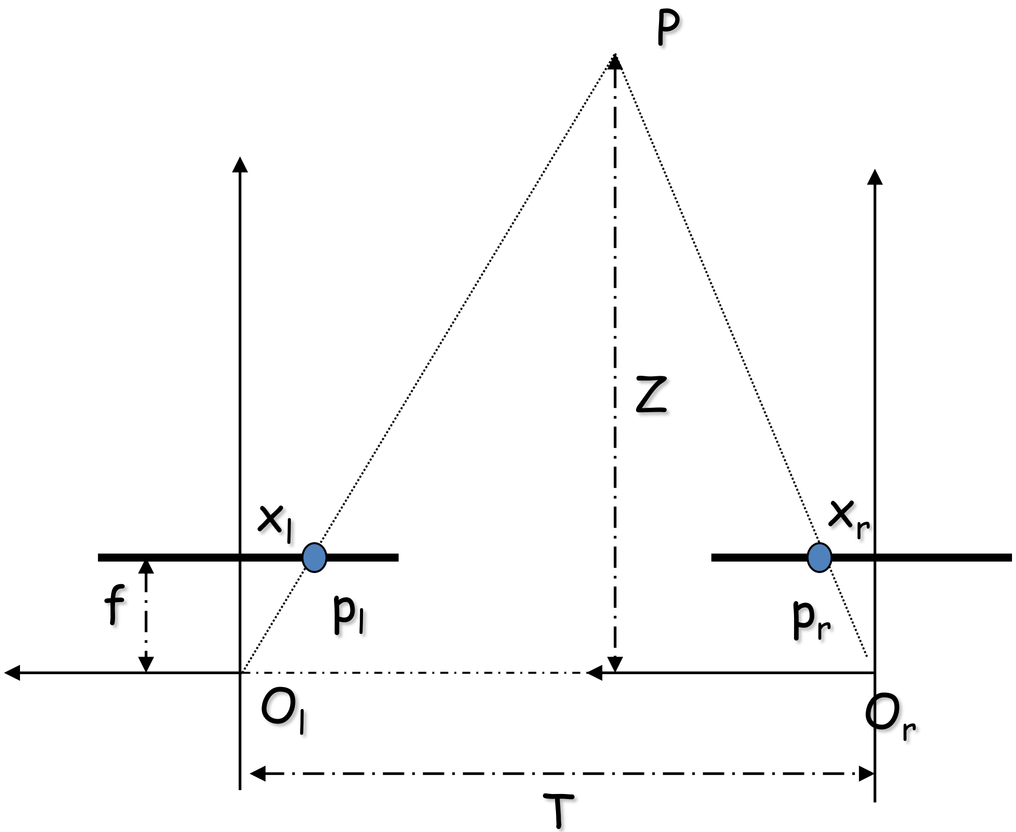


3D Reconstruction



We must solve the correspondence problem first!

A simple stereo system



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

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

视差

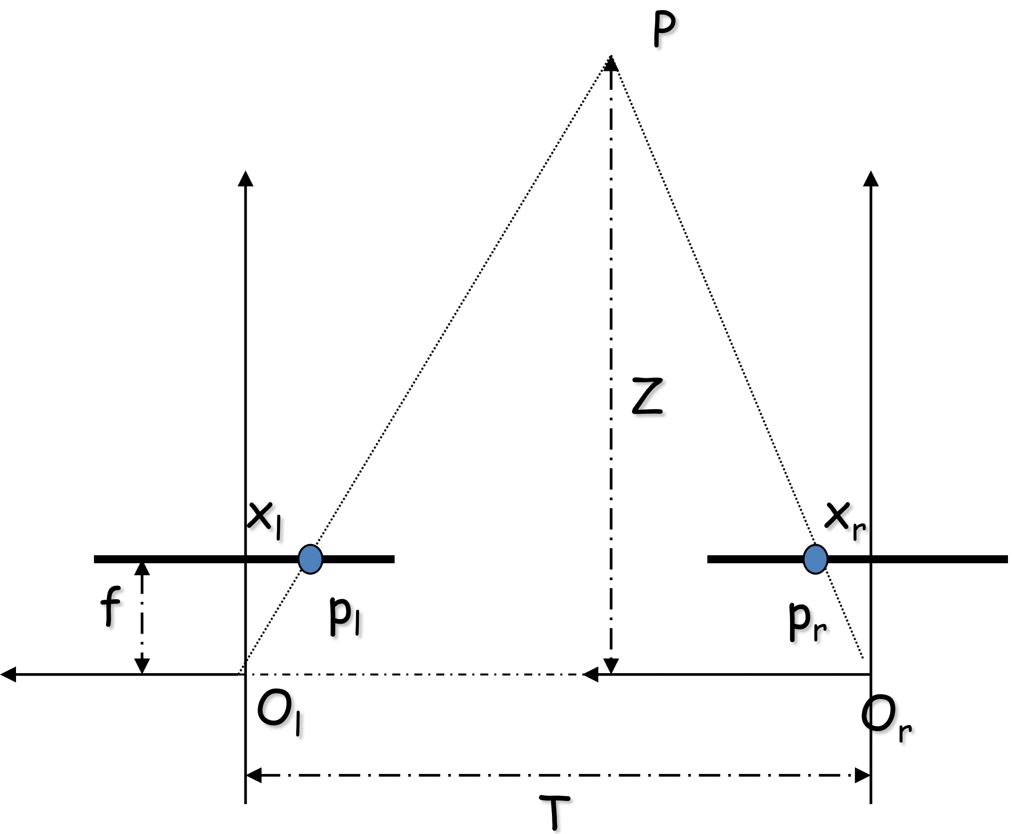
Disparity: $d = x_r - x_l$

$$\boxed{Z = f\frac{T}{d}}$$

T is the stereo baseline

d measures the difference in retinal position between corresponding points

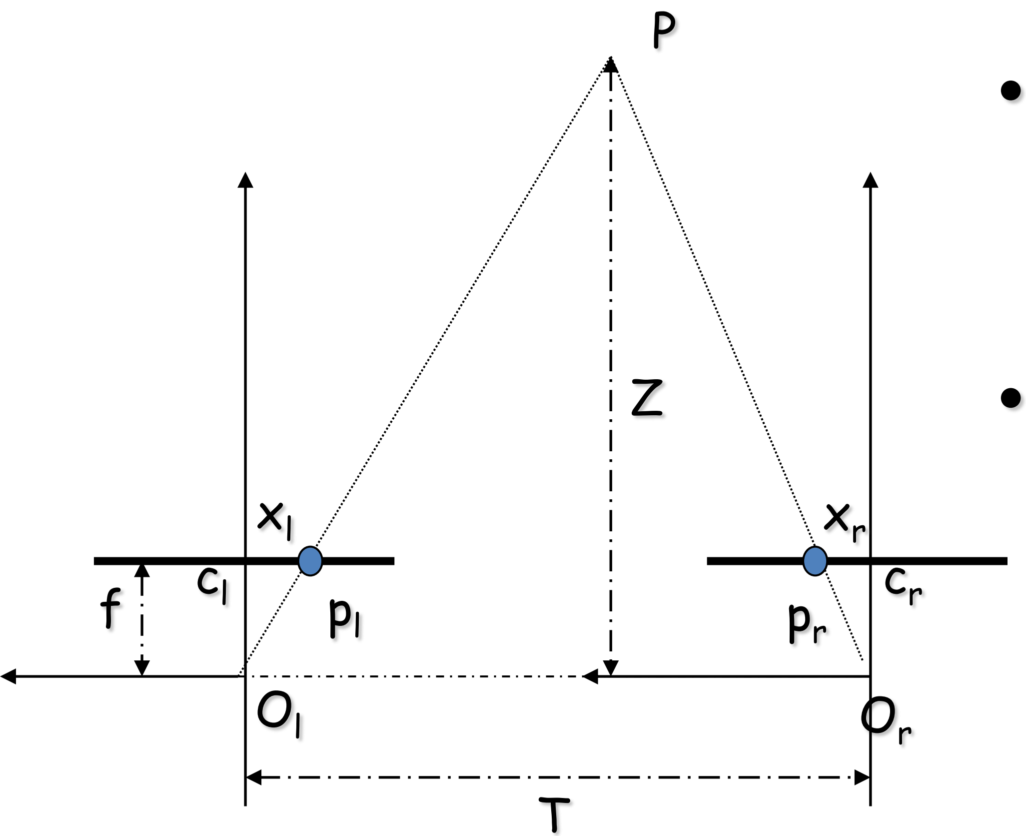
A simple stereo system



$$Z = f\frac{T}{d}$$

- Depth is inversely proportional to disparity

Parameters of a stereo system



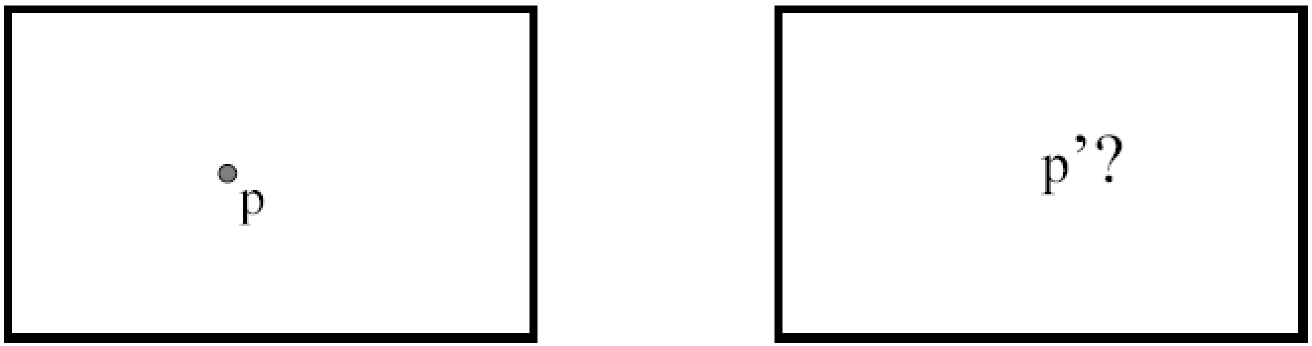
- **Intrinsic:**
  - **f: focal length of cameras**
  - **c_l and c_r: principal points**
- **Extrinsic:**
  - **T: stereo baseline**
  - **Transformation between cameras for a more general configuration**

Epipolar Geometry

Epipolar Geometry: Introduction

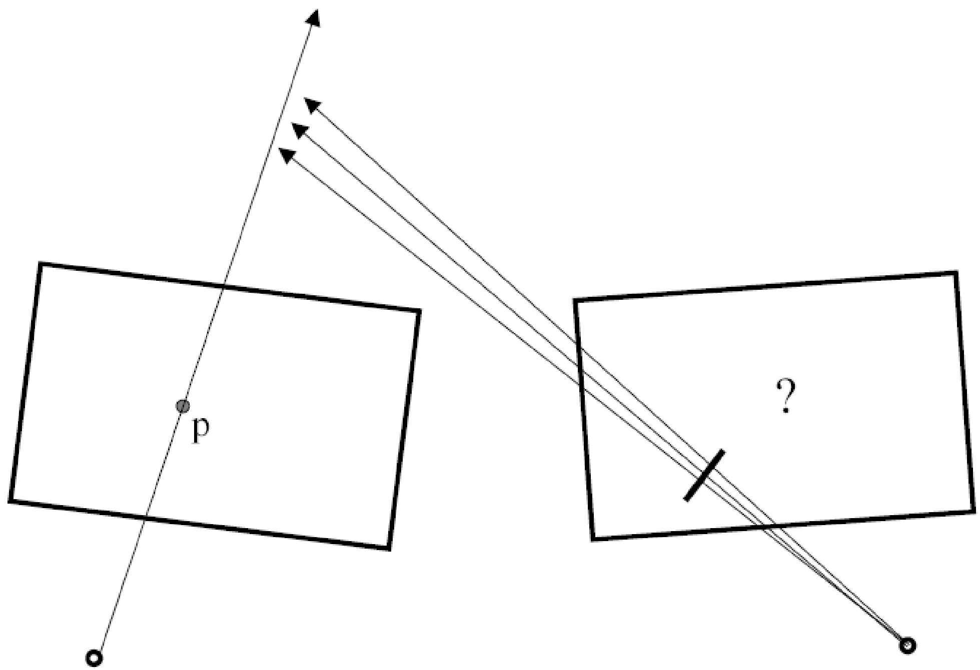
- **Depth can be reconstructed based on corresponding points (disparity)**
- **Finding corresponding points is hard & computationally expensive**
- **Epipolar geometry helps to significantly reduce search from 2-D to 1-D line.**

Constraining the Search Space



- **Finding correspondences is a search problem.**
  - **Given p in left image, where can corresponding point p' be?**
  - **Could be anywhere! Might not be same scene!**

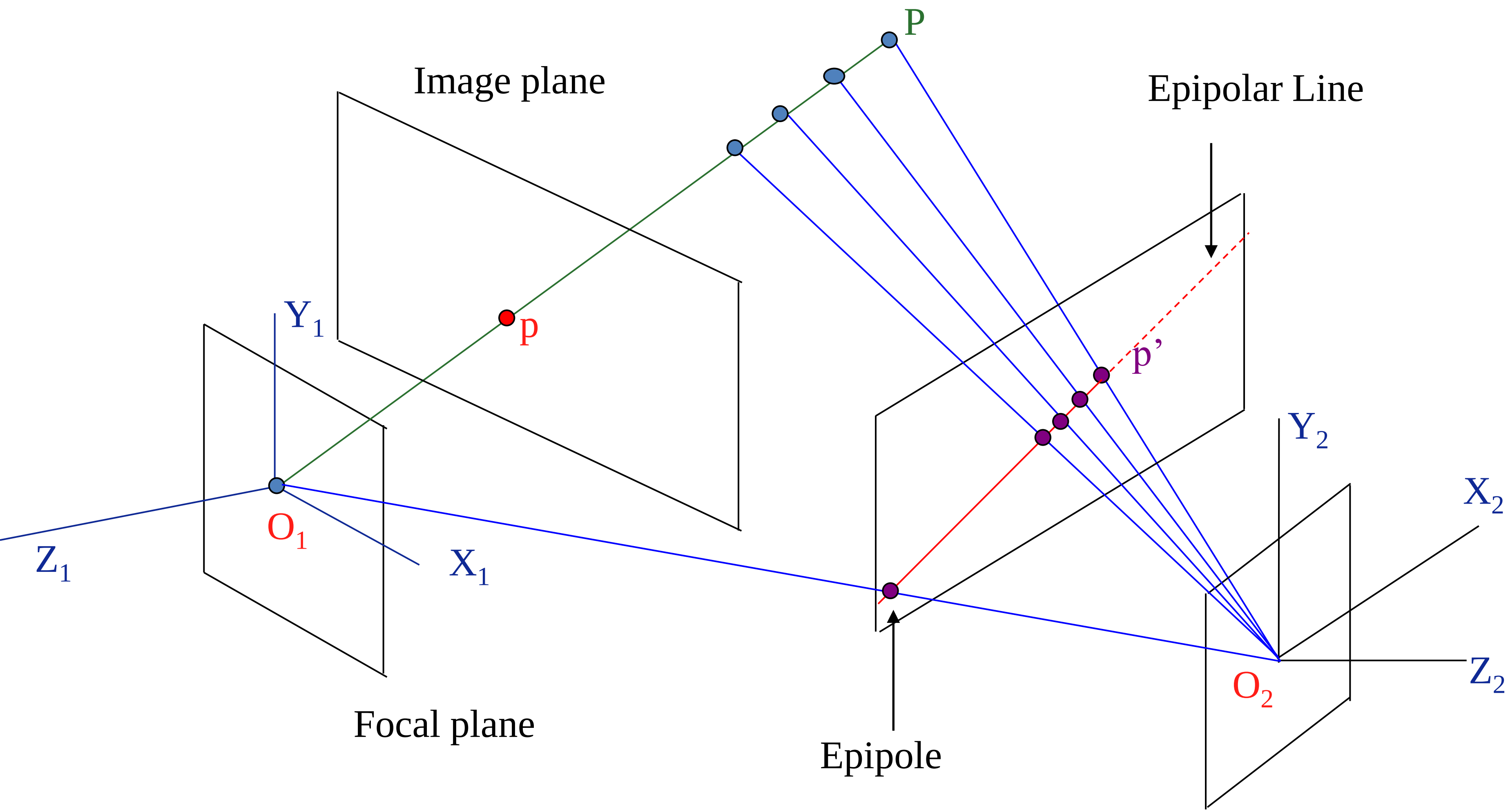
Constraining the Search Space



- **Geometry can be used to constrain the search.**

Epipolar Geometry

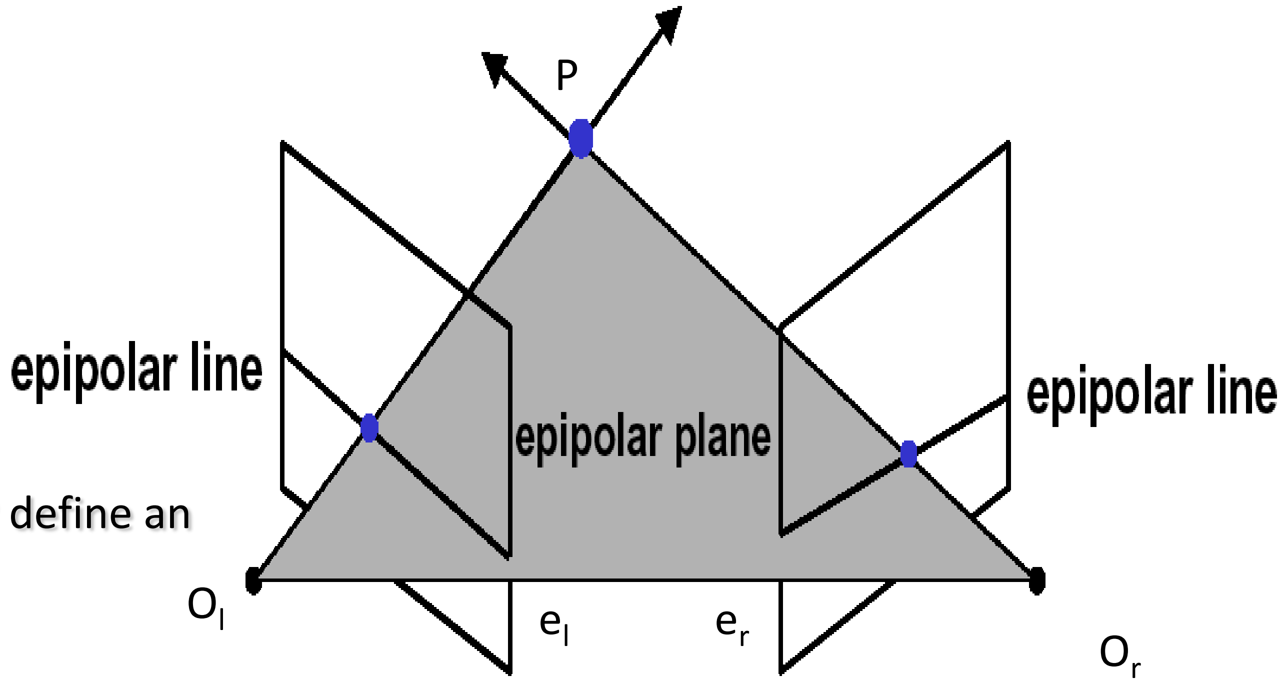
Stereo Constraints



Epipolar Constraint

Epipoles:

- e_l: left image of O_r
- e_r: right image of O_l



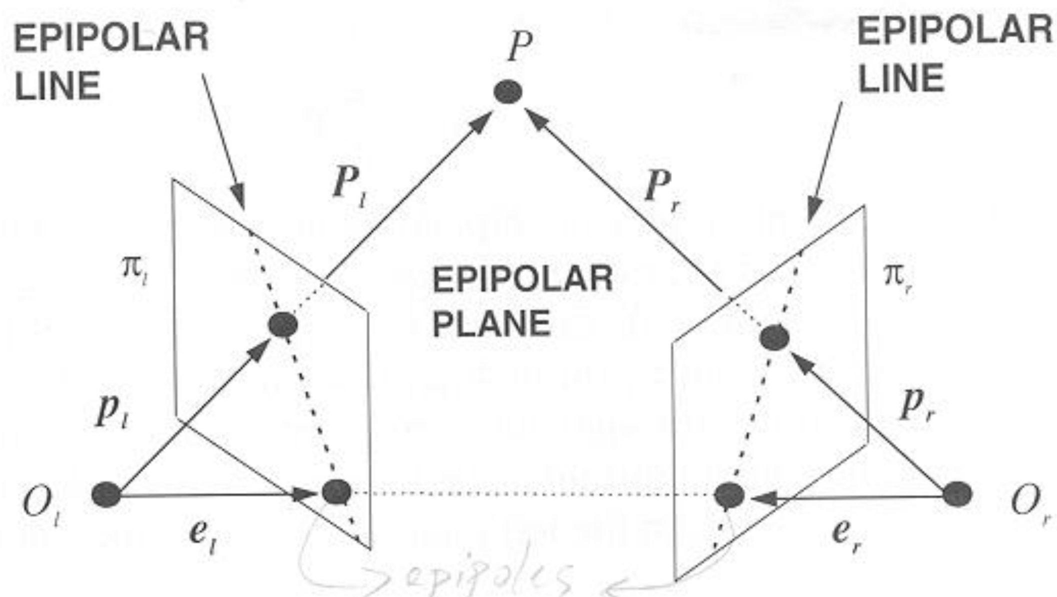
Epipolar plane:

- Three points: O_l,O_r, and P define an epipolar plane

Epipolar lines and epipolar constraint:

- Intersections of epipolar plane with the image planes
- Corresponding points are on “conjugate” epipolar lines

Epipolar Constraint



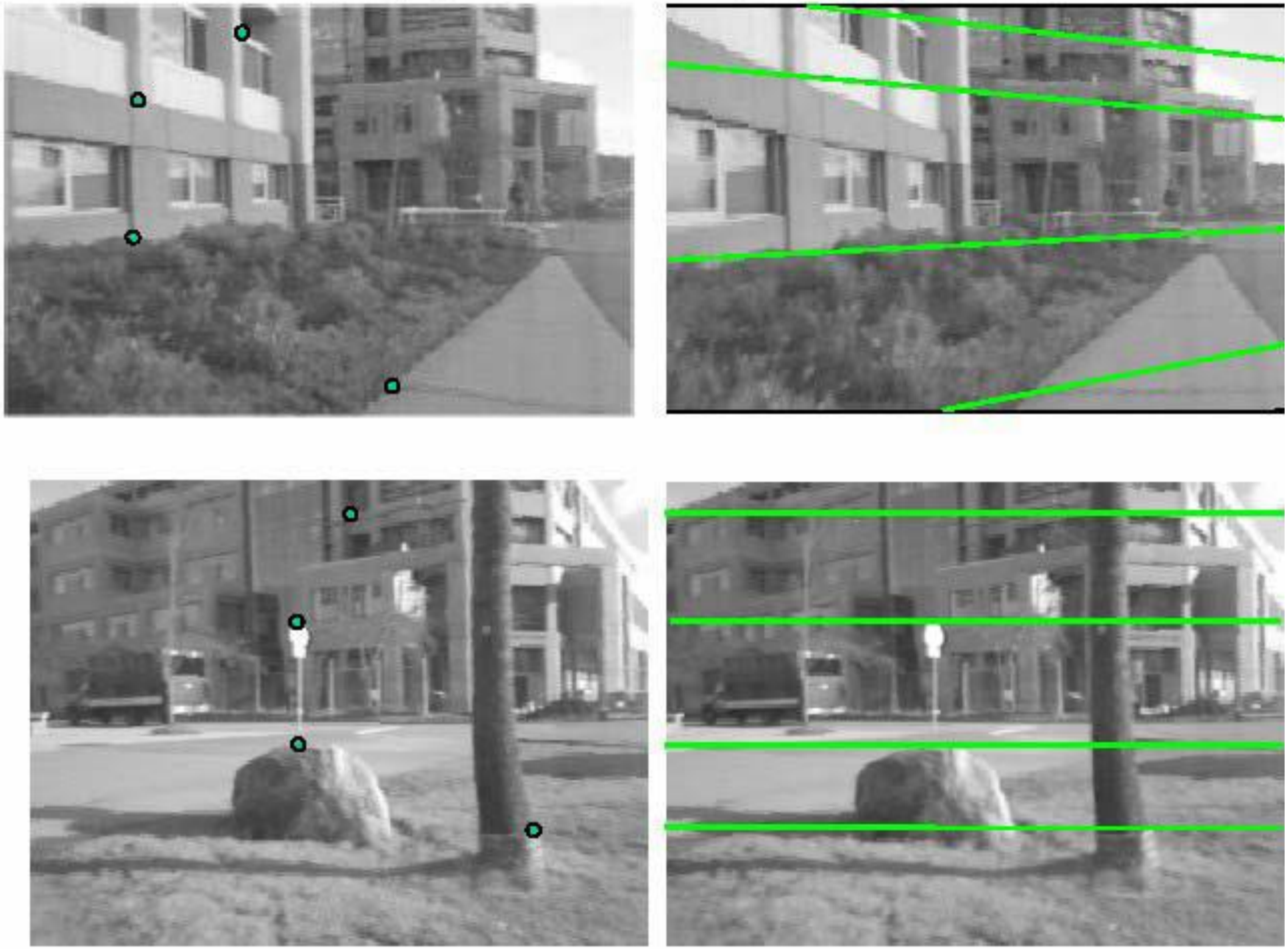
Find Epipoles:

- e_l: left image of O_r
- e_r: right image of O_l

Given p_l:

- consider its epipolar line: $p_l\ e_l$
- find epipolar plane: O_l,p_l,e_l
- intersect the epipolar plane with the right image plane
- search for p_r on the right epipolar line

Epipolar constraint example



From Geometry to Algebra

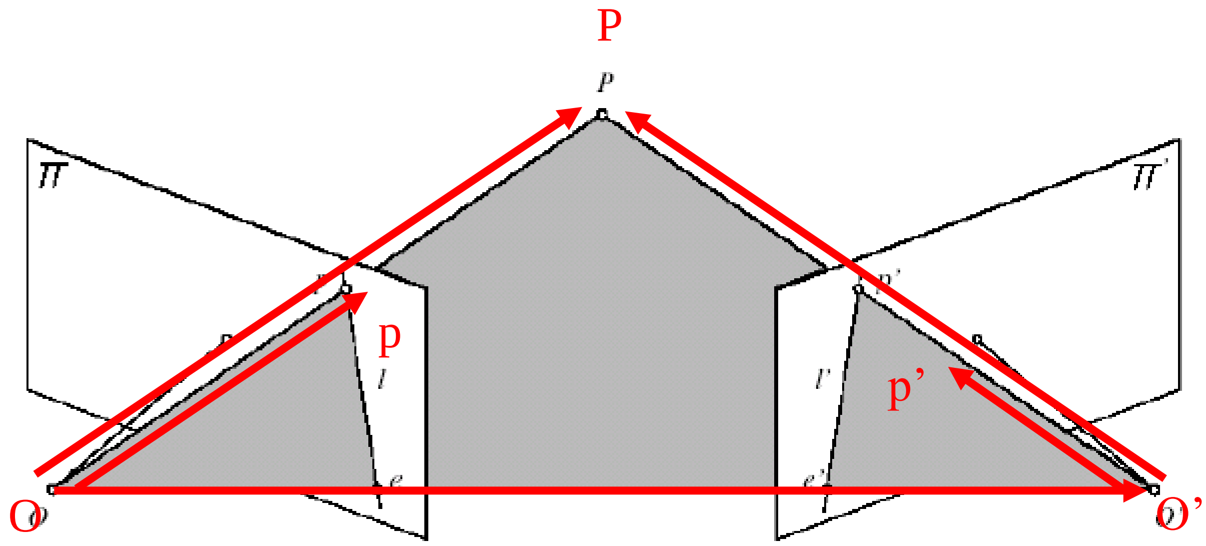
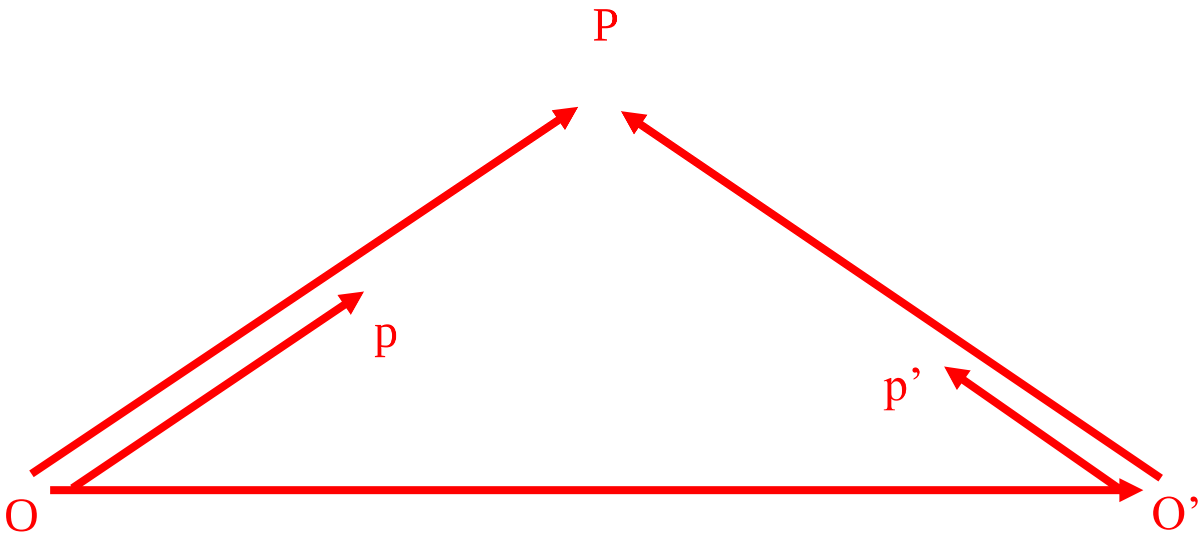


FIGURE 11.1: Epipolar geometry: the point P, the optical centers O and O' of the two cameras, and the two images p and p' of P all lie in the same plane.

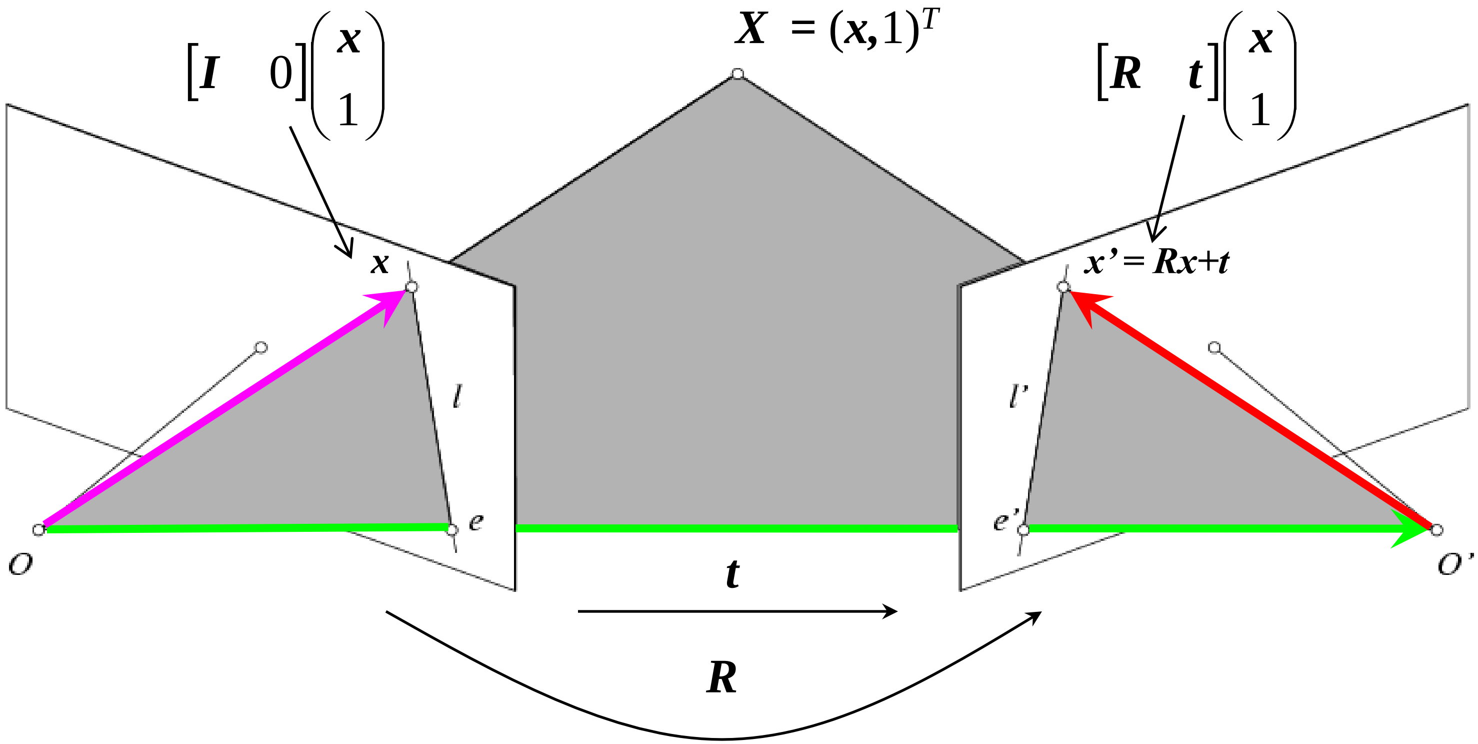
From Geometry to Algebra



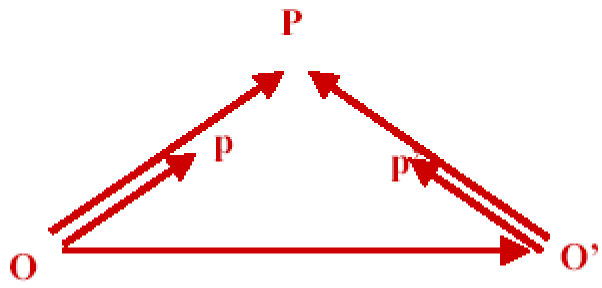
The epipolar constraint: these vectors are coplanar:

$$\overrightarrow{Op} \cdot [\overrightarrow{OO'} \times \overrightarrow{O'p'}] = 0$$

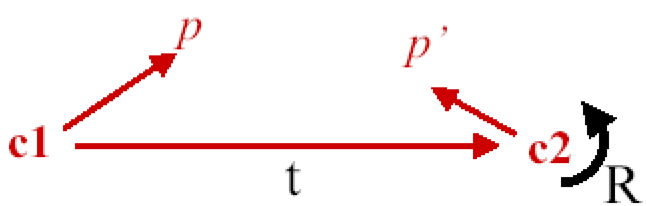
Epipolar constraint: Calibrated case



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



$$\overrightarrow{Op} \cdot [\overrightarrow{OO'} \times \overrightarrow{O'p'}] = 0$$



p,p' are image coordinates of P in c1 and c2…

c2 is related to c1 by rotation R and translation t

$$\boldsymbol{p} \cdot [\boldsymbol{t} \times (\mathcal{R}\boldsymbol{p}')] = 0$$

Linear Constraint:
Should be able to express as matrix multiplication.

Review: Matrix Form of Cross Product

The vector cross product also acts on two vectors and returns a third vector. Geometrically, this new vector is constructed such that its projection onto either of the two input vectors is zero.

$$\vec{a} \times \vec{b} = \begin{bmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{bmatrix}$$

$$\vec{a} \times \vec{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \vec{c}$$

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

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

Review: Matrix Form of Cross Product

$$\vec{a} \times \vec{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \vec{c}$$

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

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

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

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

Matrix Form

$$\boldsymbol{p} \cdot [\boldsymbol{t} \times (\mathcal{R}\boldsymbol{p}')] = 0$$

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

$$p^T [t_x] \Re p' = 0$$

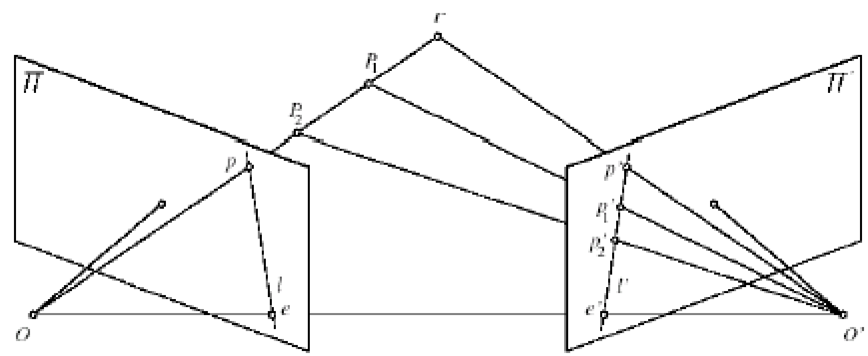
$$\varepsilon = [t_x] \Re$$

$$\boxed{\boldsymbol{p}^T \mathcal{E} \boldsymbol{p}' = 0}$$

The Essential Matrix（本征矩阵）

Matrix that relates image of point in one camera to a second camera, given translation and rotation.

$$\mathcal{E} = [t_x]\Re$$



$$\boldsymbol{p}^T \mathcal{E} \boldsymbol{p}' = 0$$

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

Longuet-Higgins, H. C. (1981). "A computer algorithm for reconstructing a scene from two projections". *Nature* **293**

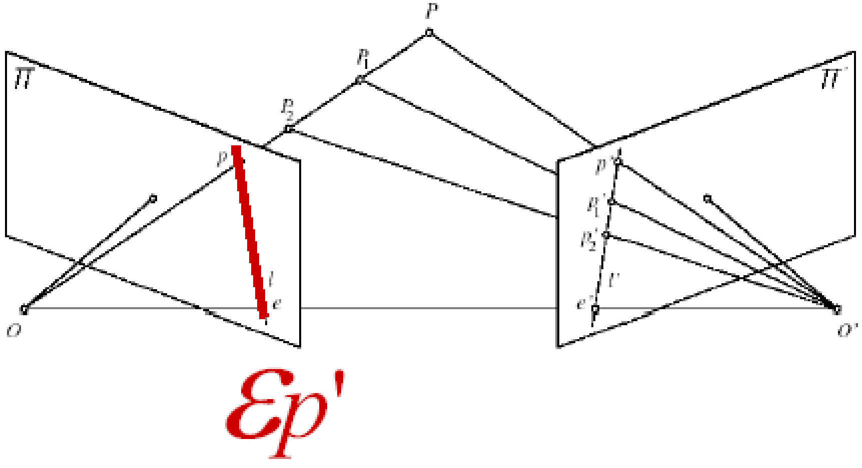
The Essential Matrix

- Based on the relative geometry of the cameras
- Assumes cameras are calibrated (i.e., intrinsic parameters are known)
- Relates image of point in one camera to a second camera (points in camera coordinate system).
- Is defined up to scale
- 5 independent parameters

The Essential Matrix

$\mathcal{E}p'$ is the epipolar line corresponding to p' in the left camera.

$$au + bv + c = 0$$



$$p = (u, v, 1)^T$$
$$l = (a, b, c)^T$$
$$l \cdot p = 0$$

$$\mathcal{E}p' \cdot p = 0$$

$$\boldsymbol{p}^T \mathcal{E} \boldsymbol{p}' = 0$$

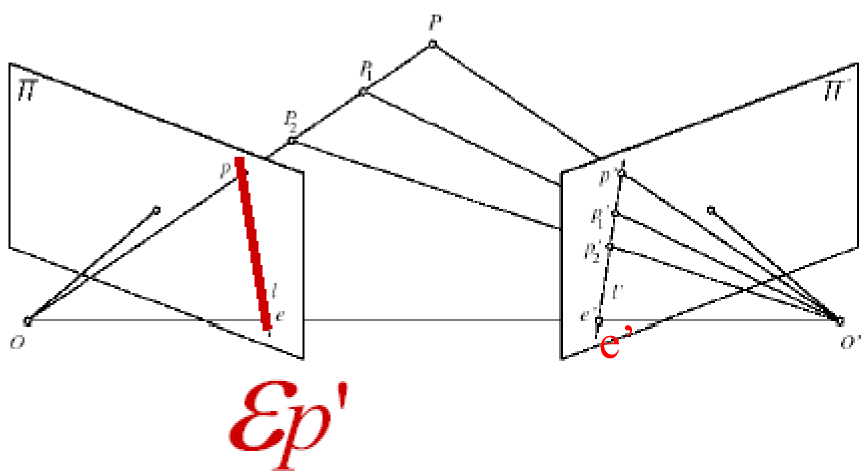
Similarly $\mathcal{E}^T p$ is the epipolar line corresponding to p in the right camera

The Essential Matrix

$$\mathcal{E} e' = [t_\times] \mathrm{R} e' = 0$$

Similarly, $$\mathcal{E}^T e = R^T [t_\times]^T e = -R^T [t_\times] e = 0$$

3 degrees of freedom of rotation matrix R
2 degrees of freedom defining the direction of the translation vector t

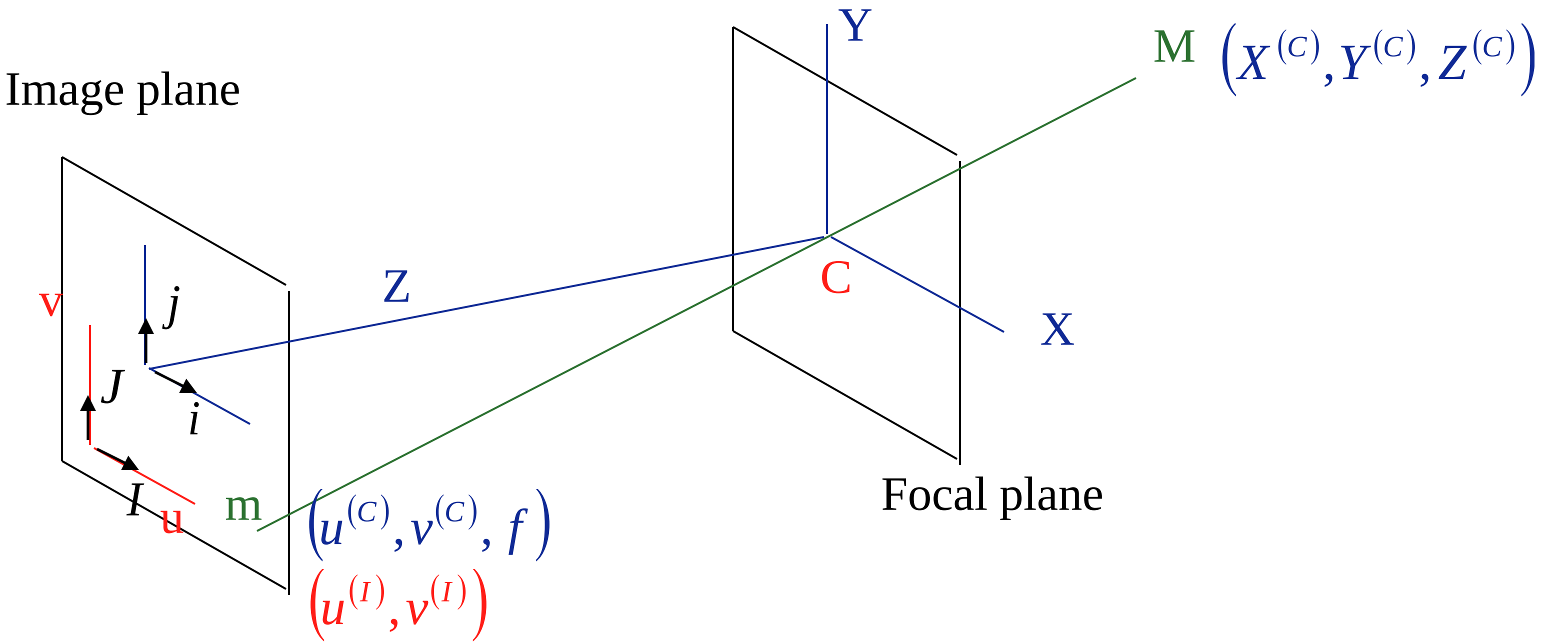


What if Camera Calibration is not known

Review: Intrinsic Camera Parameters

$$i = k_u I$$

$$j = k_v J$$



$$\begin{bmatrix} U^{(new)} \\ V^{(new)} \\ S \end{bmatrix} = \underbrace{\begin{bmatrix} -f_u & 0 & u_0 & 0 \\ 0 & -f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{K} \begin{bmatrix} X^{(C)} \\ Y^{(C)} \\ Z^{(C)} \\ 1 \end{bmatrix}$$

$$f_u = f / s_u$$

$$f_v = f / s_v$$

Fundamental Matrix （基本矩阵）

$p^T \mathcal{E} p' = 0$ $\quad p$ and p' are in camera coordinate system

If u and u' are corresponding image coordinates then we have

$$u = K_1 p \qquad u' = K_2 p' \quad \Longrightarrow \quad p = K_1^{-1} u \qquad p' = K_2^{-1} u'$$

$$u^T K_1^{-T} \mathcal{E} K_2^{-1} u' = 0$$

$$\boxed{\Rightarrow u^T F u' = 0}$$

$$F = K_1^{-T} \mathcal{E} K_2^{-1}$$

Fundamental Matrix

$$u^T F u' = 0$$

$$F = K_1^{-T} \mathcal{E} K_2^{-1}$$

Fundamental Matrix is singular with rank 2

In principal F has 7 parameters up to scale and can be estimated from 7 point correspondences

Direct Simpler Method requires 8 correspondences

Estimating Fundamental Matrix

$$u^T F u' = 0$$

The 8-point algorithm

Each point correspondence can be expressed as a linear equation

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

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

The eight-point algorithm

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

$$(u, v, 1)\begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix}\begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0$$

$$(uu', uv', u, vu', vv', v, u', v', 1)\begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{pmatrix} = 0$$

$$\begin{pmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' \\ u_2u_2' & u_2v_2' & u_2 & v_2u_2' & v_2v_2' & v_2 & u_2' & v_2' \\ u_3u_3' & u_3v_3' & u_3 & v_3u_3' & v_3v_3' & v_3 & u_3' & v_3' \\ u_4u_4' & u_4v_4' & u_4 & v_4u_4' & v_4v_4' & v_4 & u_4' & v_4' \\ u_5u_5' & u_5v_5' & u_5 & v_5u_5' & v_5v_5' & v_5 & u_5' & v_5' \\ u_6u_6' & u_6v_6' & u_6 & v_6u_6' & v_6v_6' & v_6 & u_6' & v_6' \\ u_7u_7' & u_7v_7' & u_7 & v_7u_7' & v_7v_7' & v_7 & u_7' & v_7' \\ u_8u_8' & u_8v_8' & u_8 & v_8u_8' & v_8v_8' & v_8 & u_8' & v_8' \end{pmatrix}\begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \end{pmatrix} = -\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

Minimize:

$$\sum_{i=1}^{N}(x_i^T F x_i')^2$$

under the constraint

$F_{33} = 1$

The eight-point algorithm

- **Meaning of error** $\sum_{i=1}^{N}(x_i^T F x_i')^2$:

  sum of Euclidean distances between points x_i and epipolar lines Fx'_i (or points x'_i and epipolar lines $F^T x_i$) multiplied by a scale factor

- **Nonlinear approach: minimize**

$$\sum_{i=1}^{N}\left[\mathrm{d}^2(x_i, F x_i') + \mathrm{d}^2(x_i', F^T x_i)\right]$$

Problem with eight-point algorithm

$$\begin{pmatrix} u_1u'_1 & u_1v'_1 & u_1 & v_1u'_1 & v_1v'_1 & v_1 & u'_1 & v'_1 \\ u_2u'_2 & u_2v'_2 & u_2 & v_2u'_2 & v_2v'_2 & v_2 & u'_2 & v'_2 \\ u_3u'_3 & u_3v'_3 & u_3 & v_3u'_3 & v_3v'_3 & v_3 & u'_3 & v'_3 \\ u_4u'_4 & u_4v'_4 & u_4 & v_4u'_4 & v_4v'_4 & v_4 & u'_4 & v'_4 \\ u_5u'_5 & u_5v'_5 & u_5 & v_5u'_5 & v_5v'_5 & v_5 & u'_5 & v'_5 \\ u_6u'_6 & u_6v'_6 & u_6 & v_6u'_6 & v_6v'_6 & v_6 & u'_6 & v'_6 \\ u_7u'_7 & u_7v'_7 & u_7 & v_7u'_7 & v_7v'_7 & v_7 & u'_7 & v'_7 \\ u_8u'_8 & u_8v'_8 & u_8 & v_8u'_8 & v_8v'_8 & v_8 & u'_8 & v'_8 \end{pmatrix} \begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \end{pmatrix} = - \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

Problem with eight-point algorithm

$$
\begin{bmatrix}
250906.36 & 183269.57 & 921.81 & 200931.10 & 146766.13 & 738.21 & 272.19 & 198.81 \\
2692.28 & 131633.03 & 176.27 & 6196.73 & 302975.59 & 405.71 & 15.27 & 746.79 \\
416374.23 & 871684.30 & 935.47 & 408110.89 & 854384.92 & 916.90 & 445.10 & 931.81 \\
191183.60 & 171759.40 & 410.27 & 416435.62 & 374125.90 & 893.65 & 465.99 & 418.65 \\
48988.86 & 30401.76 & 57.89 & 298604.57 & 185309.58 & 352.87 & 846.22 & 525.15 \\
164786.04 & 546559.67 & 813.17 & 1998.37 & 6628.15 & 9.86 & 202.65 & 672.14 \\
116407.01 & 2727.75 & 138.89 & 169941.27 & 3982.21 & 202.77 & 838.12 & 19.64 \\
135384.58 & 75411.13 & 198.72 & 411350.03 & 229127.78 & 603.79 & 681.28 & 379.48
\end{bmatrix}
\begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \end{pmatrix}
= -
\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}
$$

- **Poor numerical conditioning**
- **Can be fixed by rescaling the data**

The normalized eight-point algorithm

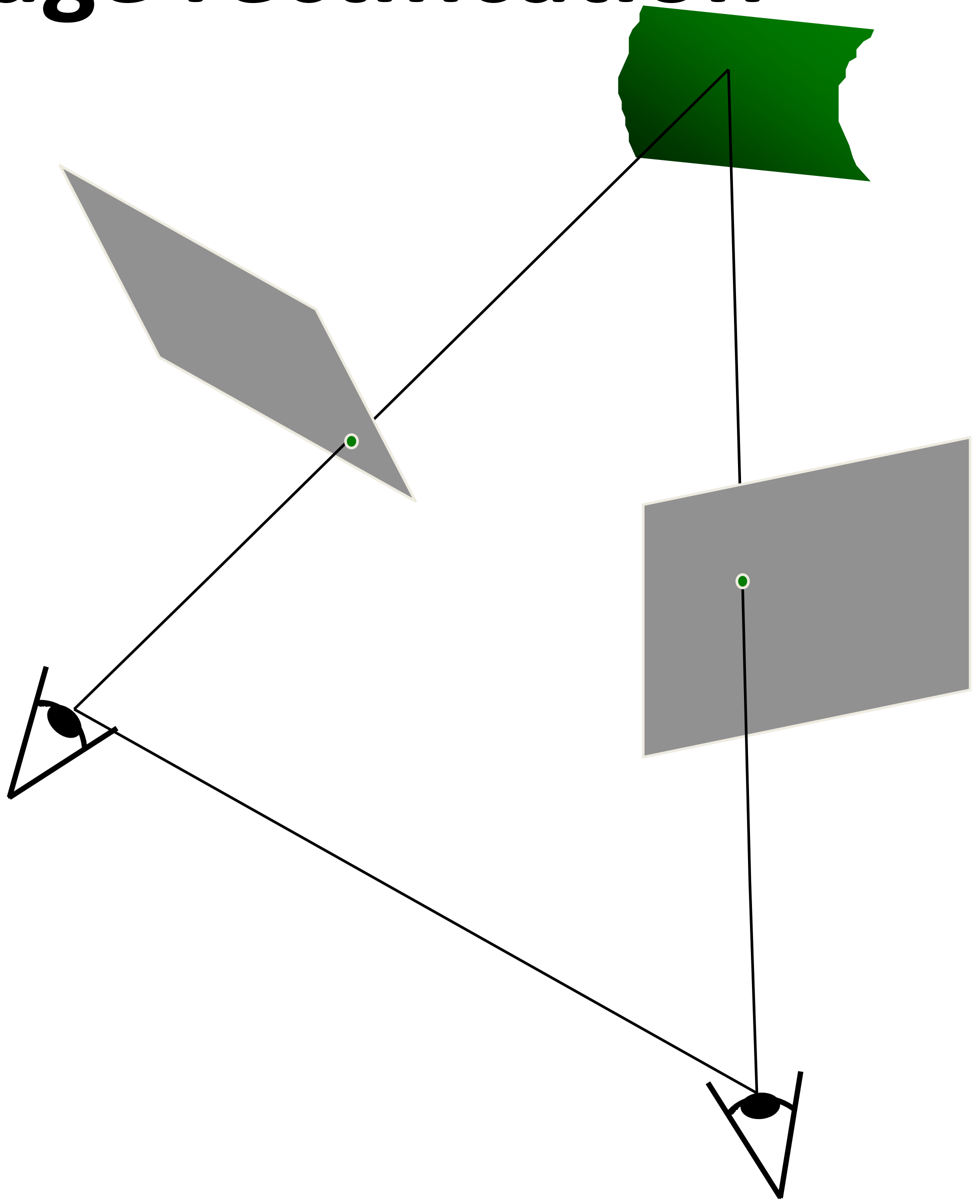
(Hartley, 1995)

- **Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels**
- **Use the eight-point algorithm to compute *F* from the normalized points**
- **Enforce the rank-2 constraint (for example, take SVD of *F* and throw out the smallest singular value)**
- **Transform fundamental matrix back to original units: if T and T' are the normalizing transformations in the two images, than the fundamental matrix in original coordinates is $T^T F T'$**

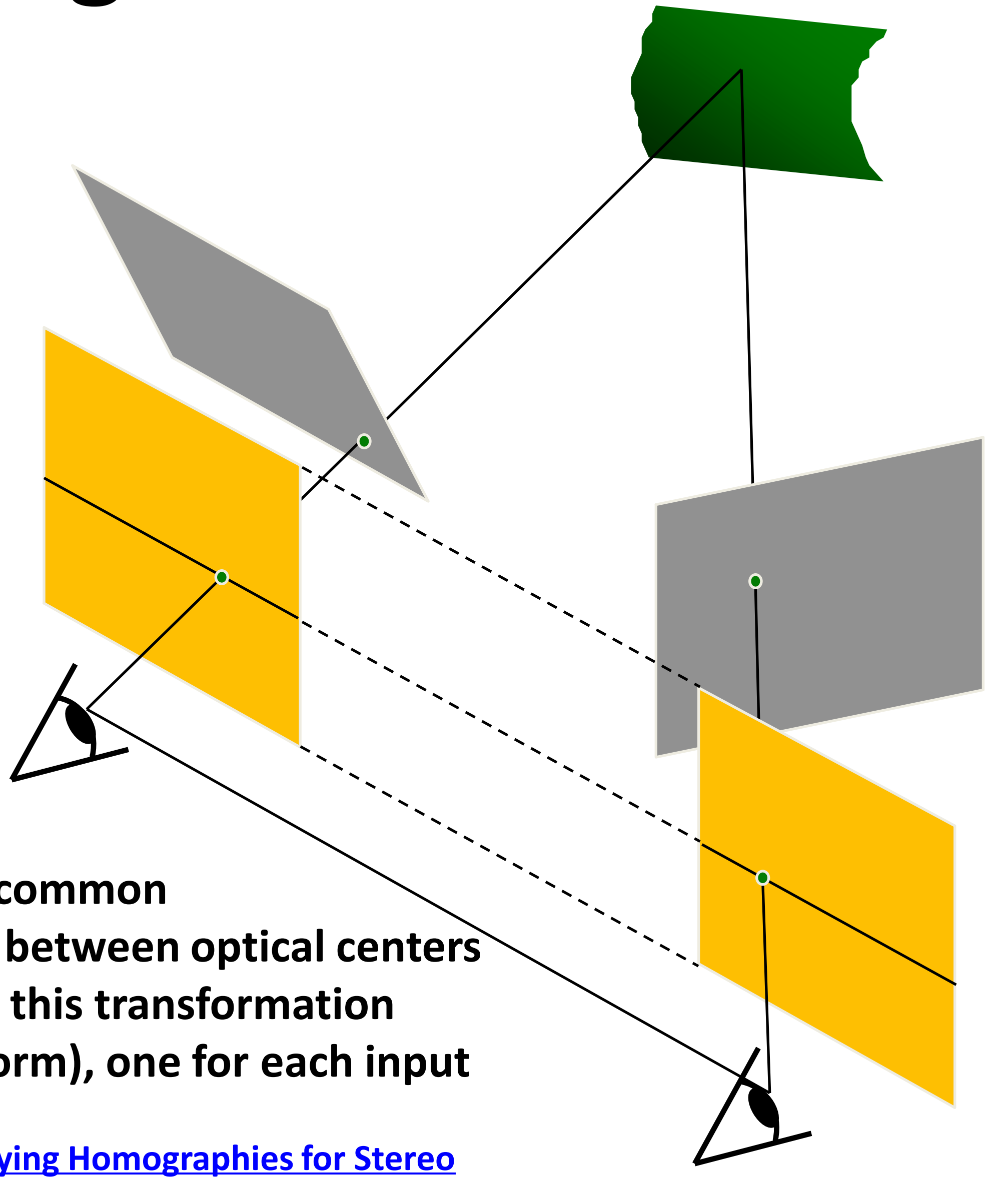
Comparison of estimation

	8-point	Normalized 8-point	Nonlinear least squares
Av. Dist. 1	2.33 pixels	0.92 pixel	0.86 pixel
Av. Dist. 2	2.18 pixels	0.85 pixel	0.80 pixel

Stereo image rectification



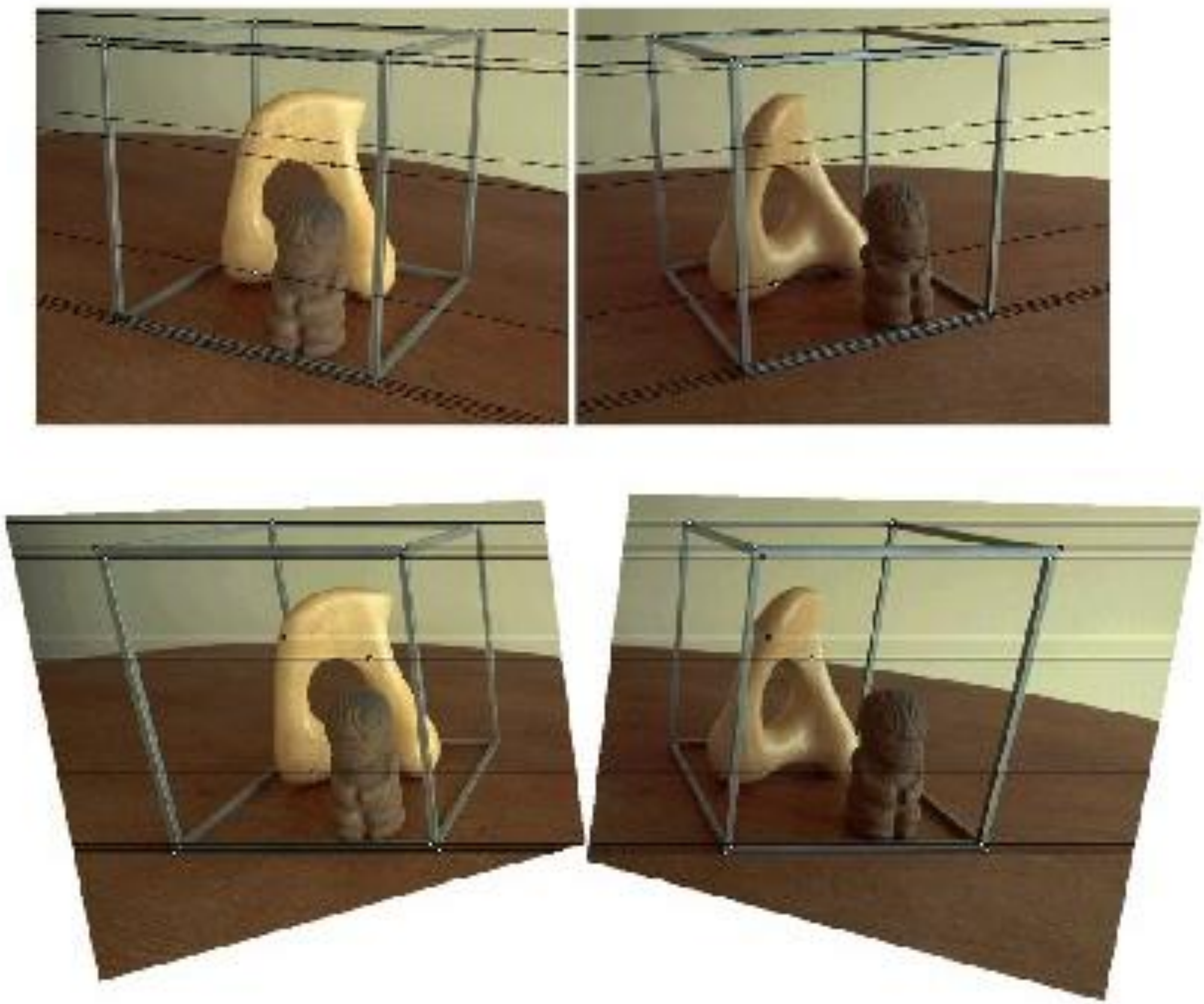
Stereo image rectification



- reproject image planes onto a common
- plane parallel to the line between optical centers
- pixel motion is horizontal after this transformation
- two homographies (3x3 transform), one for each input image reprojection

➢ C. Loop and Z. Zhang. [Computing Rectifying Homographies for Stereo Vision](). IEEE Conf. Computer Vision and Pattern Recognition, 1999.

Rectification example



Stereo correspondence algorithms

Problem statement

<u>Given</u>: two images and their associated cameras compute corresponding image points.

The methods may be top down or bottom up

Correspondence algorithms

Algorithms may be top down or bottom up – random dot stereograms are an existence proof that bottom up algorithms are possible

From here on only consider bottom up algorithms

Algorithms may be classified into two types:

1. Dense: compute a correspondence at every pixel
2. Sparse: compute correspondences only for features

Dense correspondence algorithm

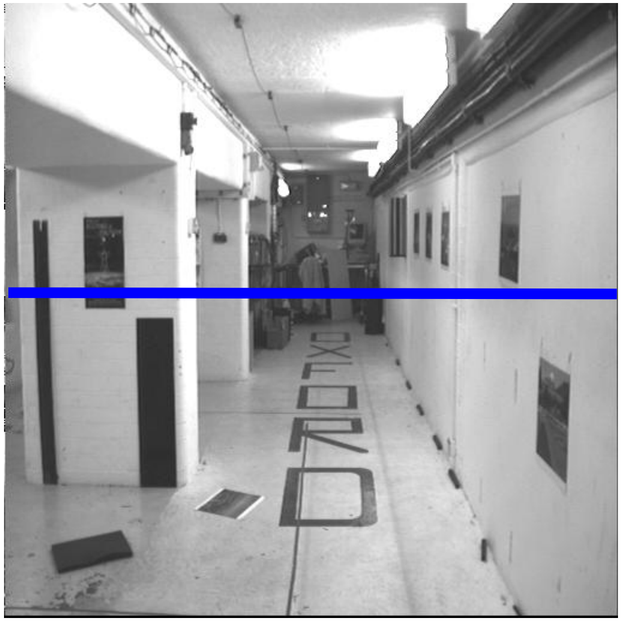
Parallel camera example – epipolar lines are corresponding rasters

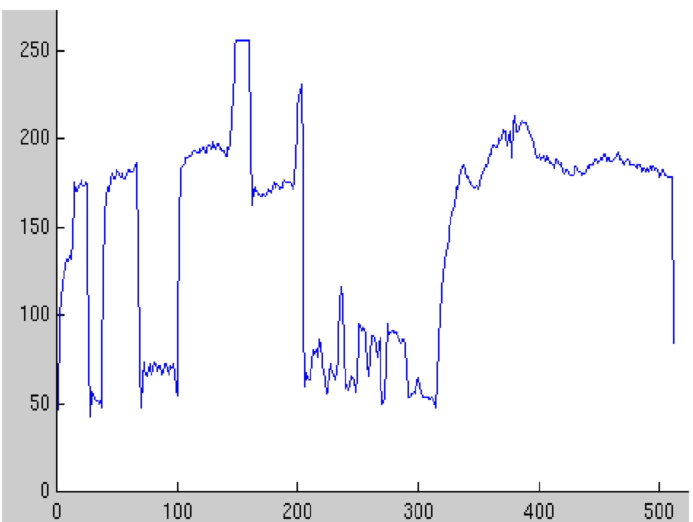
Search problem (geometric constraint): for each point in the left image, the corresponding point in the right image lies on the epipolar line (1D ambiguity)

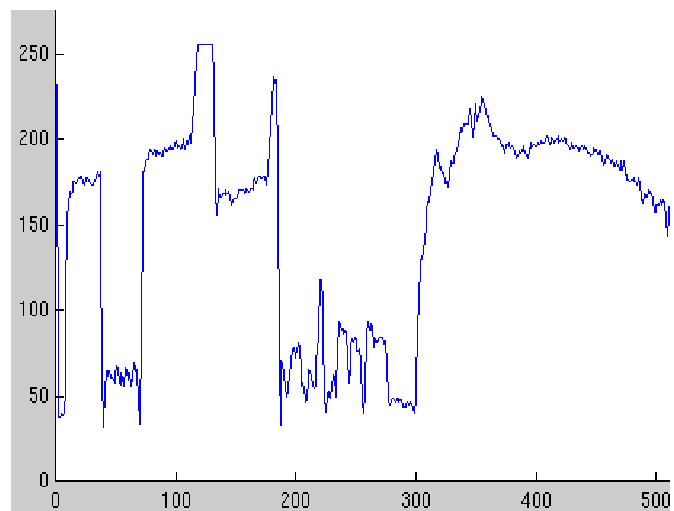
Disambiguating assumption (photometric constraint): the intensity neighbourhood of corresponding points are similar across images

Measure similarity of neighbourhood intensity by SSD or cross-correlation

Intensity profiles







- Clear correspondence between intensities, but also noise and ambiguity

Normalized Cross Correlation

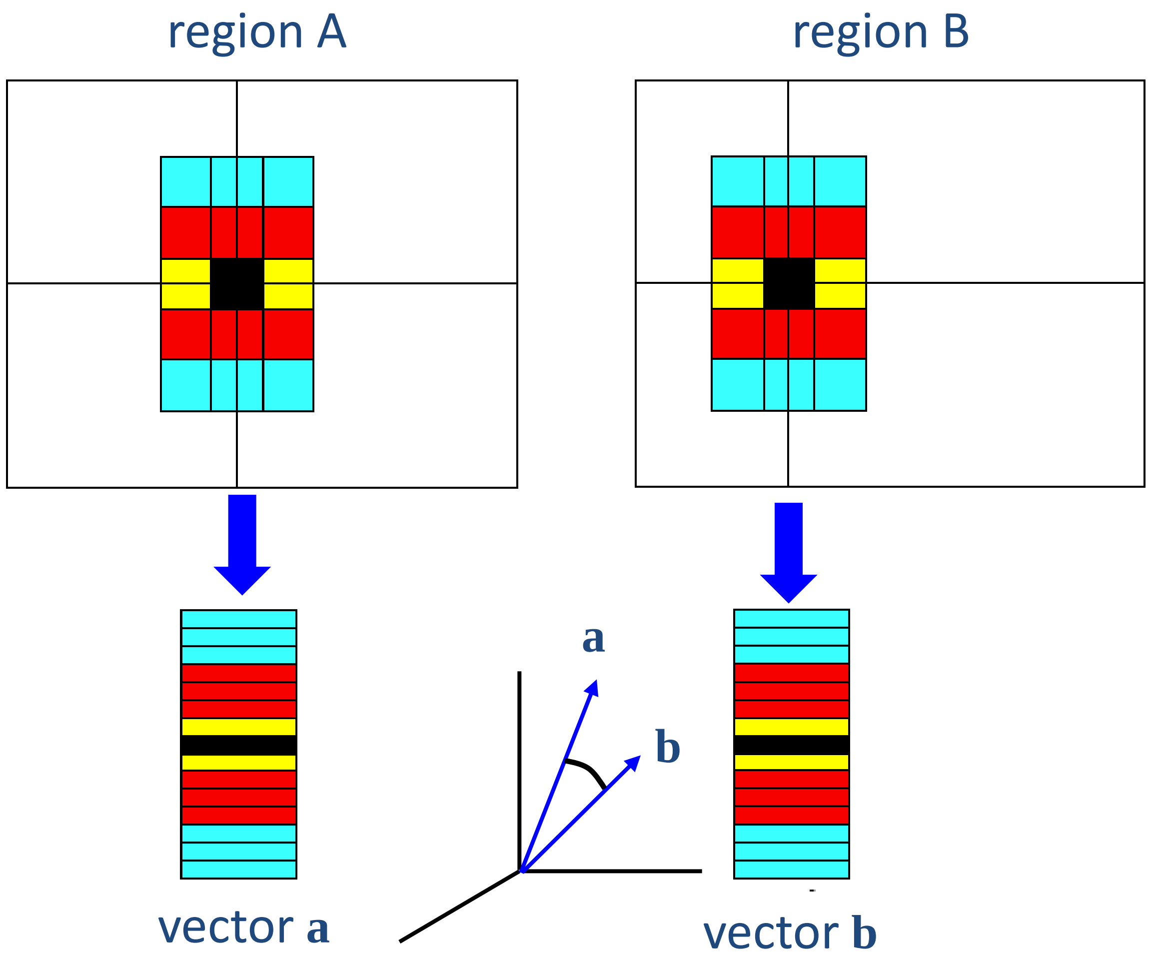
$$\mathrm{NCC} = \frac{\sum_i \sum_j A(i,j)B(i,j)}{\sqrt{\sum_i \sum_j A(i,j)^2}\sqrt{\sum_i \sum_j B(i,j)^2}}$$

write regions as vectors

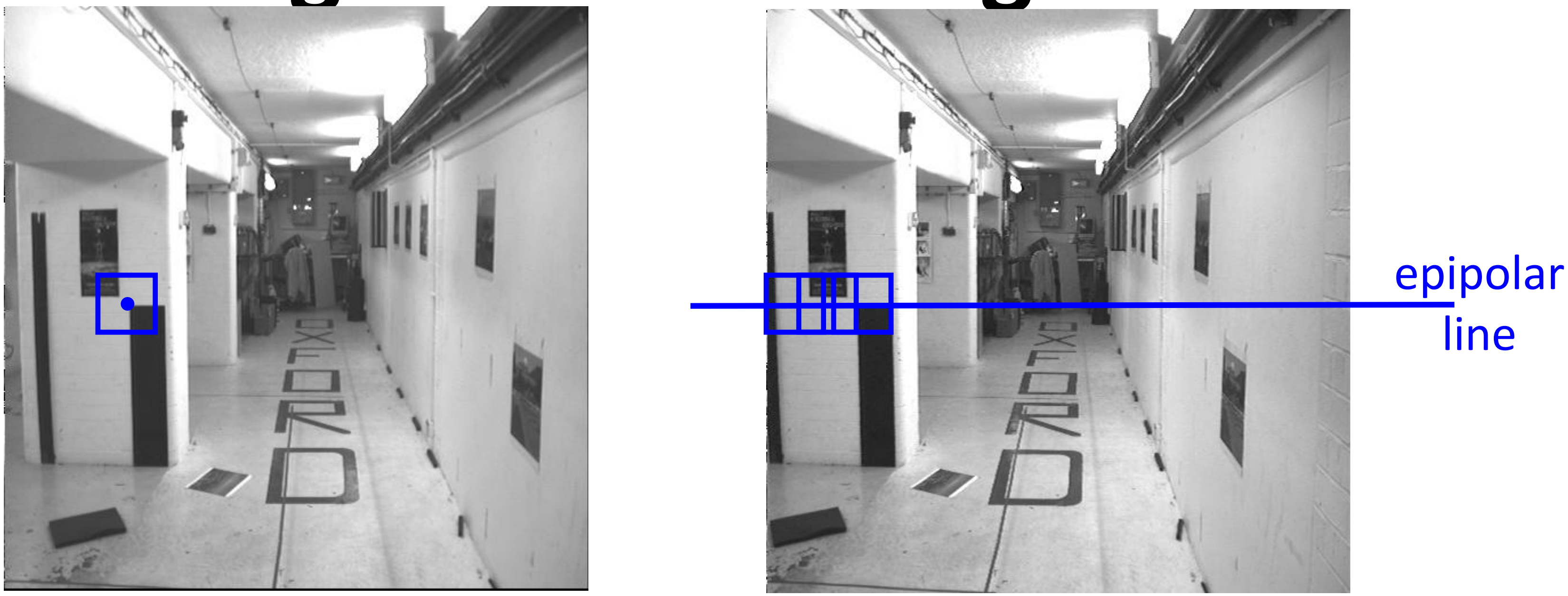
$$\mathtt{A} \rightarrow \mathbf{a}, \ \ \mathtt{B} \rightarrow \mathbf{b}$$

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

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



Cross-correlation of neighbourhood regions



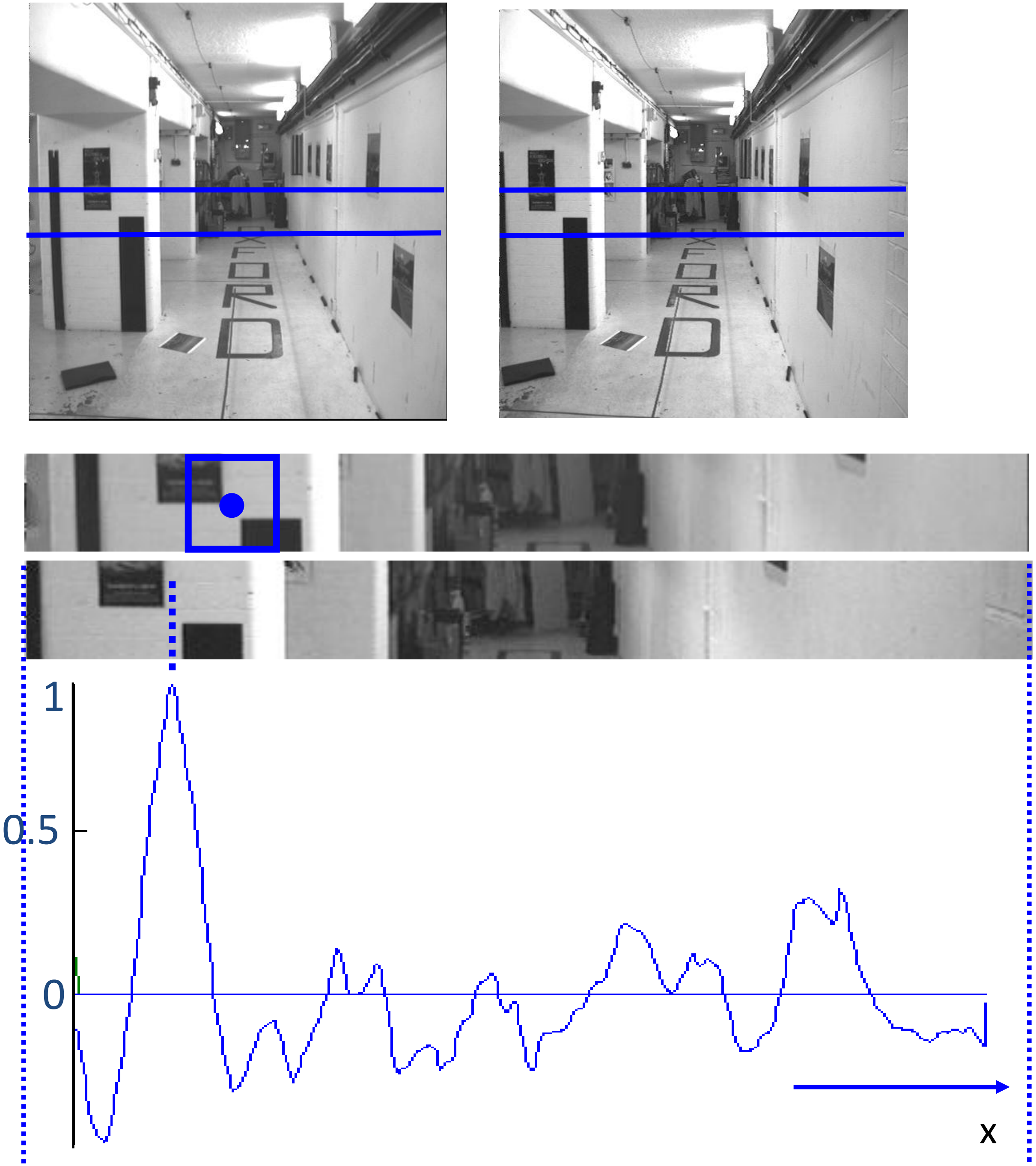
regions A, B, write as vectors $\mathbf{a}, \mathbf{b}$

translate so that mean is zero

$$\mathtt{a} \rightarrow \mathtt{a} - \langle \mathbf{a} \rangle, \quad \mathtt{b} \rightarrow \mathtt{b} - \langle \mathbf{b} \rangle$$

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

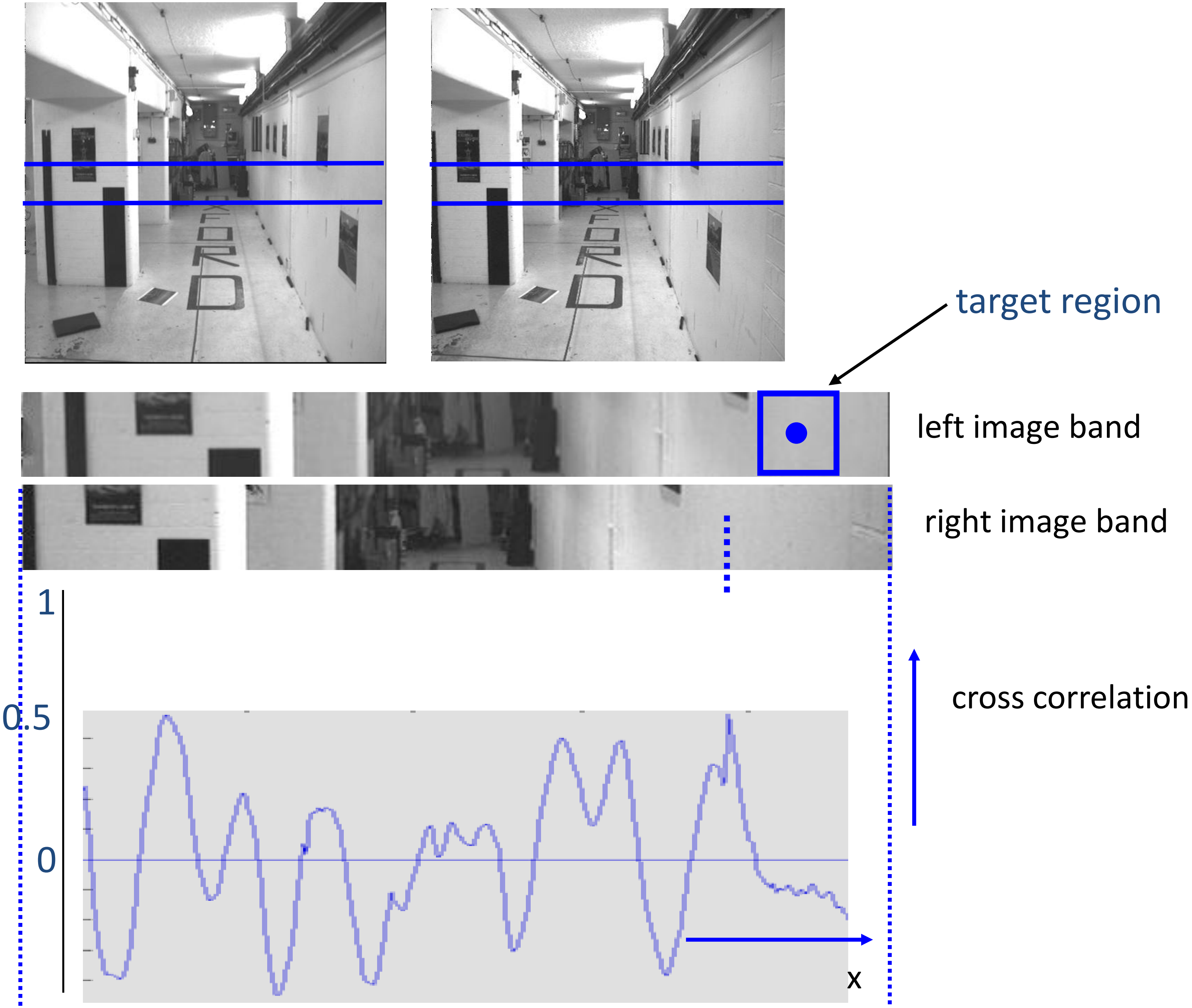
Invariant to $I \rightarrow \alpha I + \beta$



left image band

right image band

cross correlation

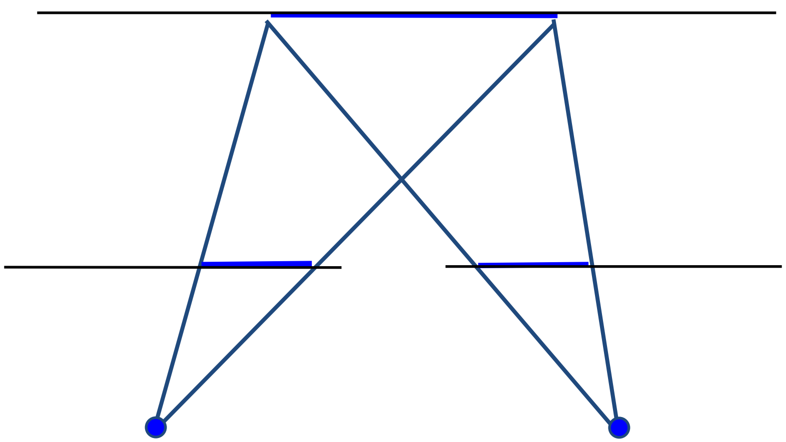


target region
left image band
right image band
1
0.5
0
cross correlation
x

Why is cross-correlation such a poor measure in the second case?

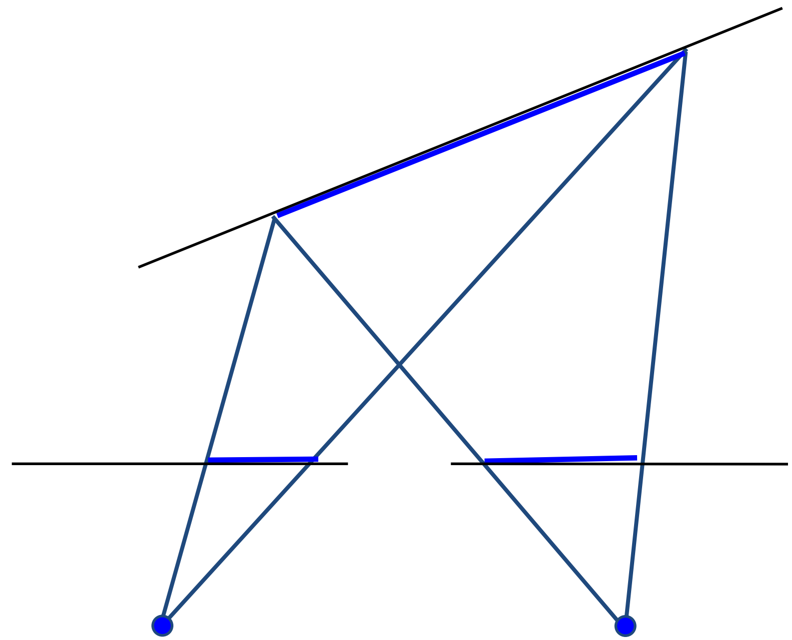
1. The neighbourhood region does not have a "distinctive" spatial intensity distribution

2. Foreshortening effects



fronto-parallel surface

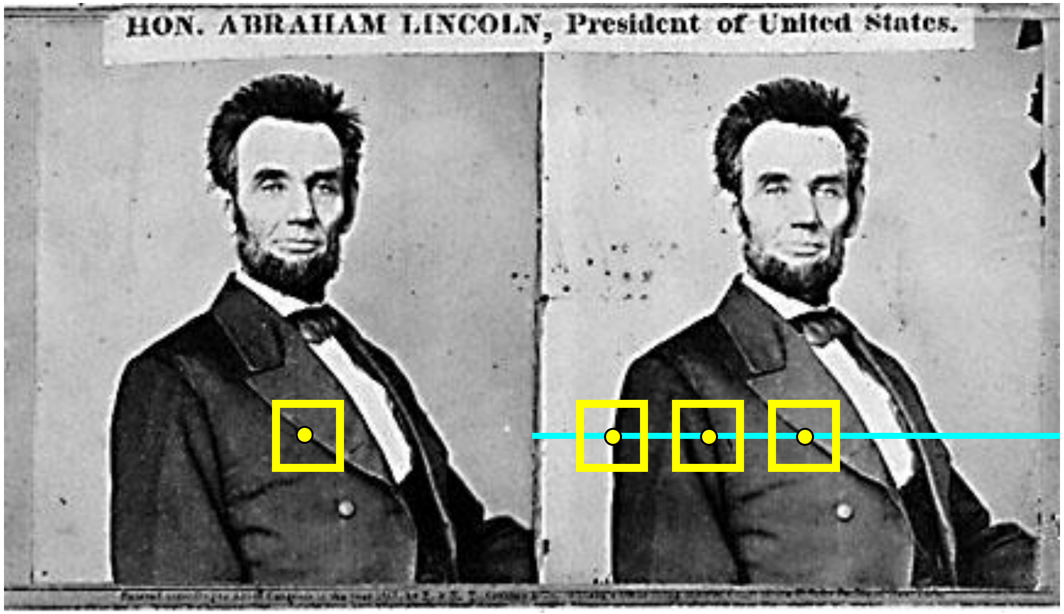
imaged length the same



slanting surface

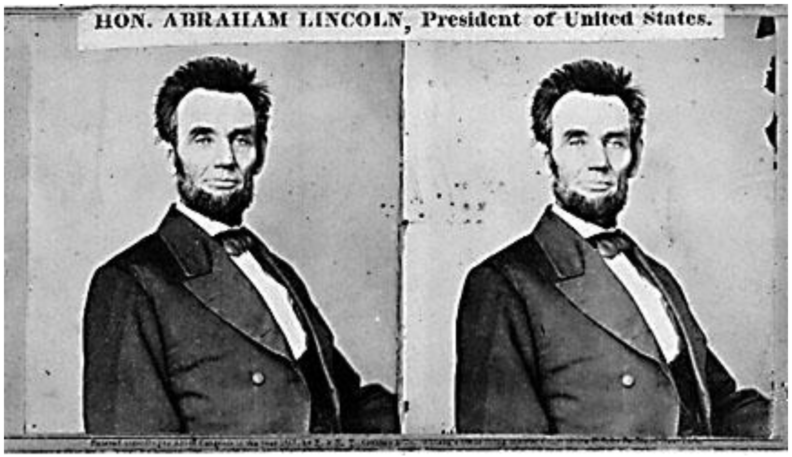
imaged lengths differ

Basic stereo matching algorithm



- If necessary, rectify the two stereo images to transform epipolar lines into scanlines
- For each pixel x in the first image
  - Find corresponding epipolar scanline in the right image
  - Examine all pixels on the scanline and pick the best match x’
  - Compute disparity x-x’ and set depth(x) = B*f/(x-x’)

Limitations of similarity constraint



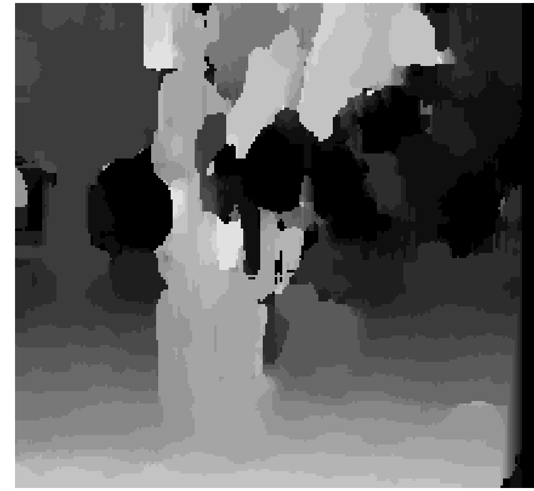
Textureless surfaces

Occlusions, repetition

Non-Lambertian surfaces, specularities

Effect of window size

W = 3

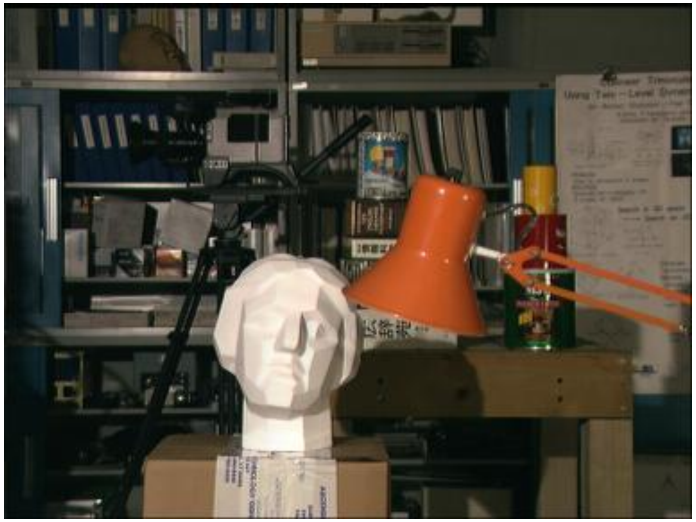


W = 20

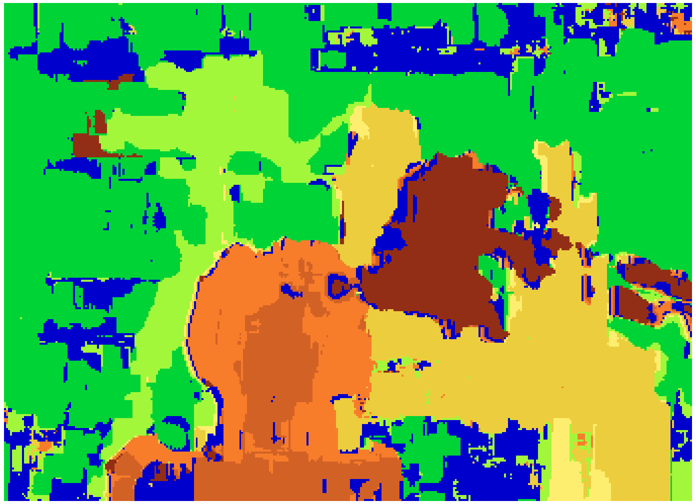
- Smaller window
  - + More detail
  - • More noise
- Larger window
  - + Smoother disparity maps
  - • Less detail

Results with window search

Data



Window-based matching



Ground truth

Sketch of a dense correspondence algorithm

For each pixel in the left image

- compute the neighbourhood cross correlation along the corresponding epipolar line in the right image
- the corresponding pixel is the one with the highest cross correlation

Parameters

- size (scale) of neighbourhood
- search disparity

Other constraints

- uniqueness
- ordering
- smoothness of disparity field

Applicability

- textured scene, largely fronto-parallel

Feature-based Methods

- Conceptually very similar to Correlation-based methods, but:
  - They only search for correspondences of a sparse set of image features.
  - Correspondences are given by the most similar feature pairs.
  - Similarity measure must be adapted to the type of feature used.

Feature-based Methods

- Constraints on feasible matches:
  - Geometric, like the epipolar constraint.
  - Analytical:
    - Compatibility
    - Uniqueness
    - Continuity

Feature-based Methods:

- Features most commonly used:
  - Corners
    - Similarity measured in terms of:
      - surrounding gray values (SSD, Cross-correlation)
      - location
  - Edges, Lines
    - Similarity measured in terms of:
      - orientation
      - contrast
      - coordinates of edge or line's midpoint
      - length of line

Example: Comparing lines

- l_l and l_r: line lengths
- θ_l and θ_r: line orientations
- (x_l,y_l) and (x_r,y_r): midpoints
- c_l and c_r: average contrast along lines
- ω_l ω_θ ω_m ω_c : weights controlling influence

$$S = \frac{1}{\omega_l(l_l - l_r)^2 + \omega_\theta(\theta_l - \theta_r)^2 + \omega_m[(x_l - x_r)^2 + (y_l - y_r)^2] + \omega_c(c_l - c_r)^2}$$

The more similar the lines, the larger S is!

FEATURE_MATCHING Algorithm

- **Inputs:**
  - I_l and I_r
  - Set of features on the left and right
- **Things that must be chosen:**
  - Search Window
  - Similarity measure

FEATURE_MATCHING Algorithm

- **For each feature f_l in the left image:**
  - **Compute the similarity measure between f_l and every feature in the search window $R(f_l)$**
  - **Select the feature in $R(f_l)$ that maximizes the similarity measure.**
  - **Save the correspondence and the disparity of f_l**
- **Output the list of correspondences and disparities.**

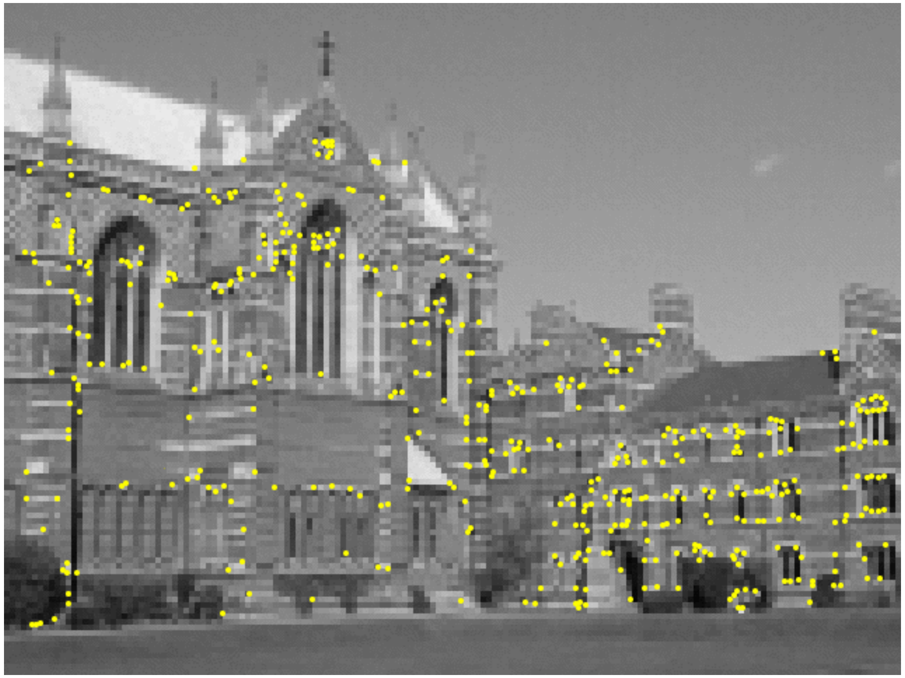
Example

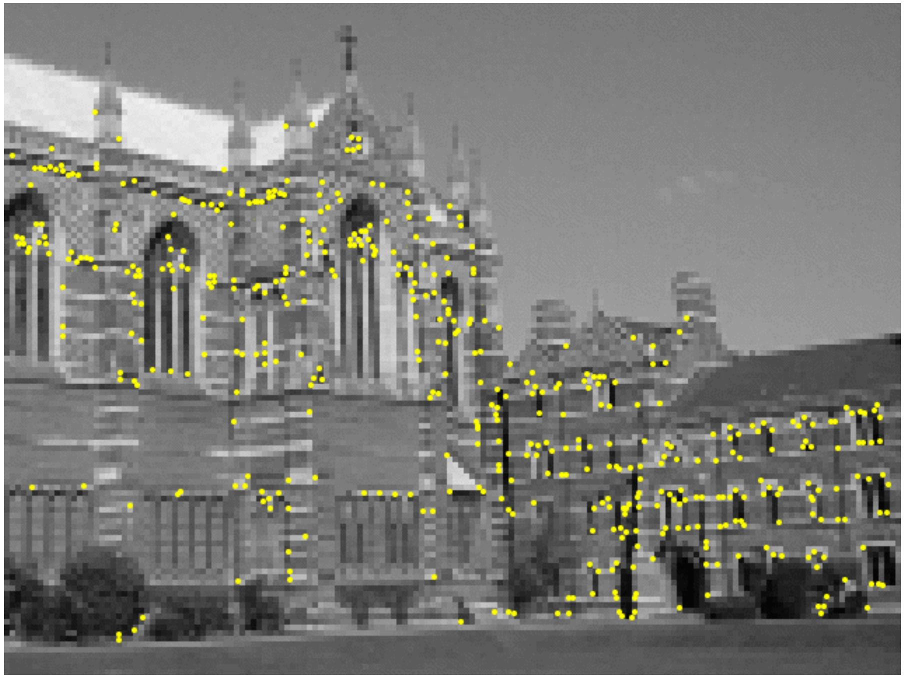
- Find the correspondences

from Hartley & Zisserman

Example

- Apply corner detectors to both images

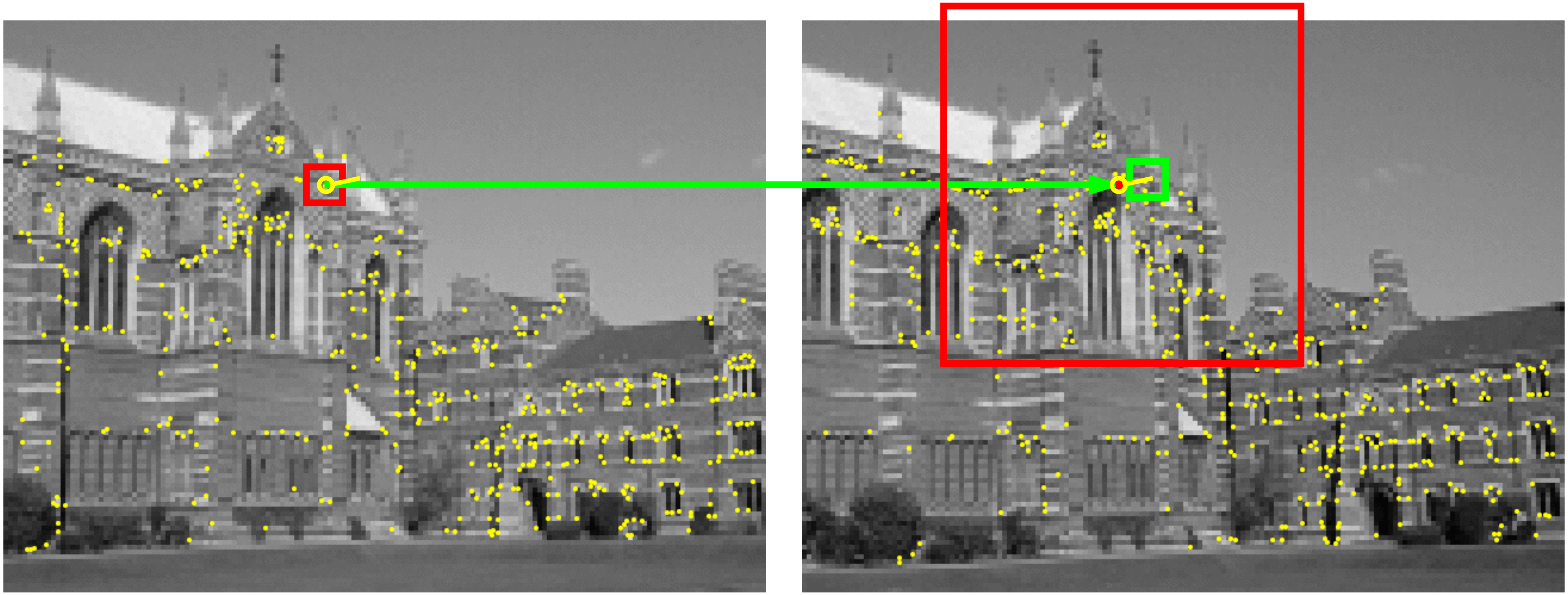




from Hartley & Zisserman

Example

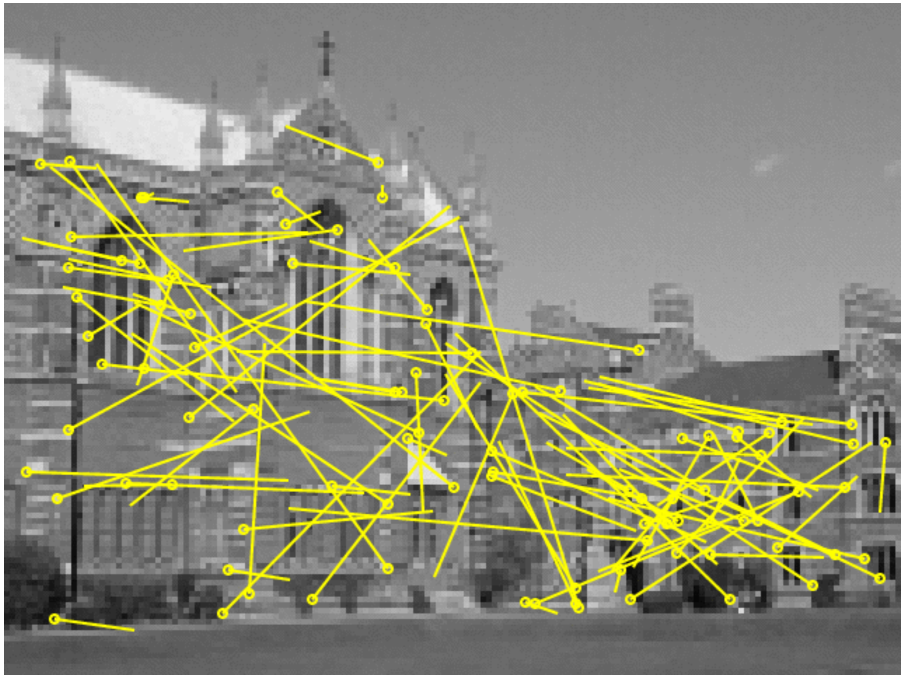
- Find the best match within a search window.



from Hartley & Zisserman

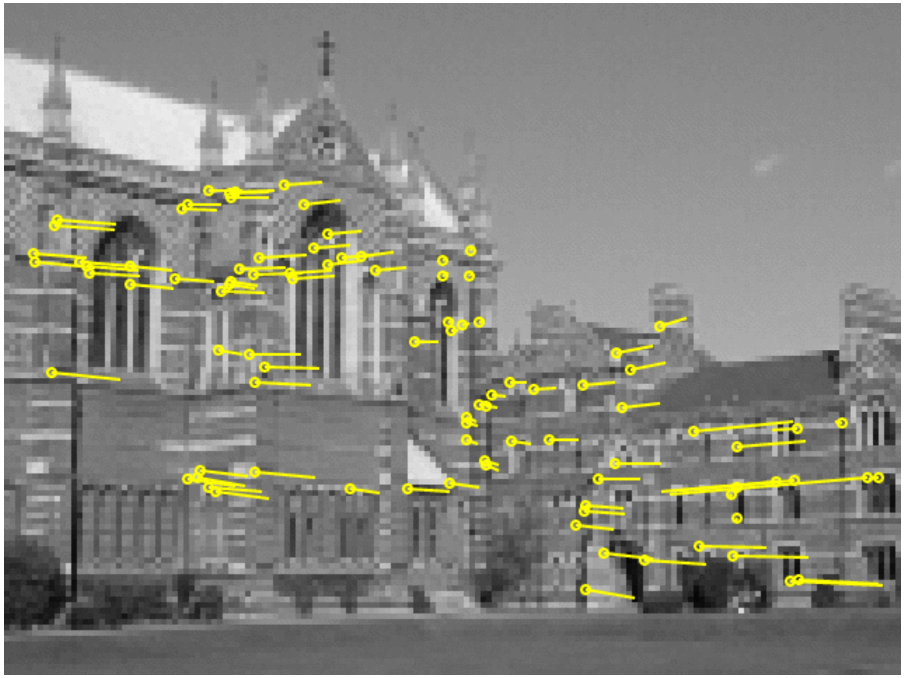
Example

- Inliers and outliers



from Hartley & Zisserman

89 outliers



99 inliers

Which method should we use?

- Correlation methods:
  - dense maps, good for surface reconstruction
  - Require textured images
  - Sensitive to illumination variations
  - Inadequate for very different viewpoints
- Feature methods:
  - Sparse maps, good for navigation
  - Require prior knowledge of type of scene
  - Must find features first

Better methods exist...

Graph cuts

Ground truth

Y. Boykov, O. Veksler, and R. Zabih, [Fast Approximate Energy Minimization via Graph Cuts](), PAMI 2001

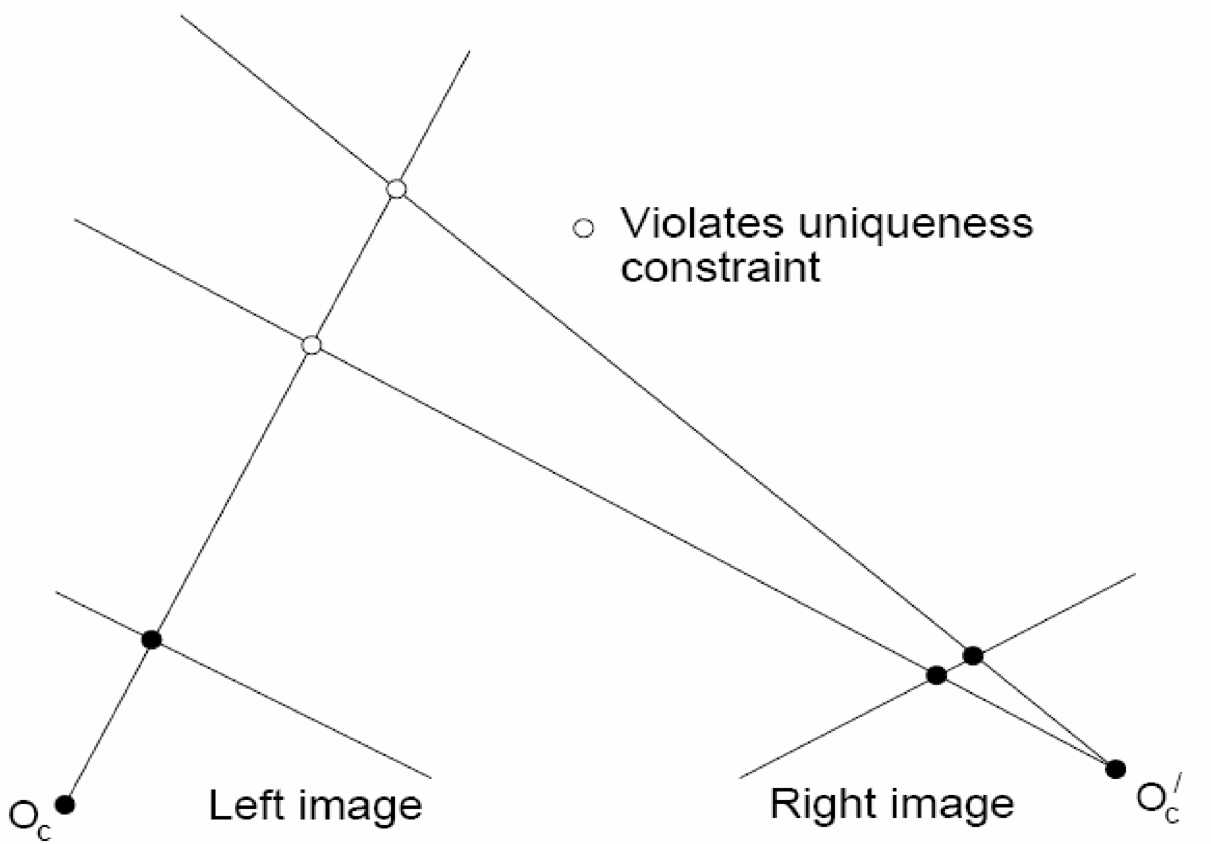
For the latest and greatest: [http://www.middlebury.edu/stereo/]()

How can we improve window-based matching?

- The similarity constraint is local (each reference window is matched independently)
- Need to enforce non-local correspondence constraints

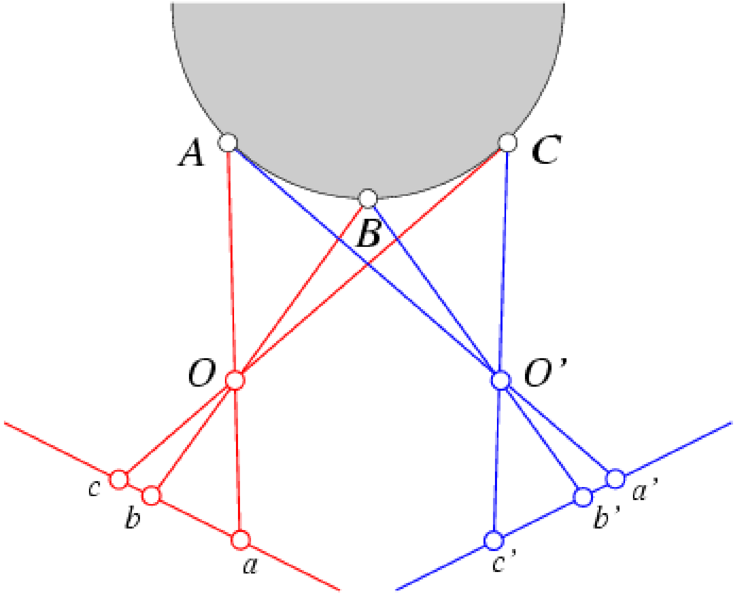
Non-local constraints

- **Uniqueness**
  - **For any point in one image, there should be at most one matching point in the other image**



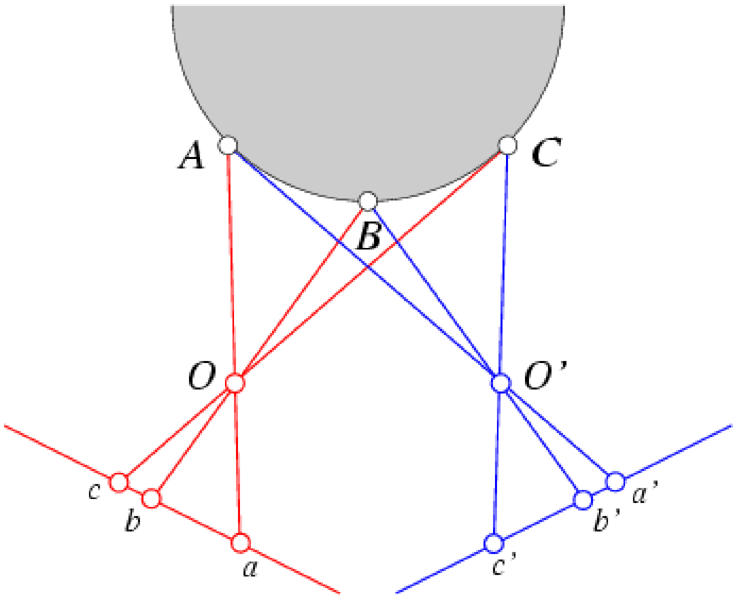
Non-local constraints

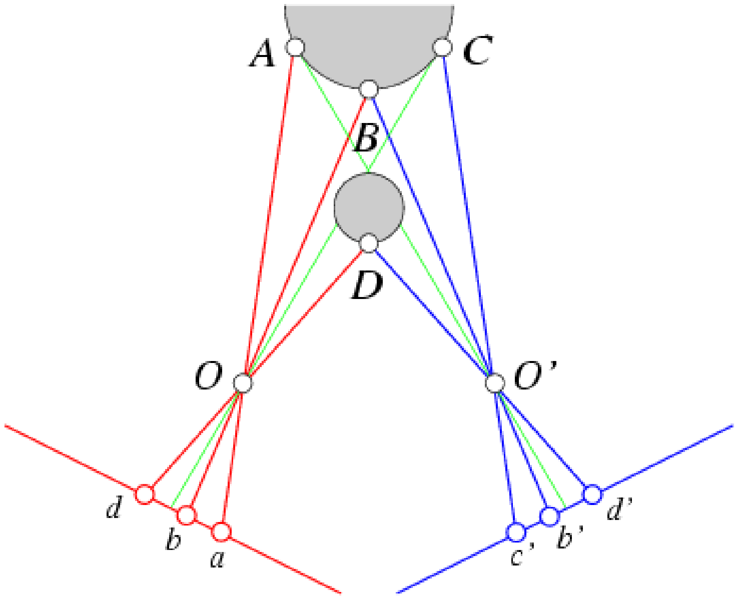
- **Uniqueness**
  - For any point in one image, there should be at most one matching point in the other image
- **Ordering**
  - Corresponding points should be in the same order in both views



Non-local constraints

- **Uniqueness**
  - For any point in one image, there should be at most one matching point in the other image
- **Ordering**
  - Corresponding points should be in the same order in both views





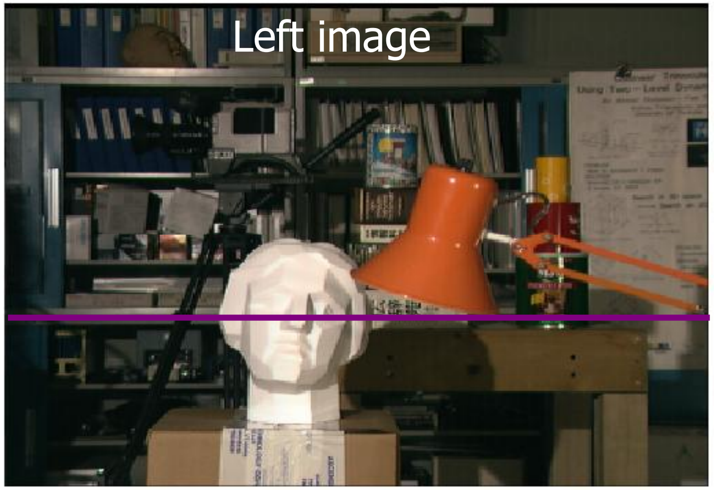
Ordering constraint doesn't hold

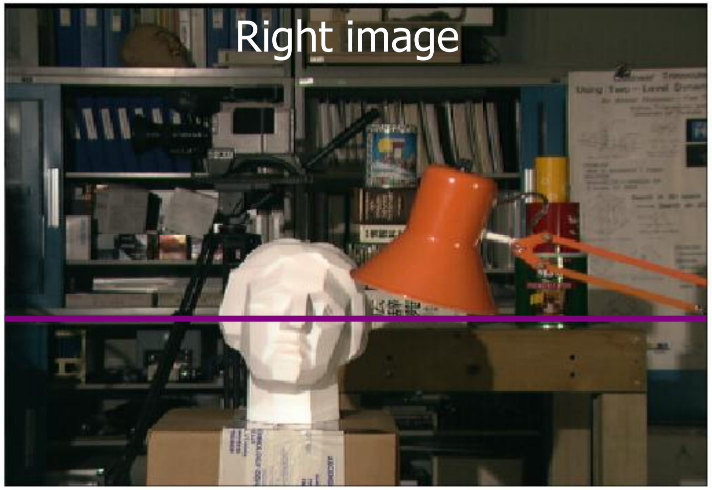
Non-local constraints

- **Uniqueness**
  - For any point in one image, there should be at most one matching point in the other image
- **Ordering**
  - Corresponding points should be in the same order in both views
- **Smoothness**
  - We expect disparity values to change slowly (for the most part)

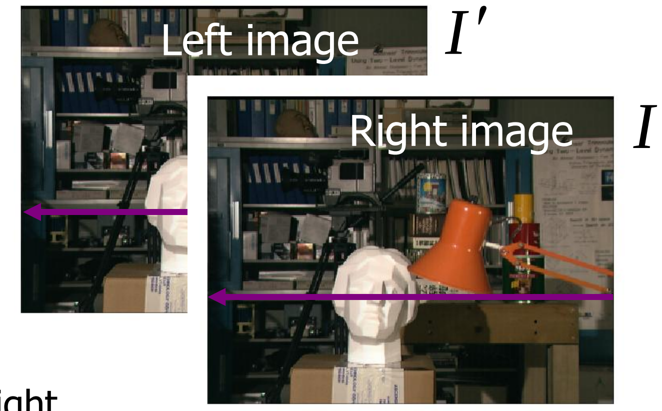
Scanline stereo

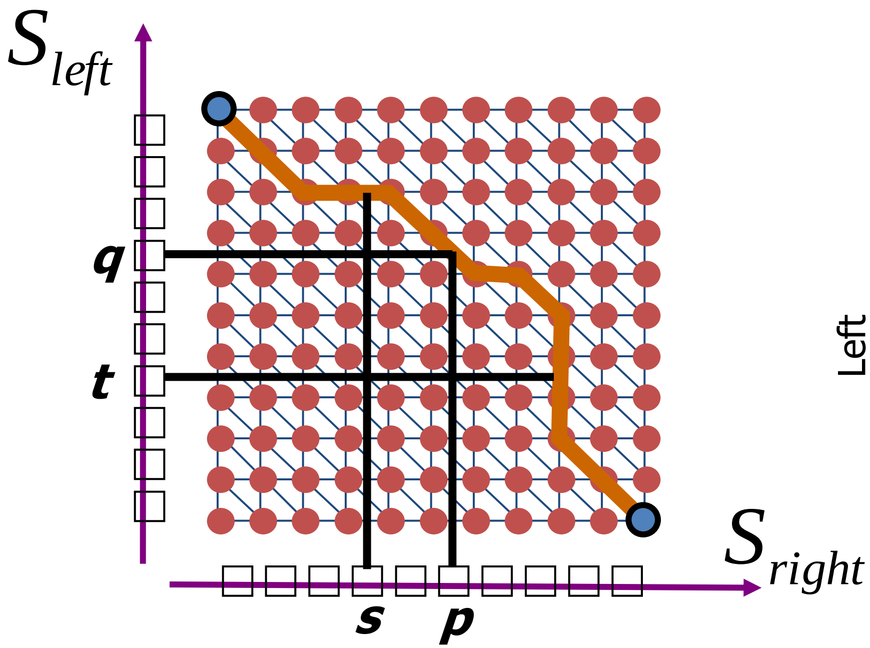
- Try to coherently match pixels on the entire scanline
- Different scanlines are still optimized independently

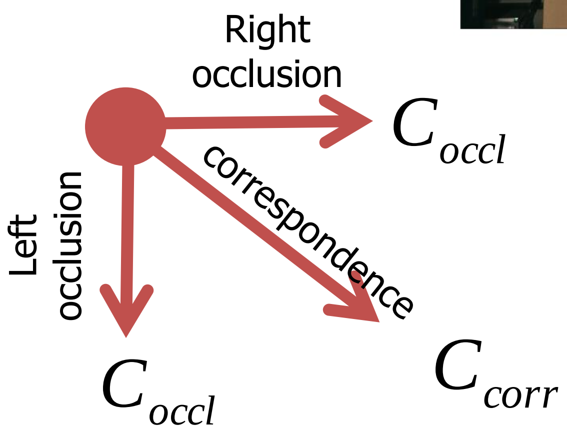




“Shortest paths” for scan-line stereo







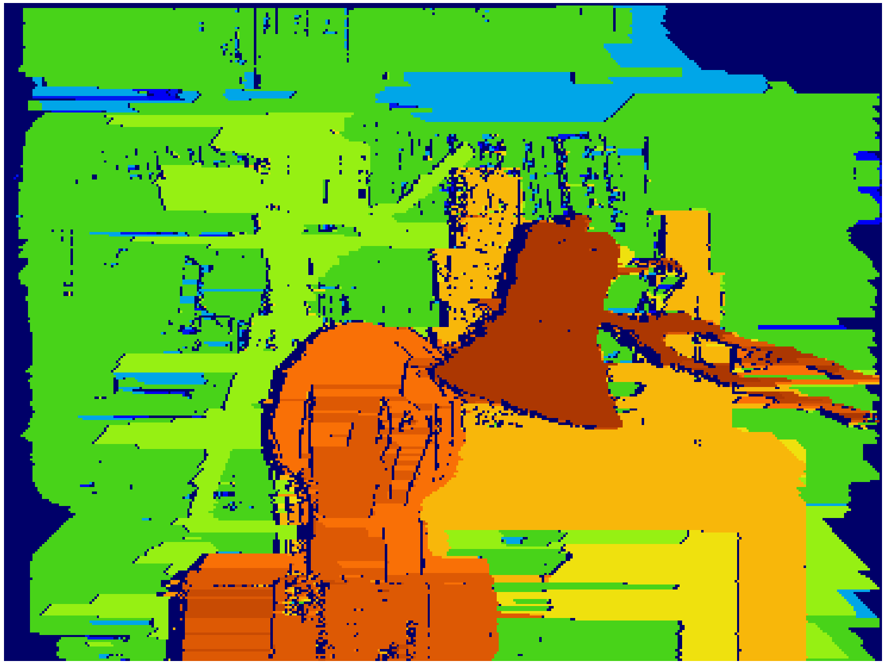
Can be implemented with dynamic programming

Ohta & Kanade ’85, Cox et al. ‘96

Slide credit: Y. Boykov

Coherent stereo on 2D grid

- Scanline stereo generates streaking artifacts



- Can’t use dynamic programming to find spatially coherent disparities/ correspondences on a 2D grid

Stereo as energy minimization

- **Expressing this mathematically**
  1. **Match quality**
     - **Want each pixel to find a good match in the other image**

$$matchCost = \sum_{x,y} \|I(x,y) - J(x + d_{xy}, y)\|$$

  2. **Smoothness**
     - **If two pixels are adjacent, they should (usually) move about the same amount**

$$smoothnessCost = \sum_{neighbor\ \ pixels\ \ p,q} |d_p - d_q|$$

- **We want to minimize** $Energy = matchCost + smoothnessCost$
  - **This is a special type of energy function known as an MRF (Markov Random Field)**
    - **Effective and fast algorithms have been recently developed:**
      - **Graph cuts, belief propagation….**
      - **for more details (and code): http://vision.middlebury.edu/MRF/**
      - **Great tutorials available online (including video of talks)**

Graph cuts solution

Graph cuts

Ground truth

Y. Boykov, O. Veksler, and R. Zabih, [Fast Approximate Energy Minimization via Graph Cuts](), PAMI 2001

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

Example dense correspondence algorithm

left image

right image

3D reconstruction

right image

depth map

intensity = depth

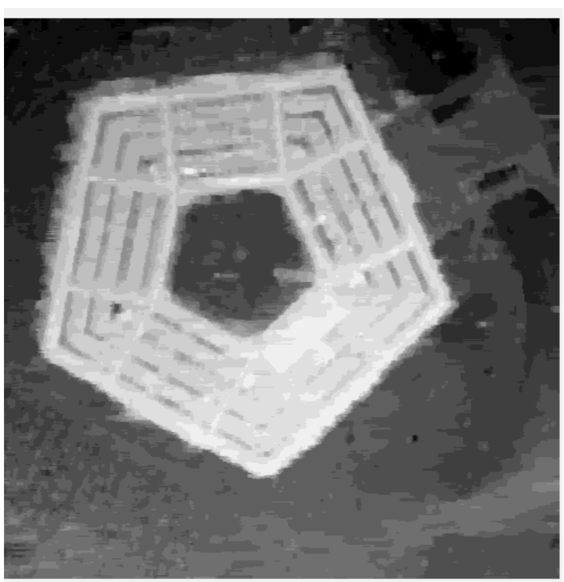
Texture mapped 3D triangulation

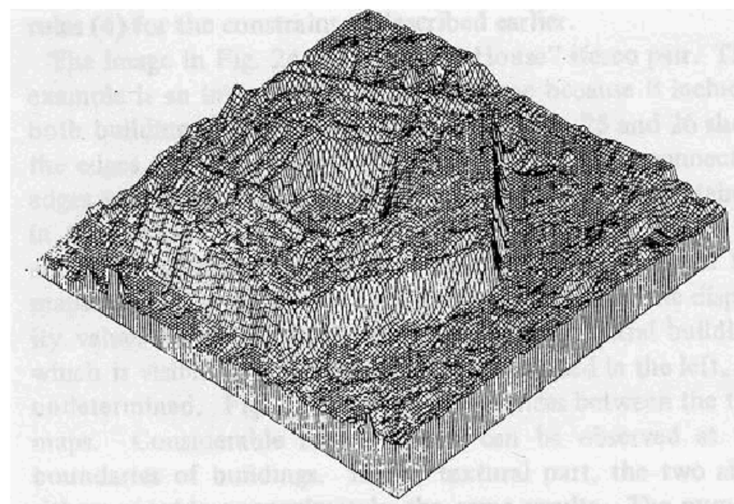
Pentagon example

left image

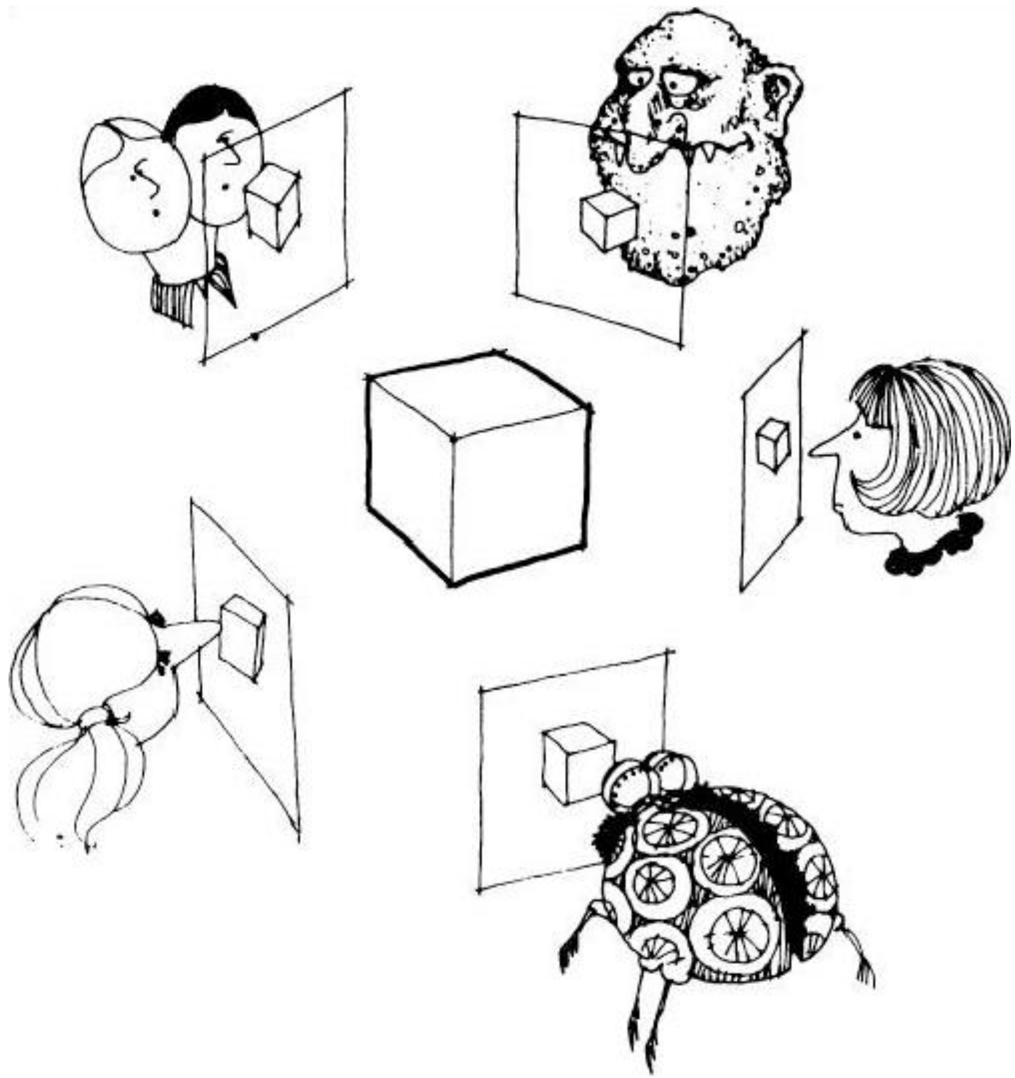
right image

range map





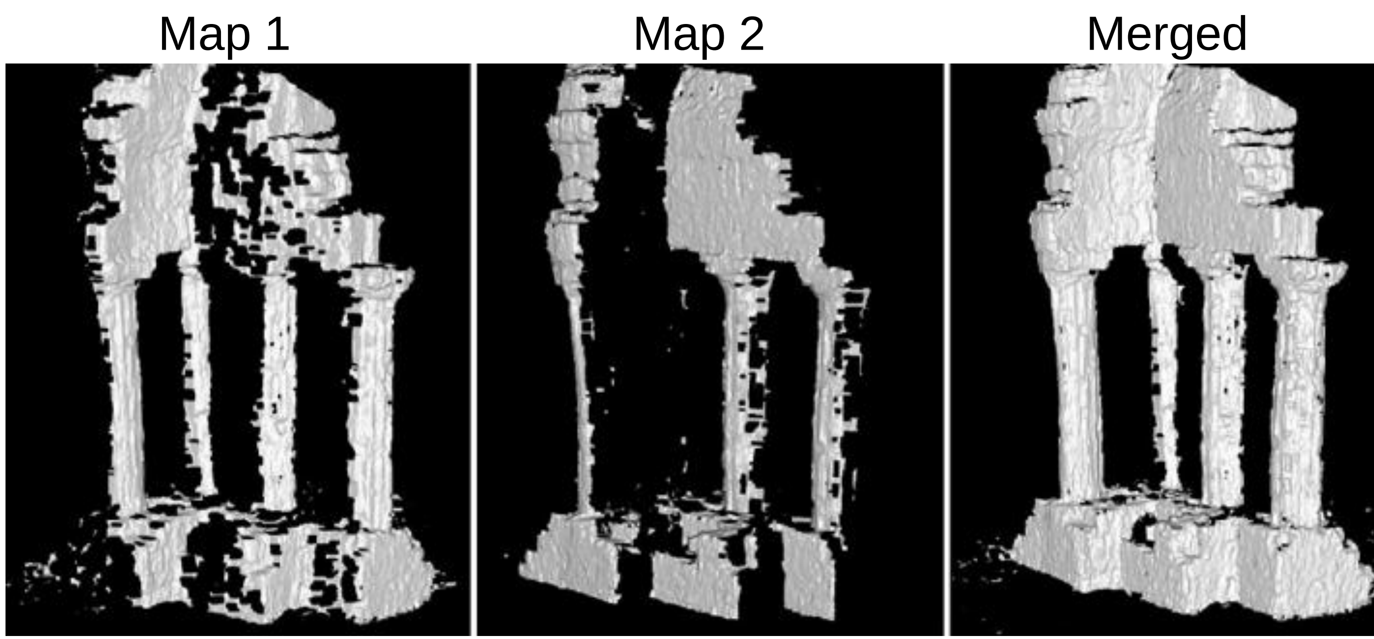
Multi-view stereo



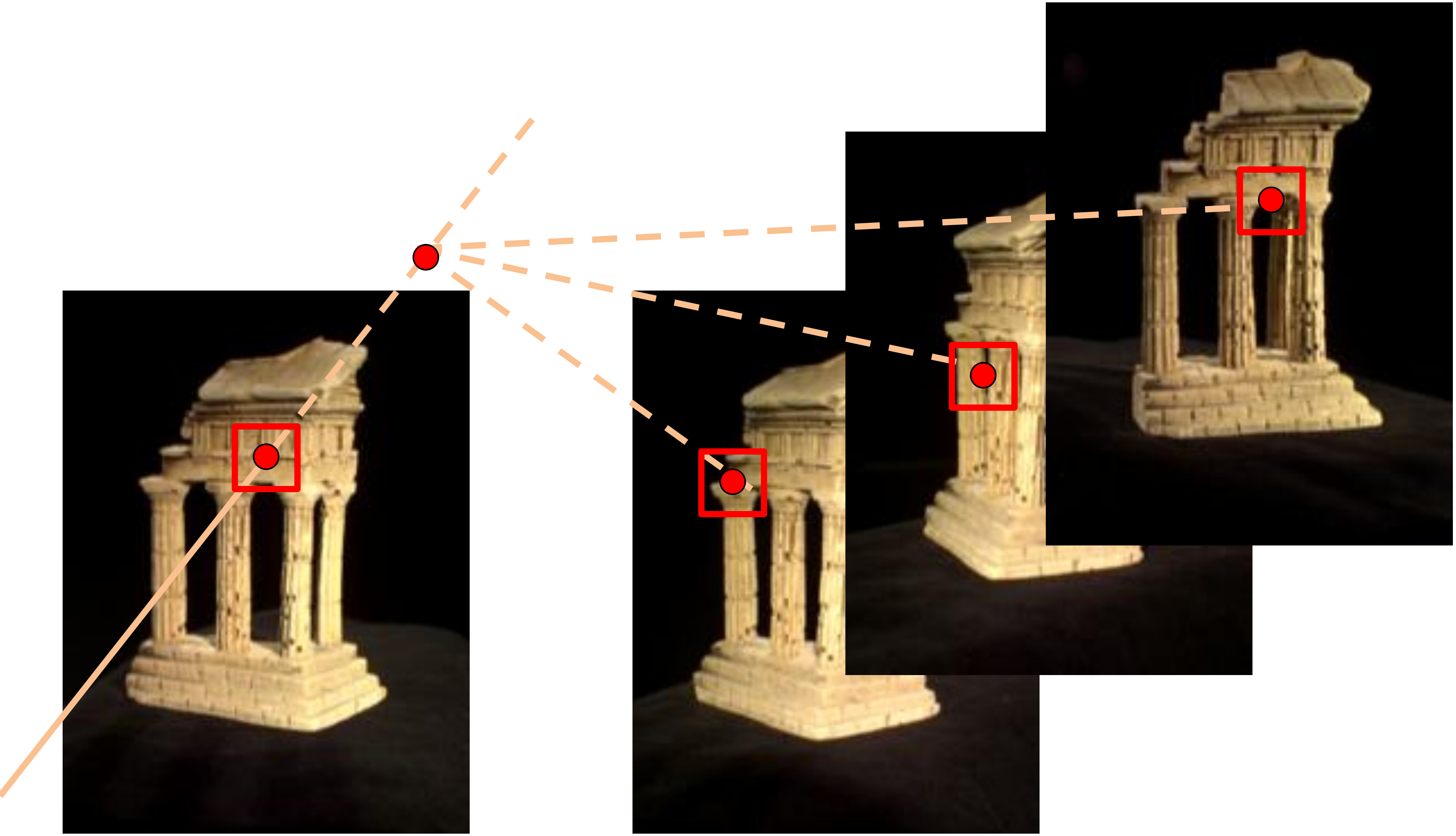
Slides from S. Lazebnik who adapted many from S. Seitz

Beyond two-view stereo

- **A single depth map cannot adequately capture complex 3D structure**

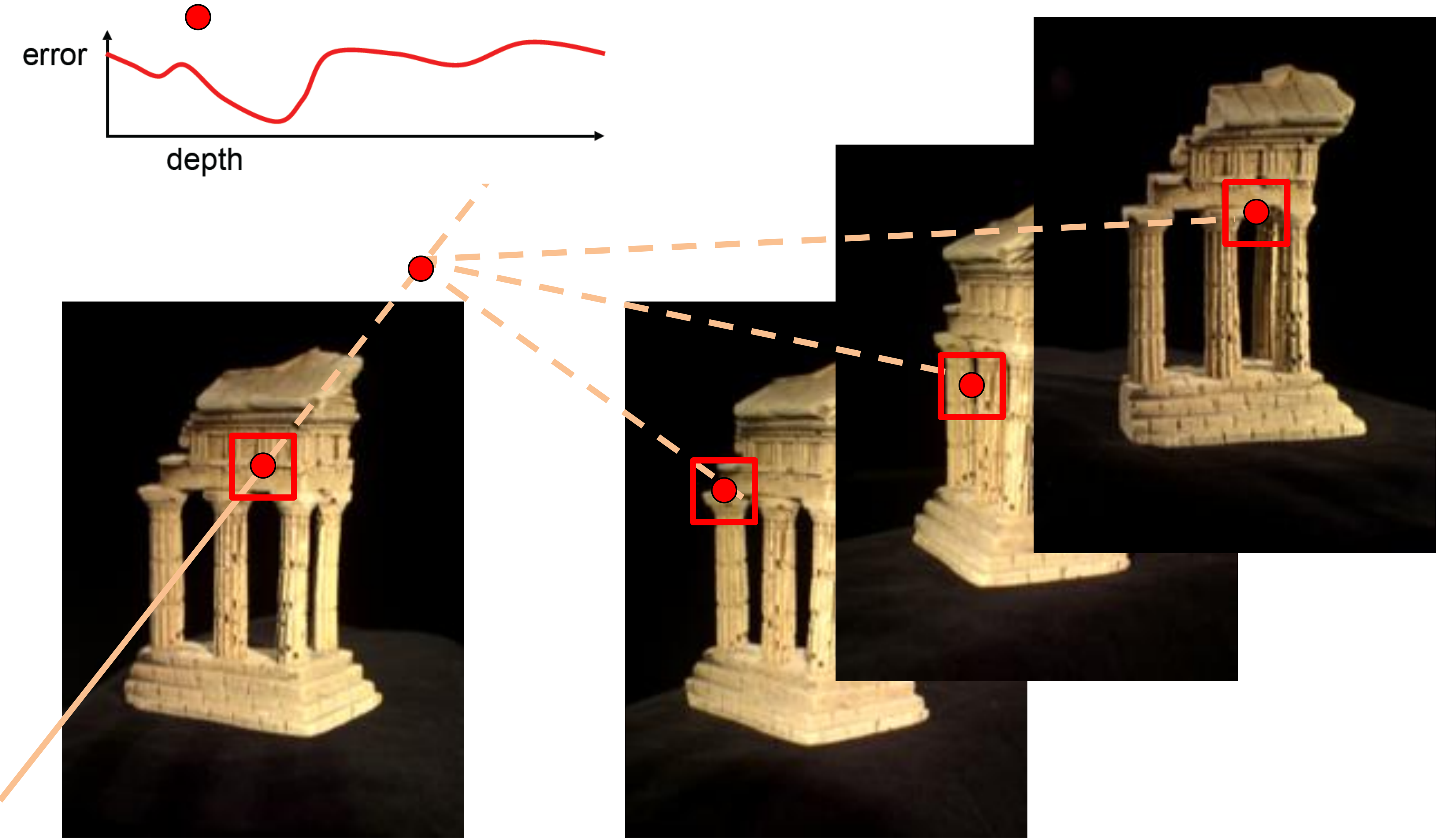


Multi-view stereo: Basic idea



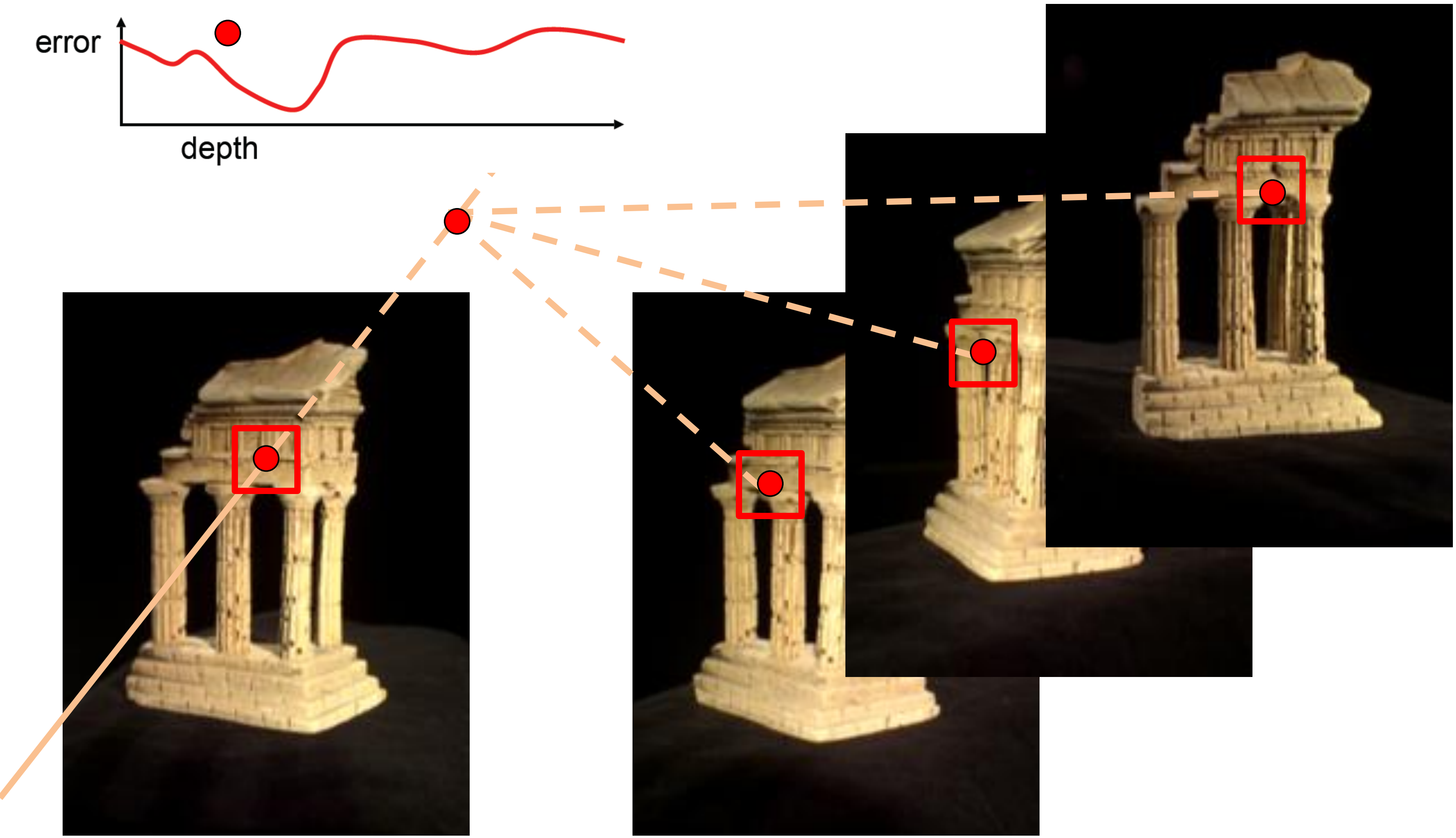
Source: Y. Furukawa

Multi-view stereo: Basic idea



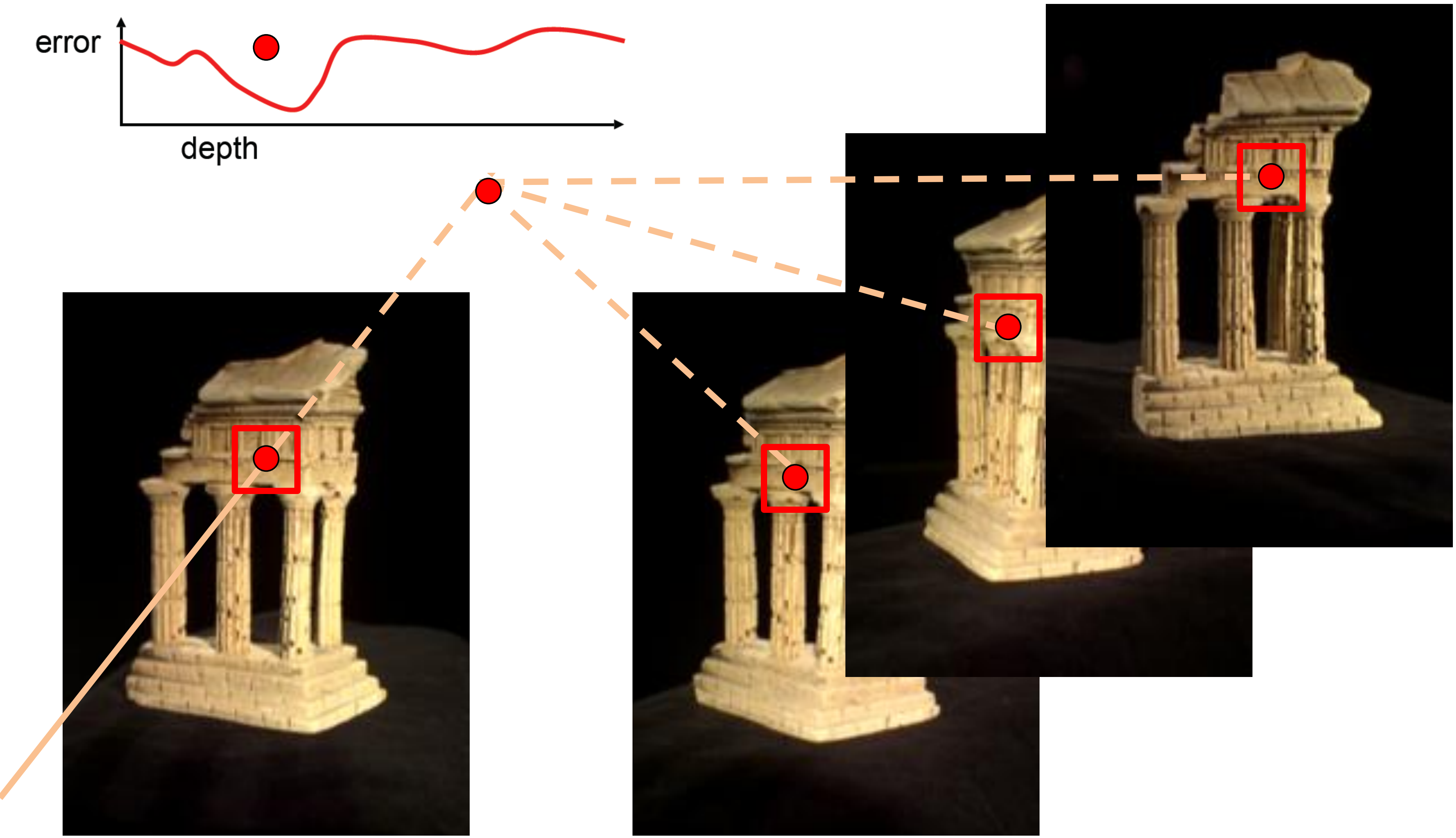
Source: Y. Furukawa

Multi-view stereo: Basic idea



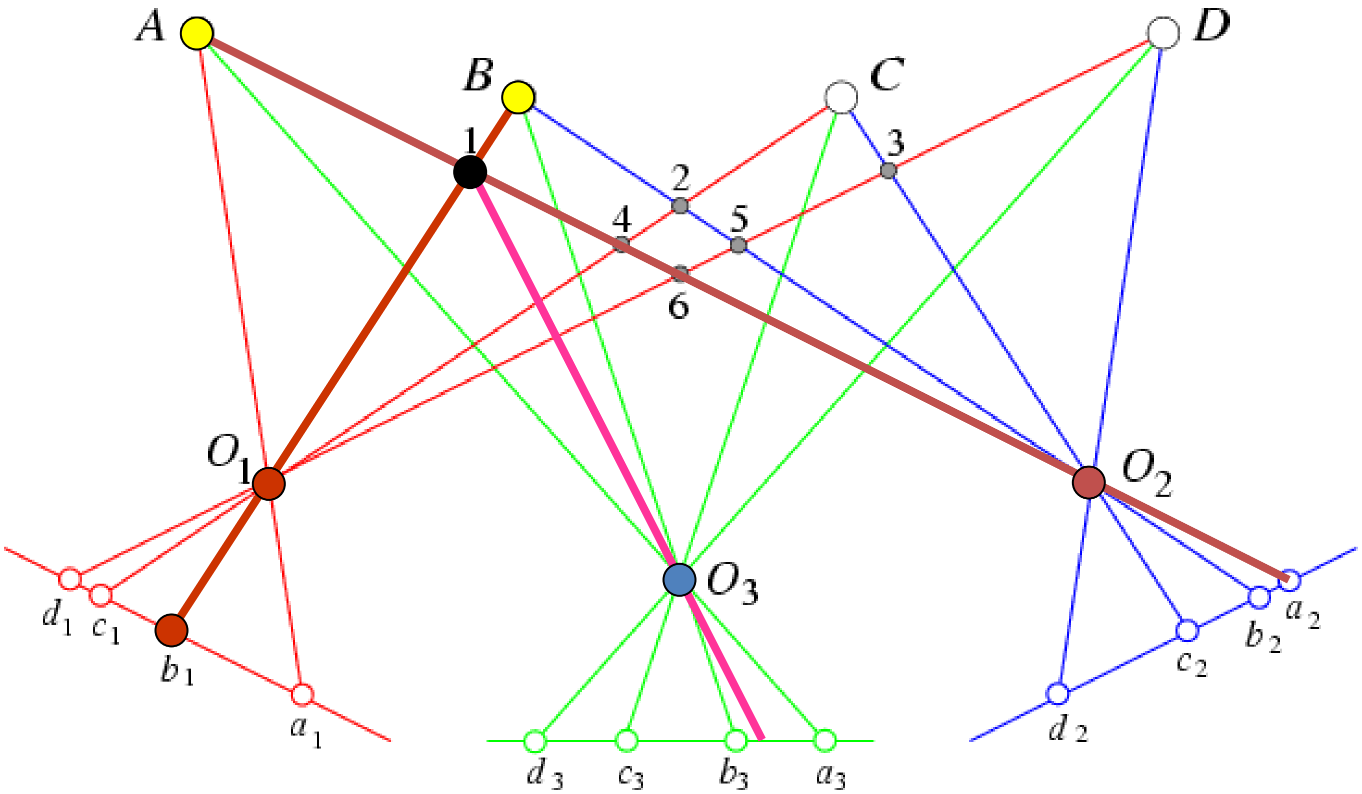
Source: Y. Furukawa

Multi-view stereo: Basic idea



Source: Y. Furukawa

Beyond two-view stereo

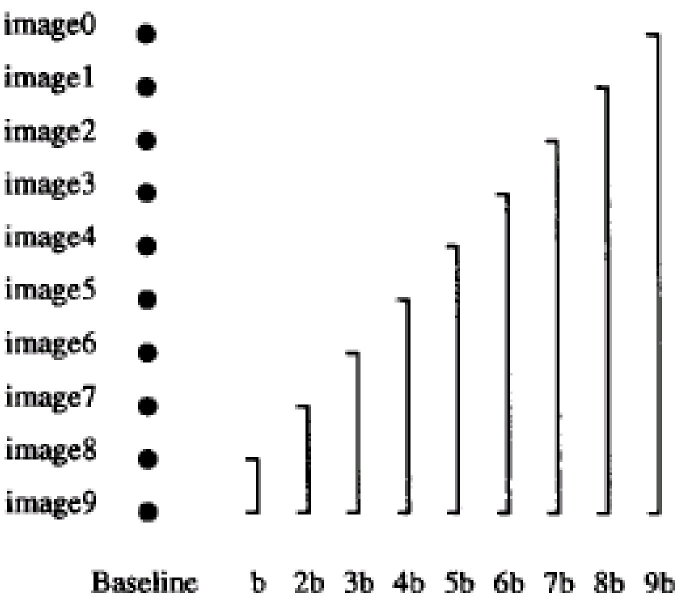


The third view can be used for verification

Multiple-baseline stereo

- **Pick a reference image, and slide the corresponding window along the corresponding epipolar lines of all other images, using inverse depth relative to the first image as the search parameter**

Figure 2: An example scene. The grid pattern in the background has ambiguity of matching.



Remember? disparity

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

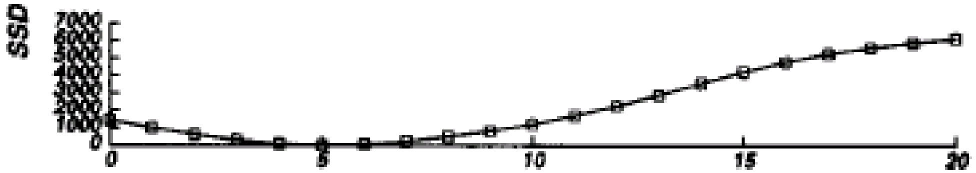
where B is baseline, f is focal length and Z is the depth

This equation indicates that for the same depth the disparity is proportional to the baseline.

M. Okutomi and T. Kanade, [“A Multiple-Baseline Stereo System,”]() IEEE Trans. on Pattern Analysis and Machine Intelligence, 15(4):353-363 (1993).

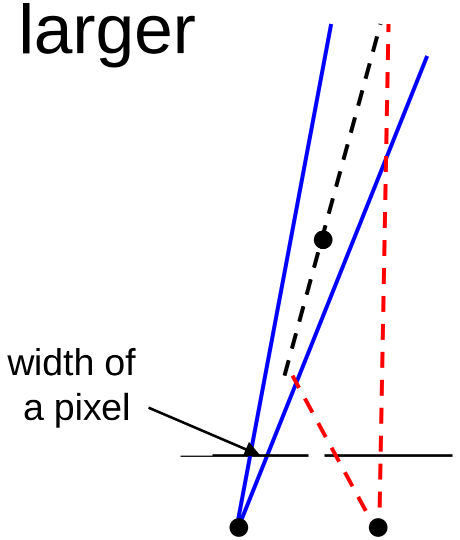
Multiple-baseline stereo

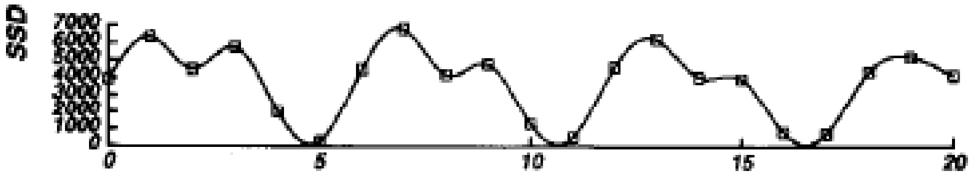
- For larger baselines, must search larger area in second image



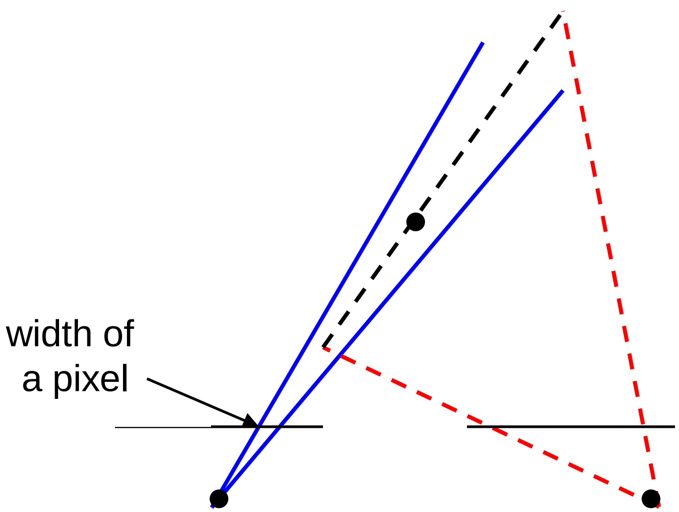
1/z

pixel matching score





1/z



Multiple-baseline stereo

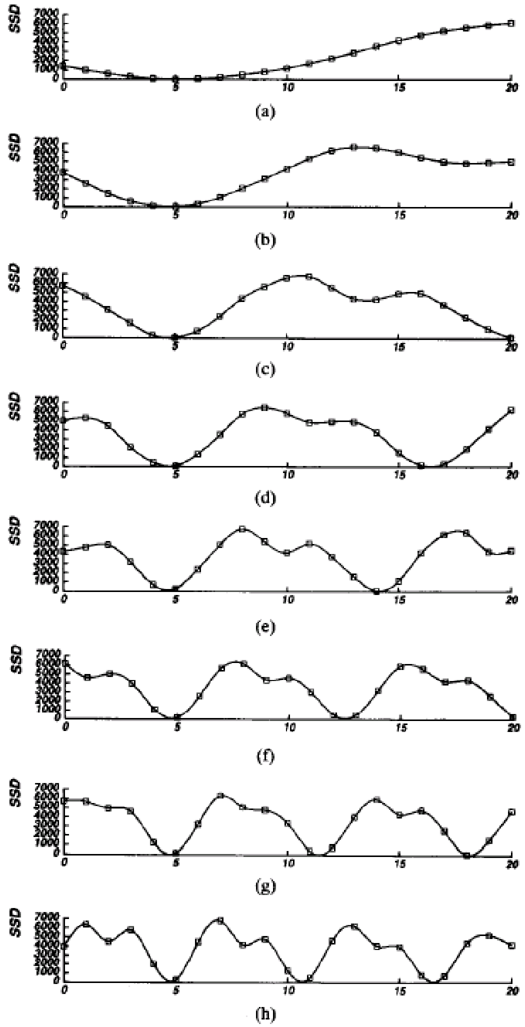


Fig. 5. SSD values versus inverse distance: (a) $B = b$; (b) $B = 2b$; (c) $B = 3b$; (d) $B = 4b$; (e) $B = 5b$; (f) $B = 6b$; (g) $B = 7b$; (h) $B = 8b$. The horizontal axis is normalized such that $8bF = 1$.

Use the sum of SSD scores to rank matches

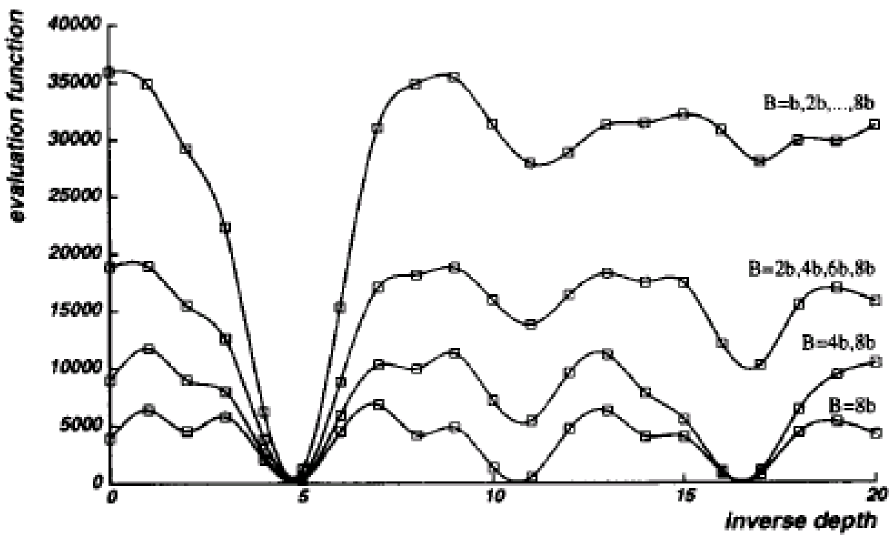


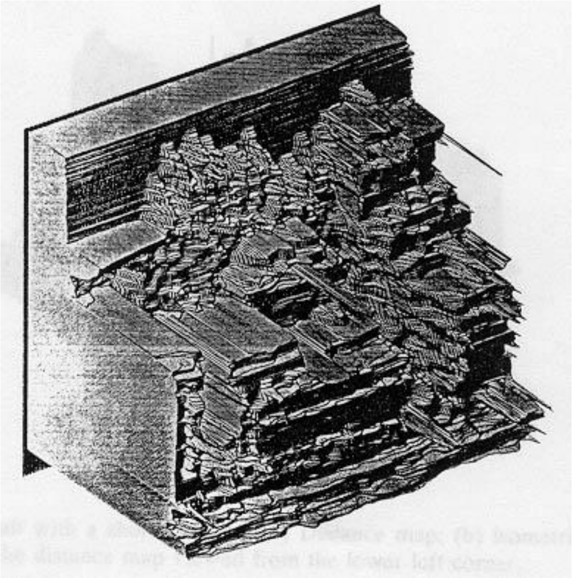
Fig. 7. Combining multiple baseline stereo pairs.

Multiple-baseline stereo results

I1

I2

I10



M. Okutomi and T. Kanade, [“A Multiple-Baseline Stereo System,”](#) IEEE Trans. on Pattern Analysis and Machine Intelligence, 15(4):353-363 (1993).

Summary: Multiple-baseline stereo

- Pros
  - Using multiple images reduces the ambiguity of matching
- Cons
  - Must choose a reference view
  - Occlusions become an issue for large baseline

Readings

- **Chapter 7**

- **D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision, 47(1):7-42, May 2002.**