C280, Computer Vision

Prof. Trevor Darrell
trevor@eecs.berkeley.edu

Lecture 10: Stereo

Roadmap

- Previous: Image formation, filtering, local features, (Texture)…
- Tues: Feature-based Alignment
  - Stitching images together
  - Homographies, RANSAC, Warping, Blending
  - Global alignment of planar models
- Today: Dense Motion Models
  - Local motion / feature displacement
  - Parametric optic flow
- No classes next week: ICCV conference
- **Oct 6th: Stereo / 'Multi-view': Estimating depth with known inter-camera pose**
- Oct 8th: 'Structure-from-motion': Estimation of pose and 3D structure
  - Factorization approaches
  - Global alignment with 3D point models

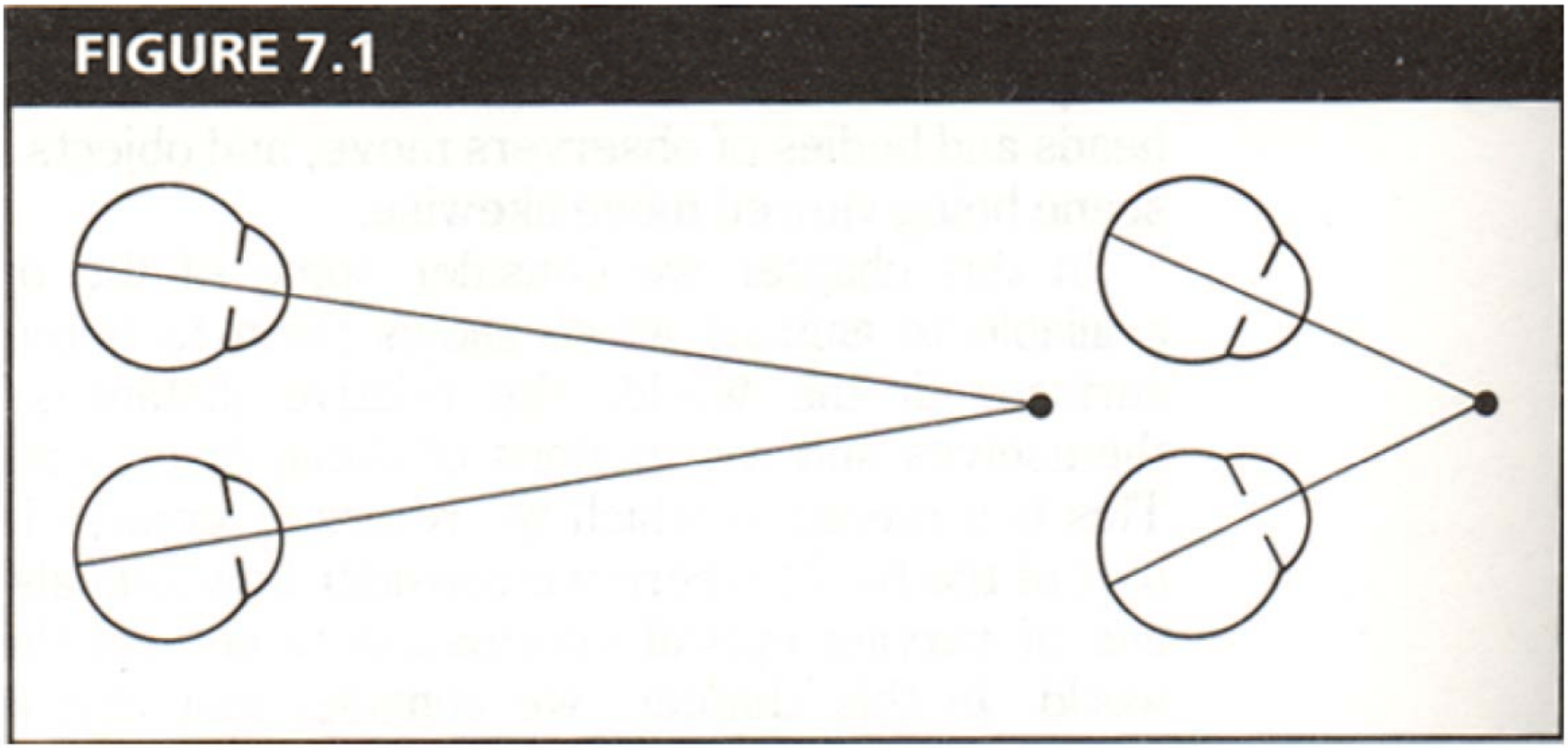
Last time: Motion and Flow

- Motion estimation
- Patch-based motion (optic flow)
- Regularization and line processes
- Parametric (global) motion
- Layered motion models

Today: Stereo

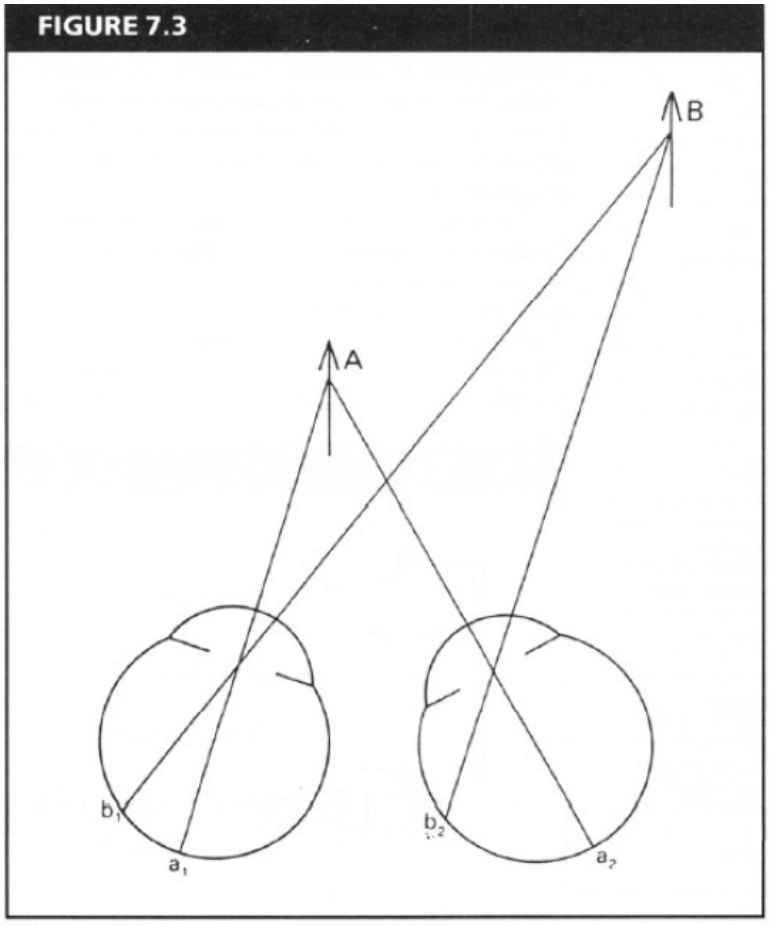
- Human stereopsis & stereograms
- Epipolar geometry and the epipolar constraint
  - Case example with parallel optical axes
  - General case with calibrated cameras
- Correspondence search
- The Essential and the Fundamental Matrix
- Multi-view stereo

Fixation, convergence



From Bruce and Green, Visual Perception, Physiology, Psychology and Ecology

Human stereopsis: disparity

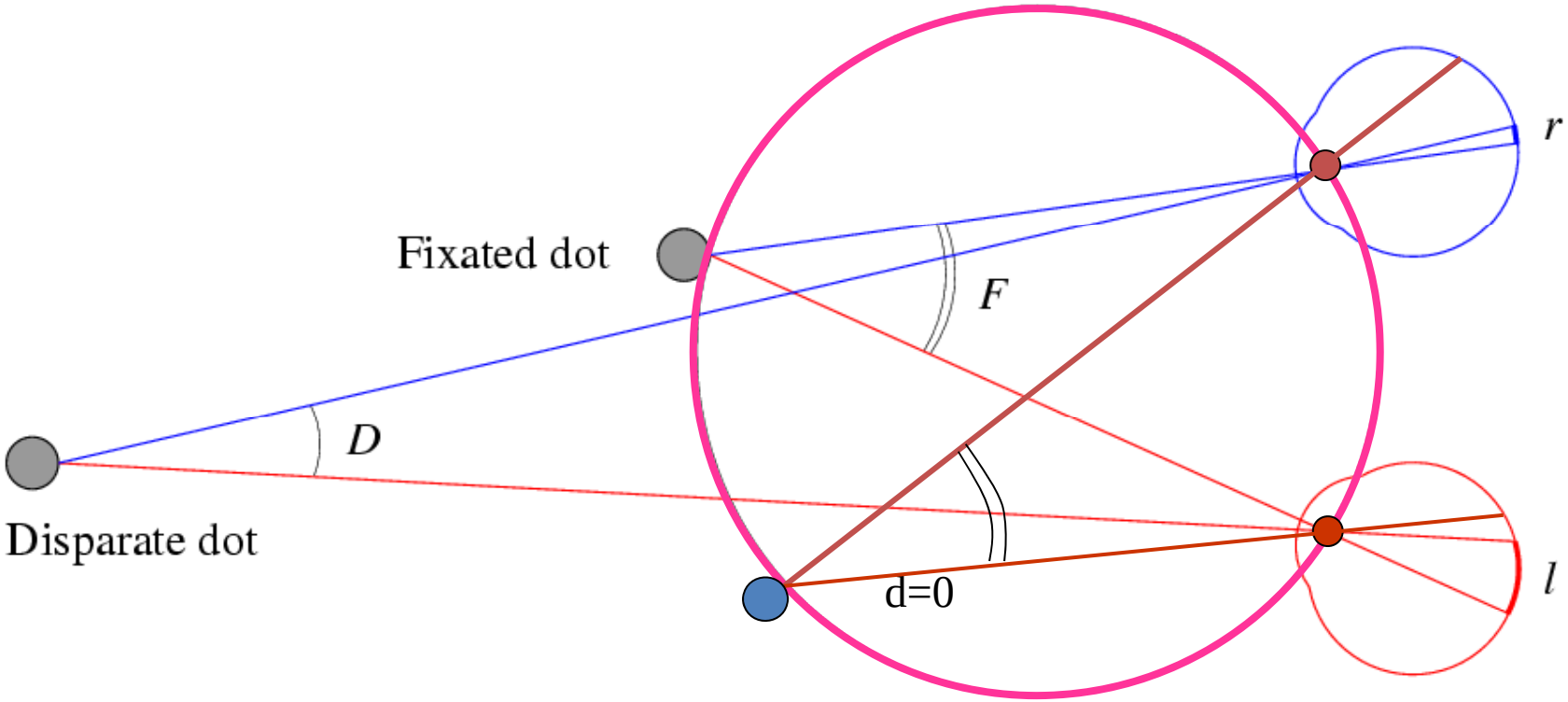


From Bruce and Green, Visual Perception, Physiology, Psychology and Ecology

Disparity occurs when eyes fixate on one object; others appear at different visual angles

Adapted from David Forsyth, UC Berkeley

Human stereopsis: disparity



Disparity: $d = r - l = D - F$.

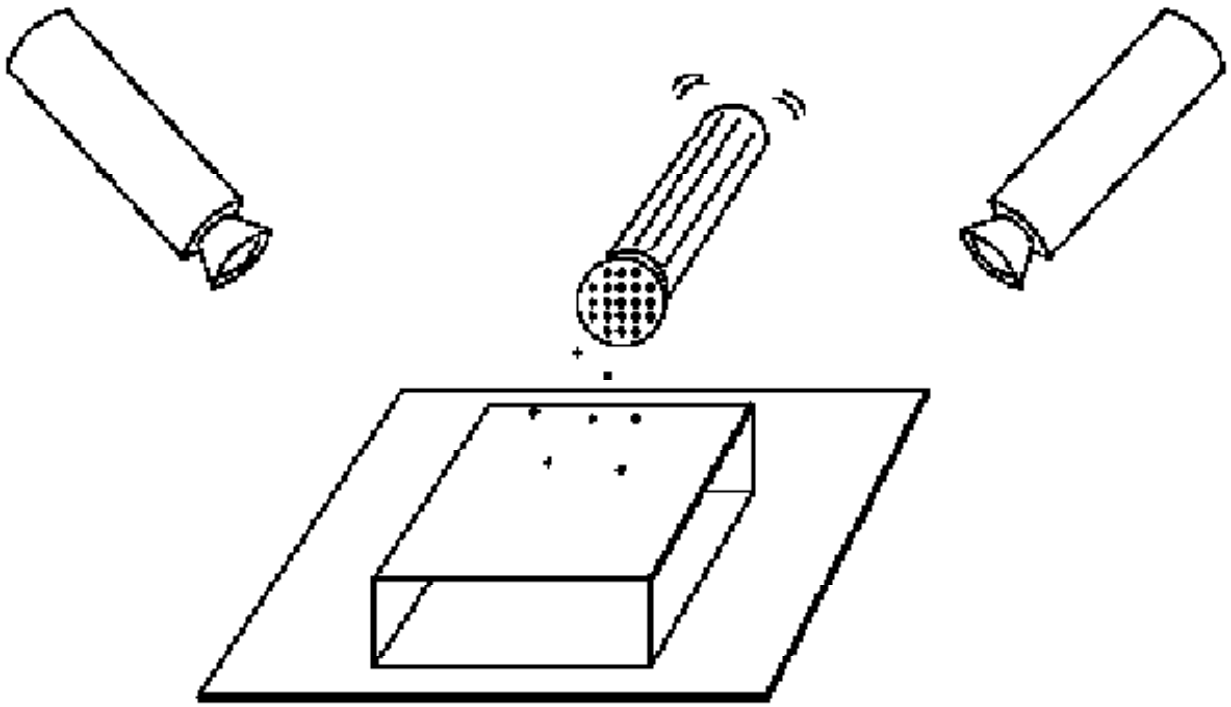
Adapted from M. Pollefeys

Random dot stereograms

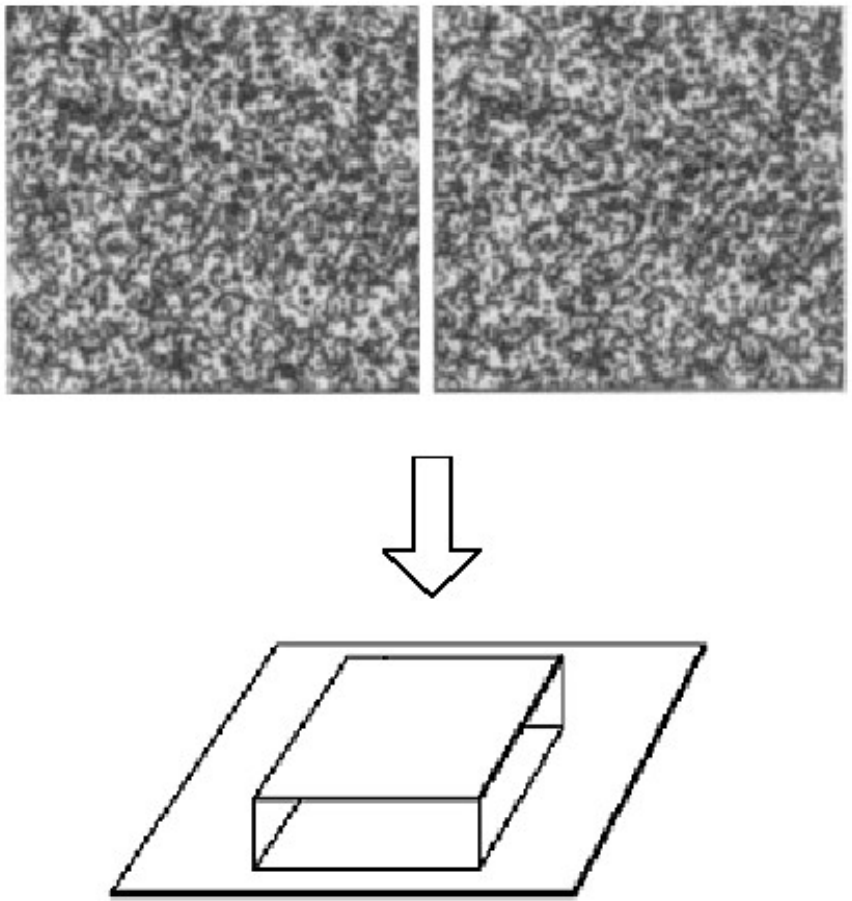
- Julesz 1960: Do we identify local brightness patterns before fusion (monocular process) or after (binocular)?

- To test: pair of synthetic images obtained by randomly spraying black dots on white objects

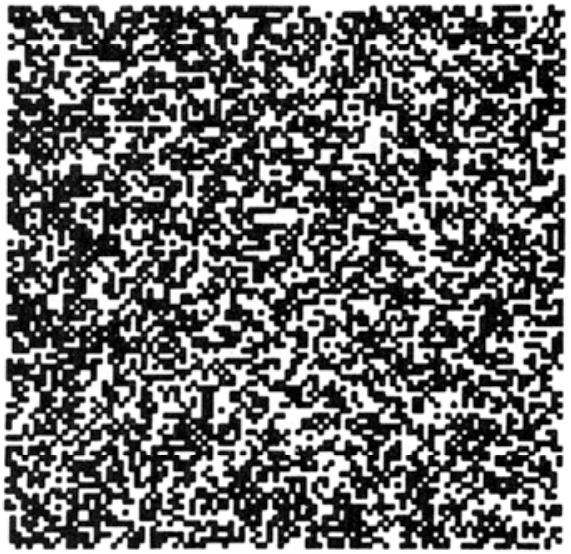
Random dot stereograms



Random dot stereograms



Random dot stereograms



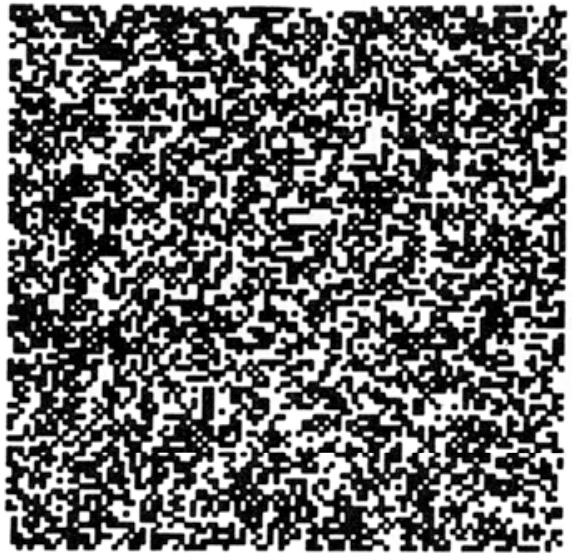


Figure 5.3.8 A random dot stereogram. These two images are derived from a single array of randomly placed squares by laterally displacing a region of them as described in the text. When they are viewed with crossed disparity (by crossing the eyes) so that the right eye's view of the left image is combined with the left eye's view of the right image, a square will be perceived to float above the page. (See pages 210–211 for instructions on fusing stereograms.)

From Palmer, "Vision Science", MIT Press

Random dot stereograms

- When viewed monocularly, they appear random; when viewed stereoscopically, see 3d structure.

- Conclusion: human binocular fusion not directly associated with the physical retinas; must involve the central nervous system
- Imaginary "*cyclopean retina*" that combines the left and right image stimuli as a single unit

Autostereograms

Exploit disparity as depth cue using single image

(Single image random dot stereogram, Single image stereogram)

Images from magiceye.com

Autostereograms

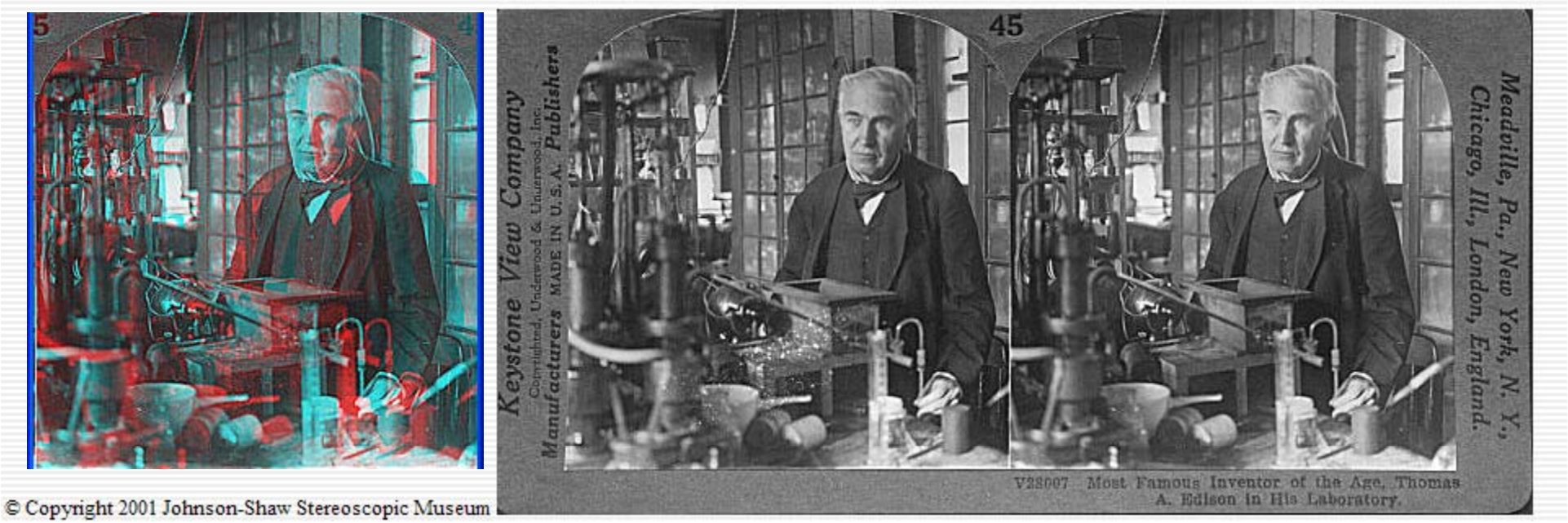
Images from magiceye.com

Stereo photography and stereo viewers

Take two pictures of the same subject from two slightly different viewpoints and display so that each eye sees only one of the images.

Invented by Sir Charles Wheatstone, 1838

Image courtesy of fisher-price.com



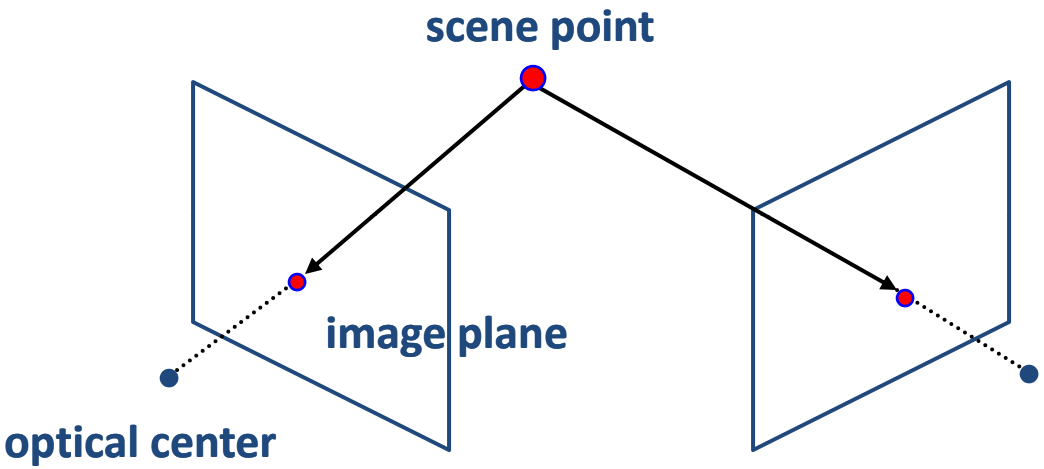
http://www.johnsonshawmuseum.org

http://www.johnsonshawmuseum.org

Public Library, Stereoscopic Looking Room, Chicago, by Phillips, 1923

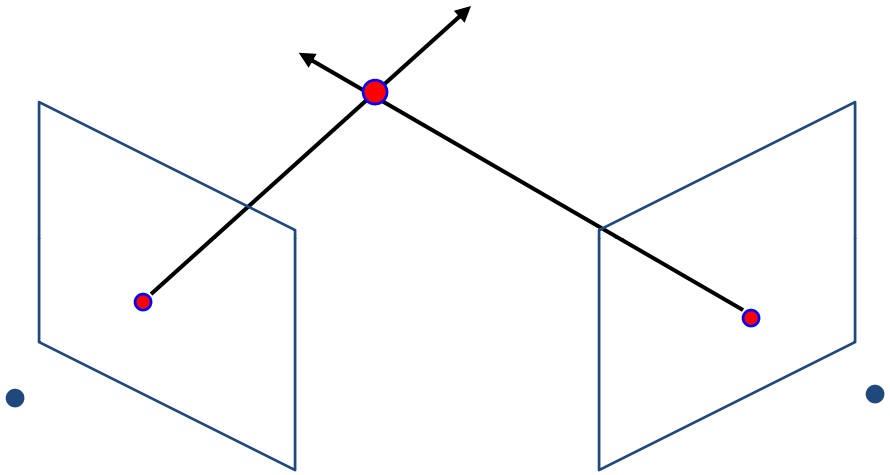
http://www.well.com/~jimg/stereo/stereo_list.html

Depth with stereo: basic idea



Source: Steve Seitz

Depth with stereo: basic idea

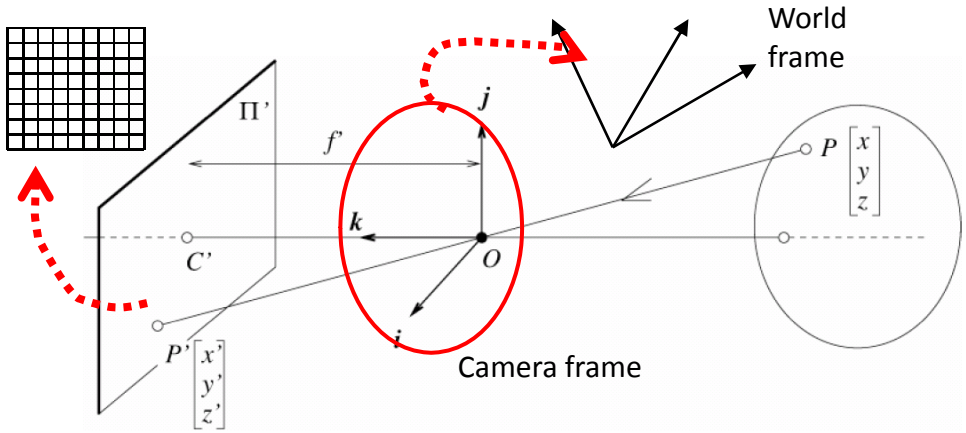


Basic Principle: Triangulation

- Gives reconstruction as intersection of two rays
- Requires
  - camera pose (calibration)
  - ***point correspondence***

Source: Steve Seitz

Camera calibration



Extrinsic parameters:
Camera frame ←→ Reference frame

Intrinsic parameters:
Image coordinates relative to camera
←→ Pixel coordinates

- *Extrinsic* params: rotation matrix and translation vector
- *Intrinsic* params: focal length, pixel sizes (mm), image center point, radial distortion parameters

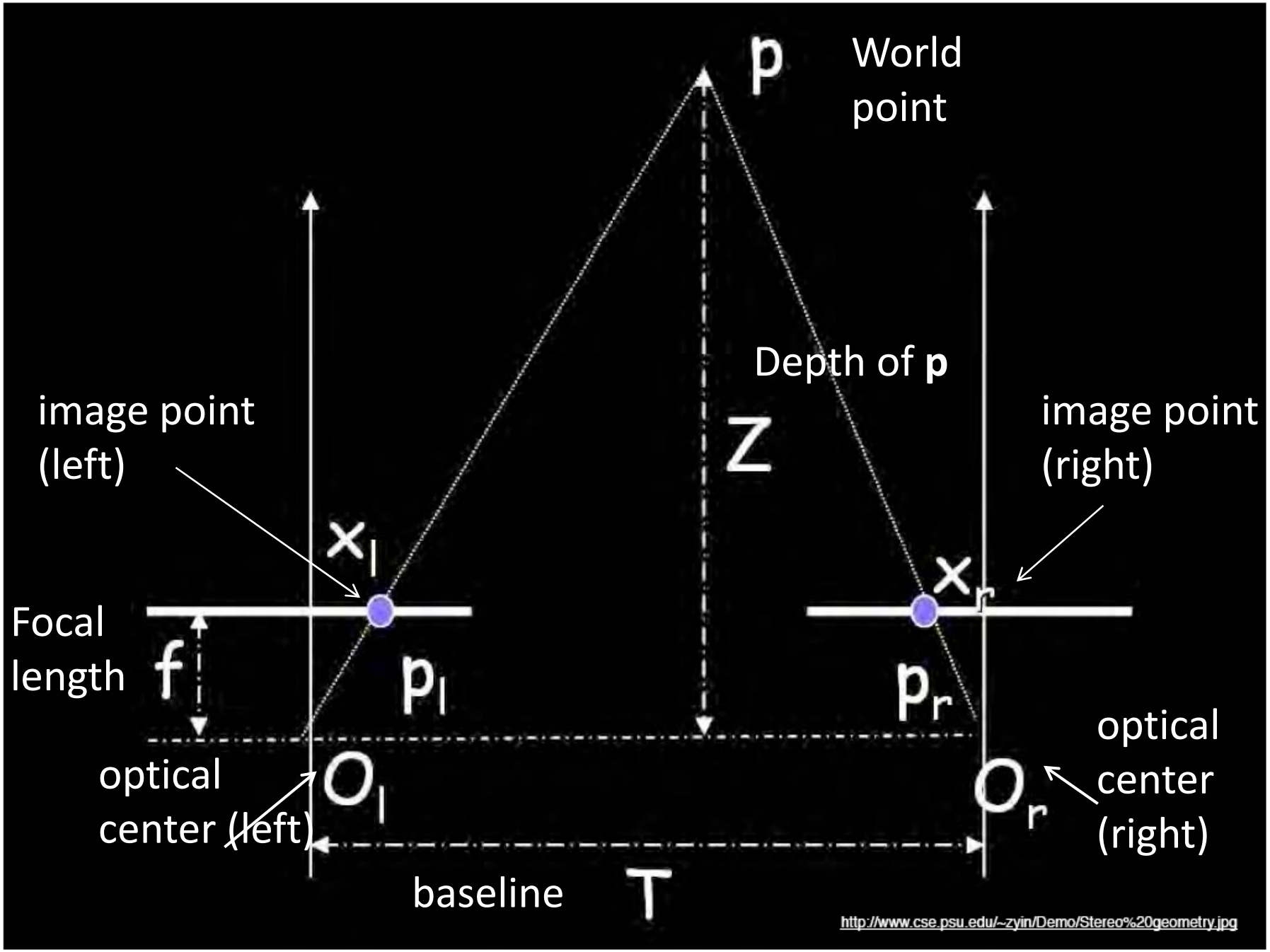
We'll assume for now that these parameters are given and fixed.

Today

- Human stereopsis
- Stereograms
- Epipolar geometry and the epipolar constraint
  - Case example with parallel optical axes
  - General case with calibrated cameras
- Stereopsis
  - Finding correspondences along the epipolar line

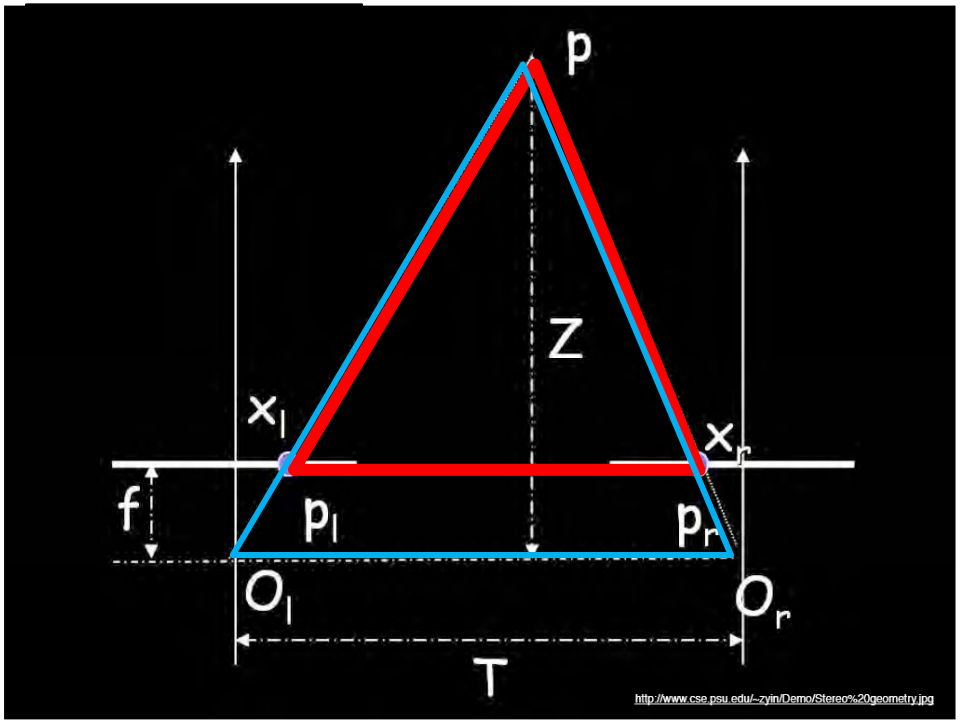
Geometry for a simple stereo system

- First, assuming parallel optical axes, known camera parameters (i.e., calibrated cameras):


p
World point
Depth of p
Z
image point (left)
image point (right)
x_l
x_r
Focal length
f
p_l
p_r
optical center (left)
O_l
O_r
optical center (right)
baseline
T
http://www.cse.psu.edu/~zyin/Demo/Stereo%20geometry.jpg

Geometry for a simple stereo system

- Assume parallel optical axes, known camera parameters (i.e., calibrated cameras). We can triangulate via:



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

Disparity example

image I(x,y)

Disparity map D(x,y)

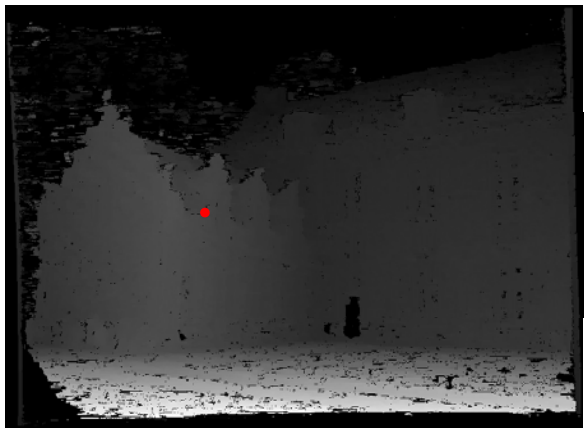
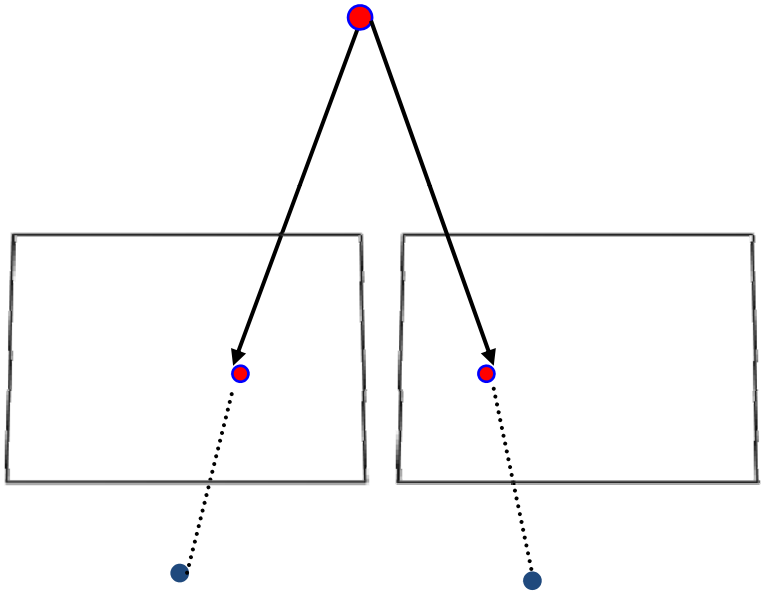


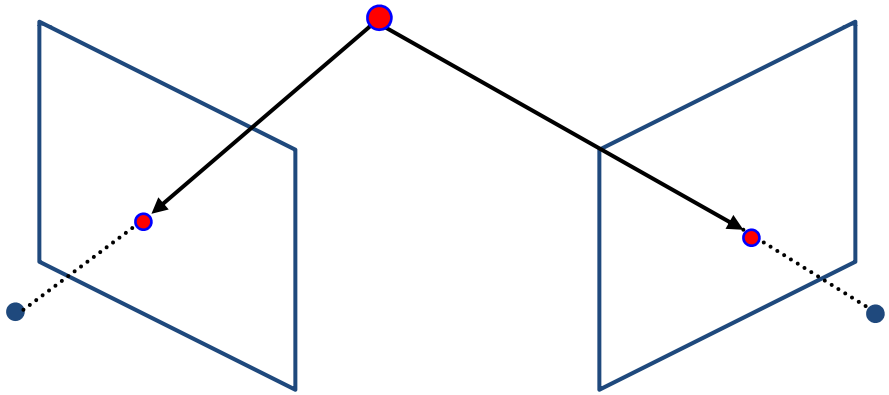
image I´(x´,y´)

$(x´,y´)=(x+D(x,y), y)$

General case, with calibrated cameras

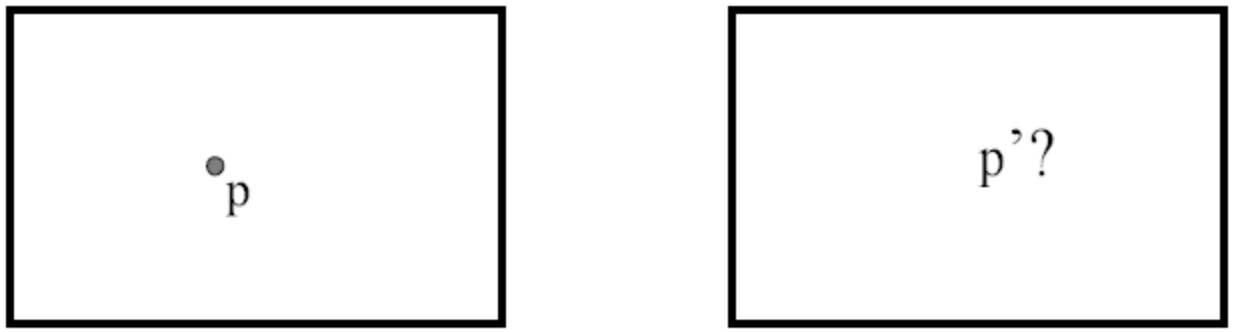
- The two cameras need not have parallel optical axes.





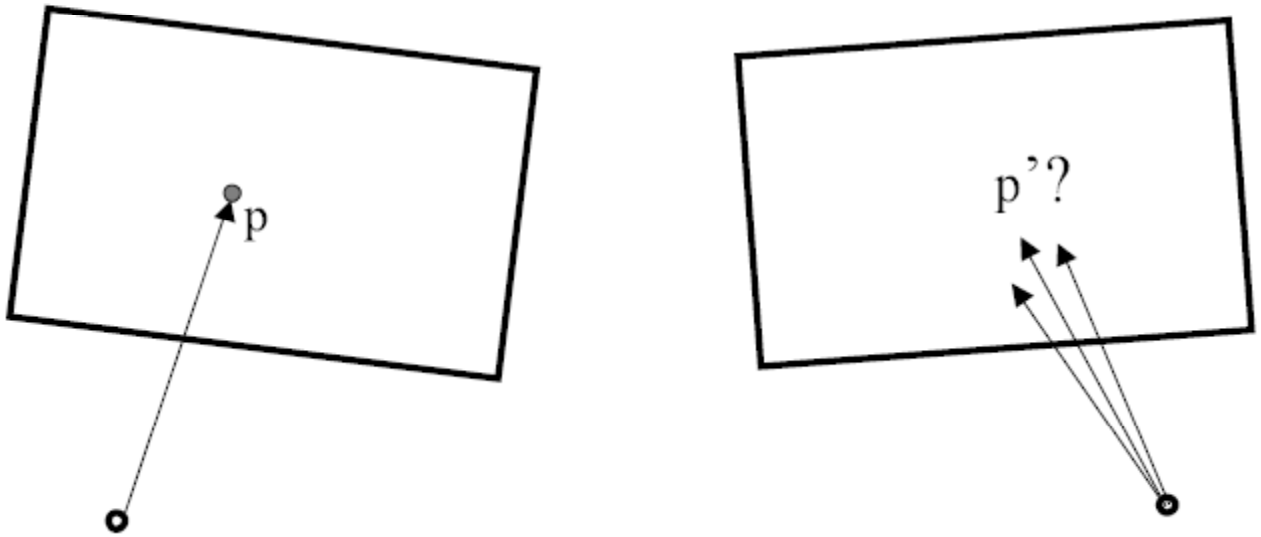
Vs.

Stereo correspondence constraints



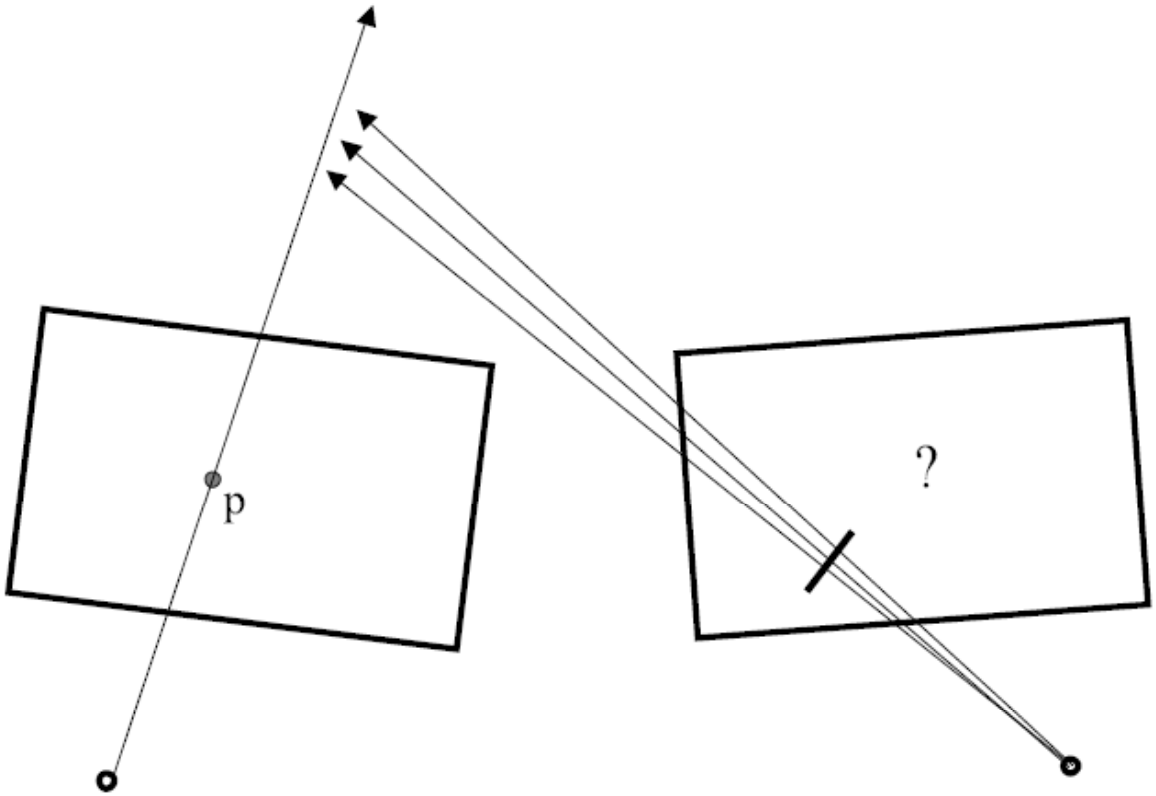
- Given p in left image, where can corresponding point p' be?

Stereo correspondence constraints



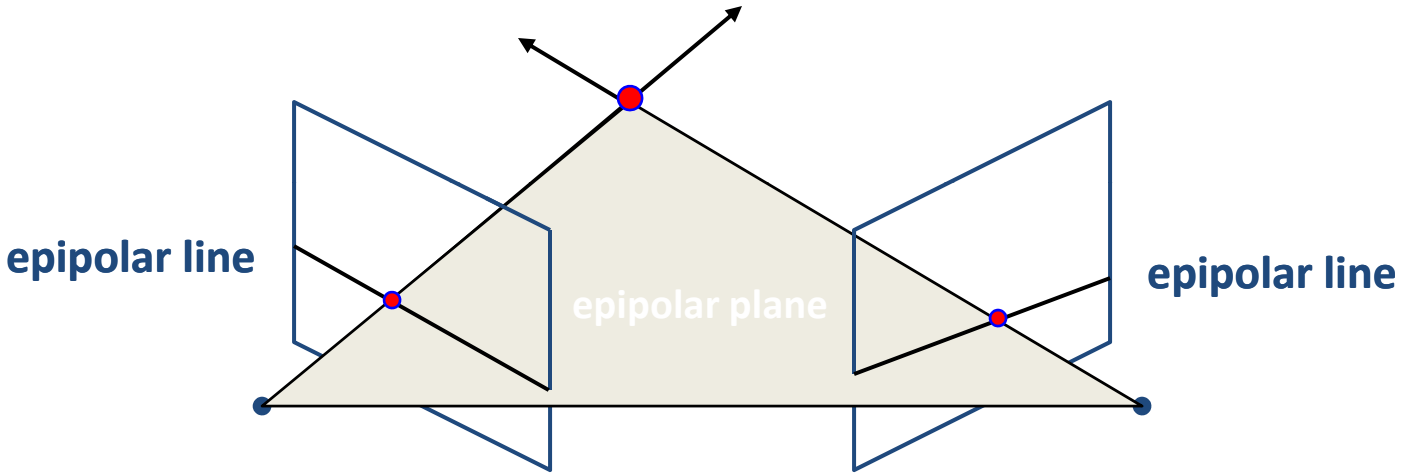
- Given p in left image, where can corresponding point p' be?

Stereo correspondence constraints



Stereo correspondence constraints

•Geometry of two views allows us to constrain where the corresponding pixel for some image point in the first view must occur in the second view.

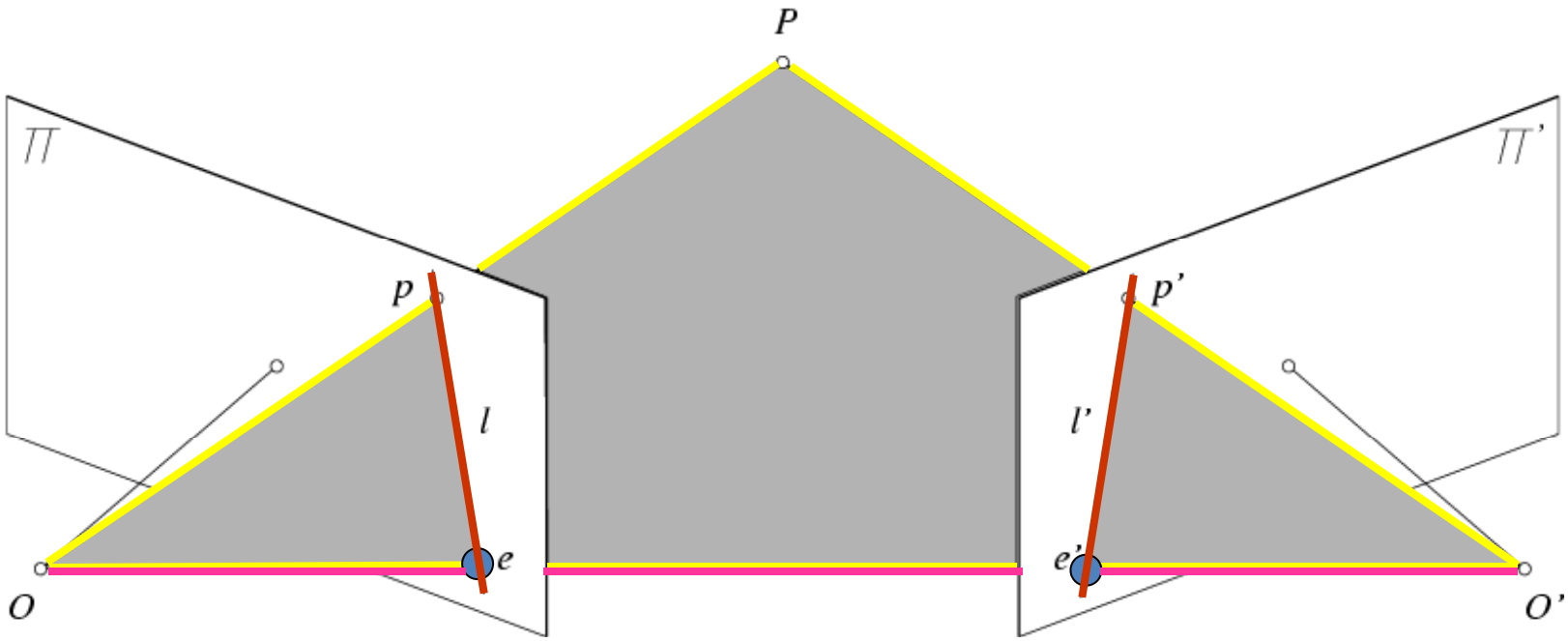


Epipolar constraint: Why is this useful?

- Reduces correspondence problem to 1D search along *conjugate epipolar lines*

Adapted from Steve Seitz

Epipolar geometry



- Epipolar Plane
- Baseline
- Epipoles
- Epipolar Lines

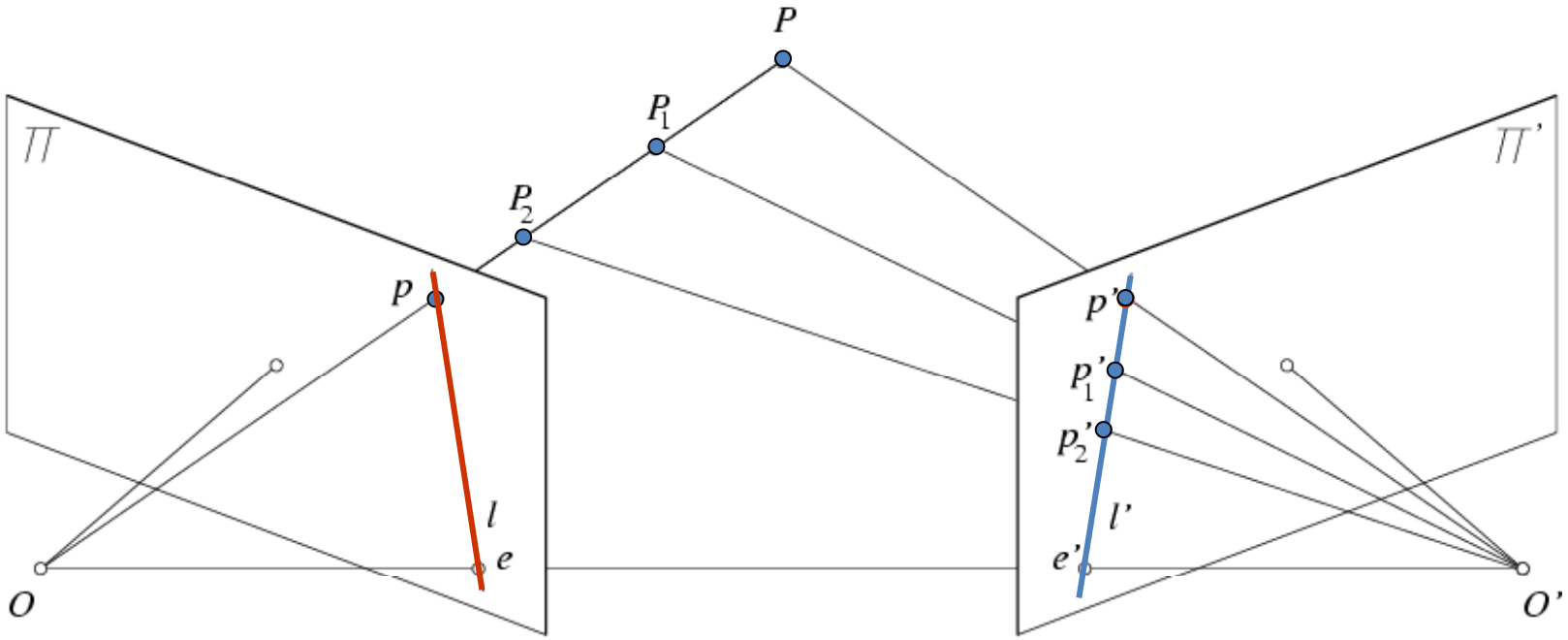
Adapted from M. Pollefeys, UNC

Epipolar geometry: terms

- **Baseline**: line joining the camera centers
- **Epipole**: point of intersection of baseline with the image plane
- **Epipolar plane**: plane containing baseline and world point
- **Epipolar line**: intersection of epipolar plane with the image plane

- All epipolar lines intersect at the epipole
- An epipolar plane intersects the left and right image planes in epipolar lines

Epipolar constraint

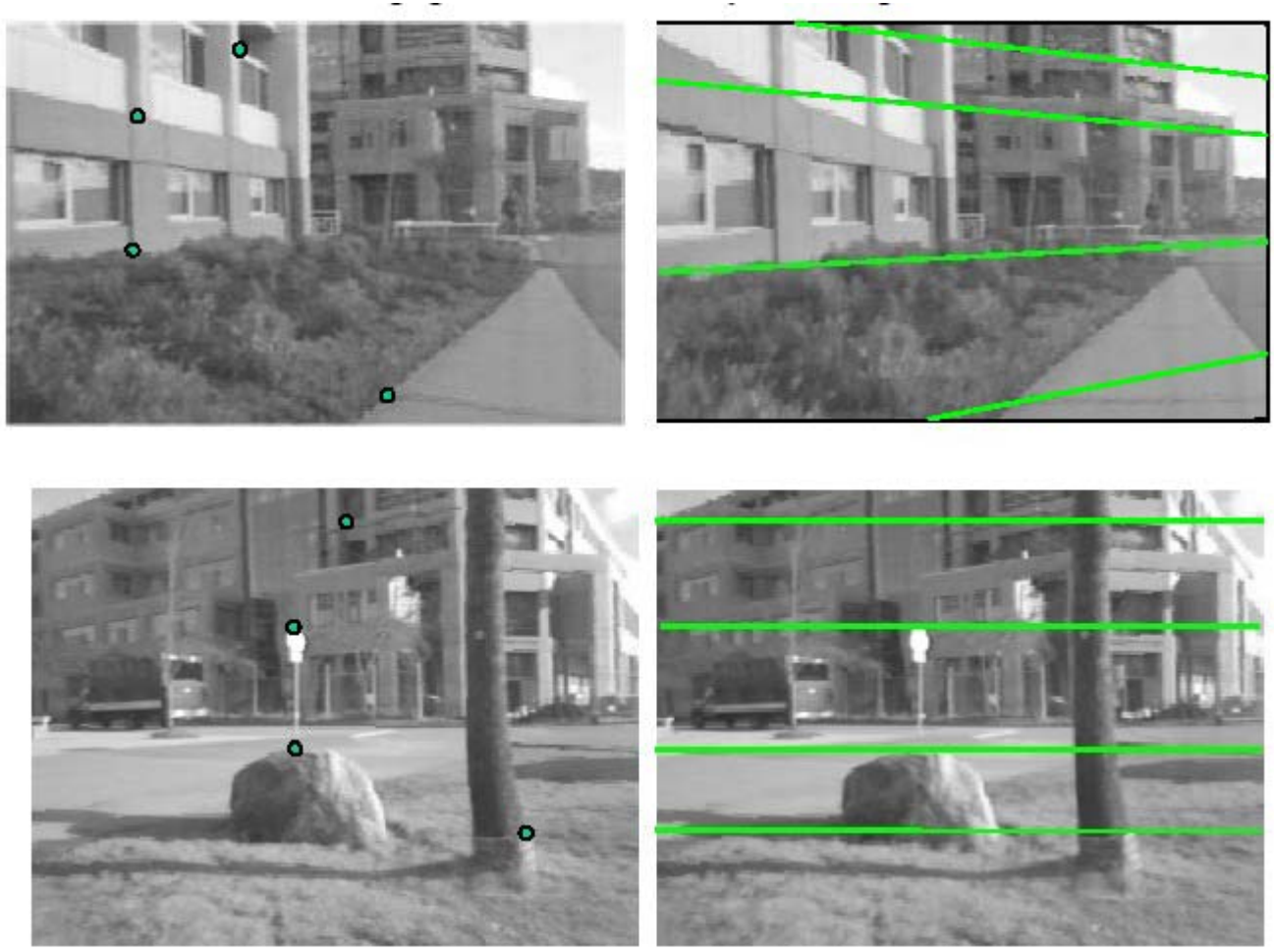


- Potential matches for *p* have to lie on the corresponding epipolar line *l'*.
- Potential matches for *p'* have to lie on the corresponding epipolar line *l*.

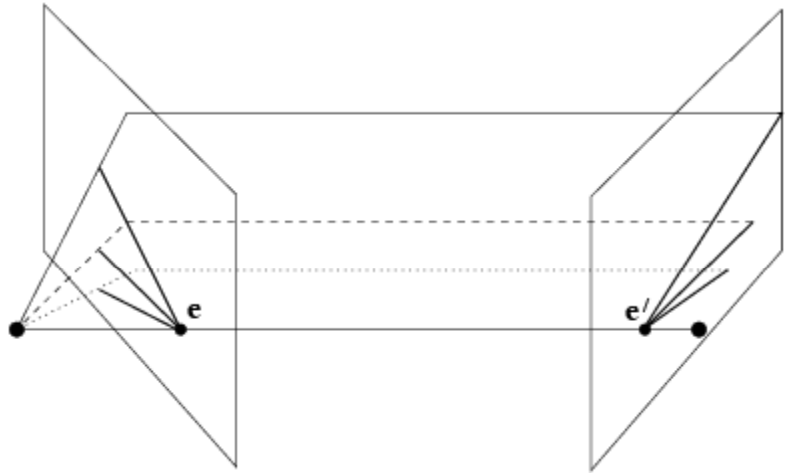
http://www.ai.sri.com/~luong/research/Meta3DViewer/EpipolarGeo.html

Source: M. Pollefeys

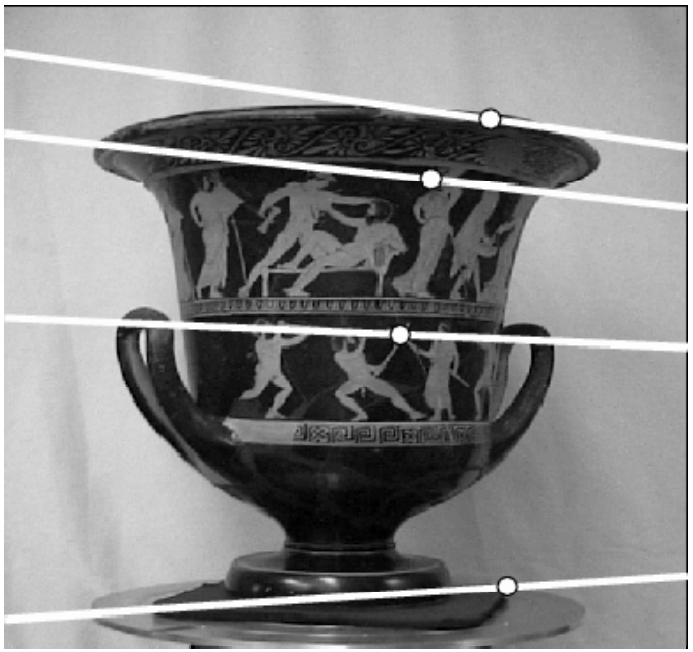
Example



Example: converging cameras



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



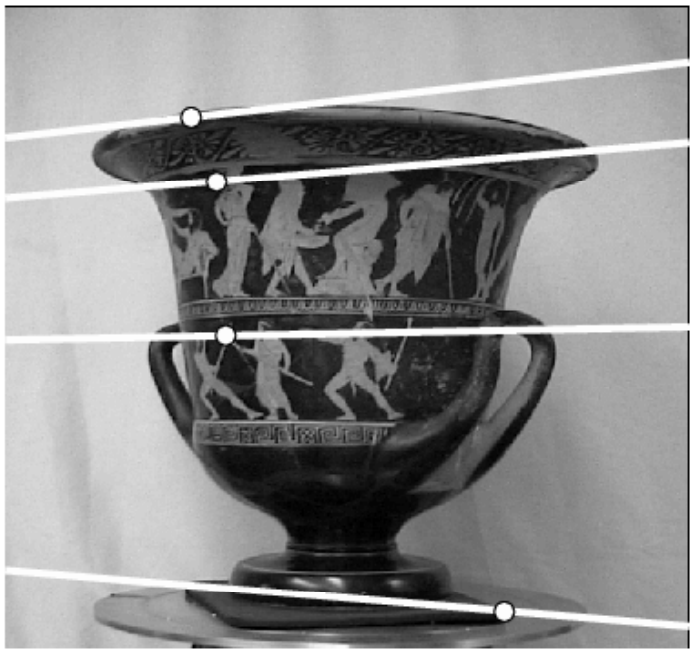
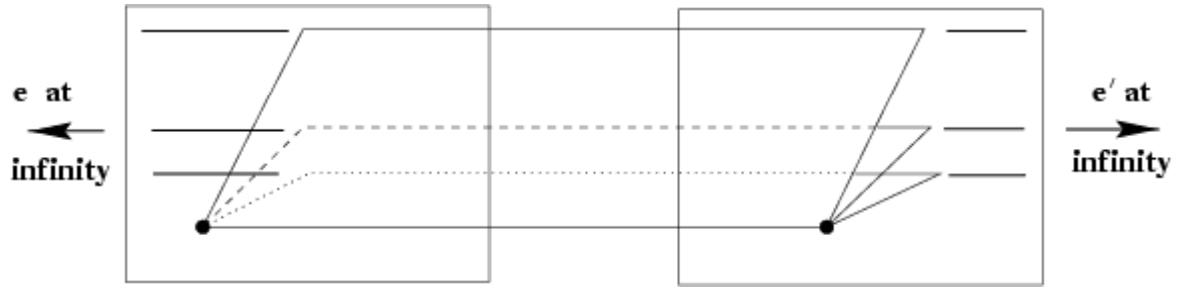


Figure from Hartley & Zisserman

Example: motion parallel with image plane



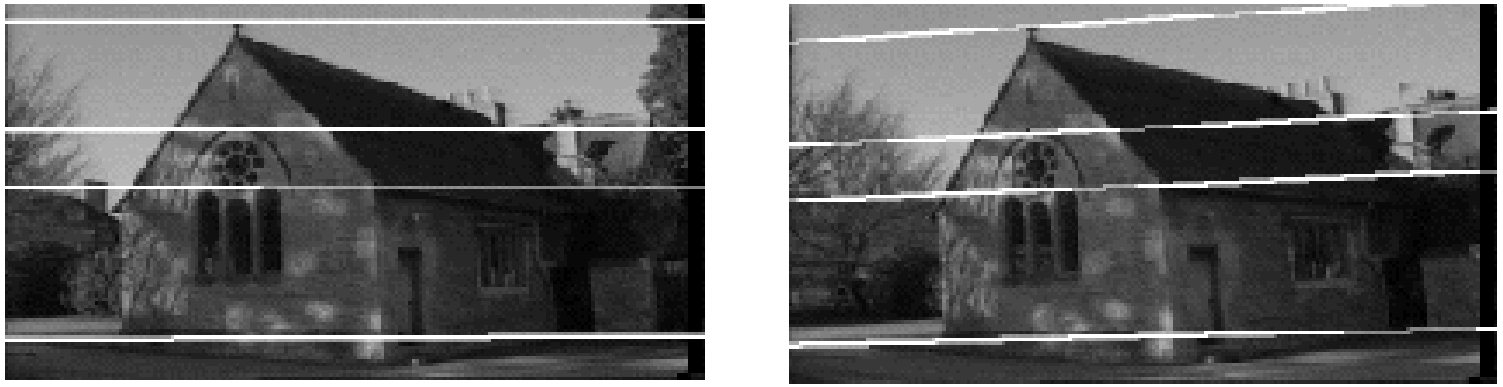
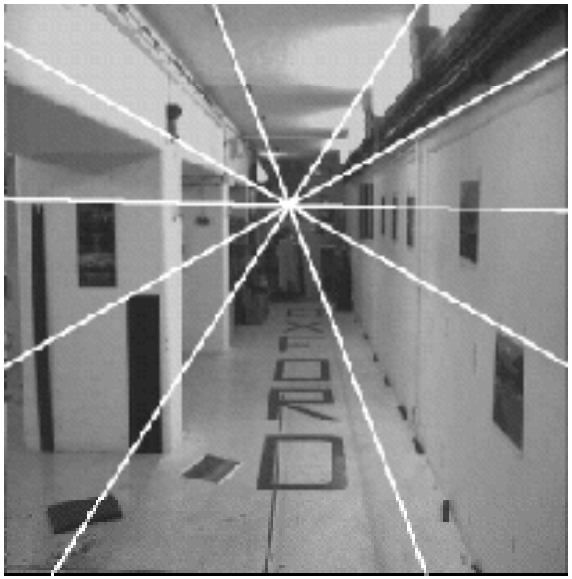
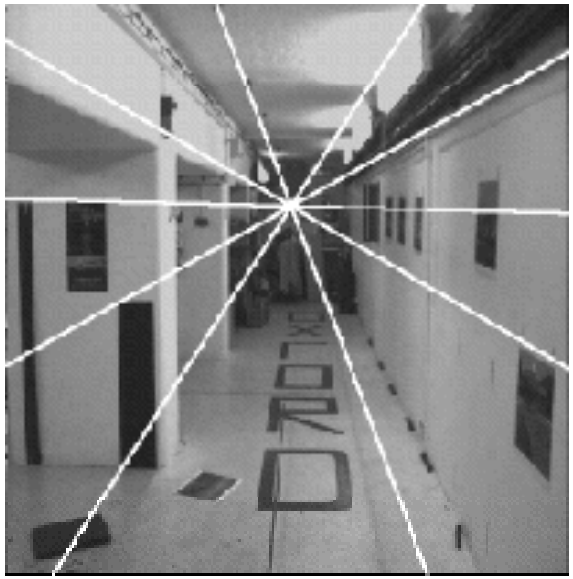
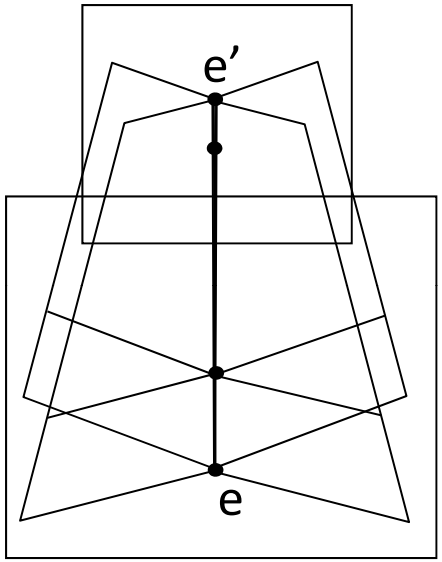


Figure from Hartley & Zisserman

Example: forward motion





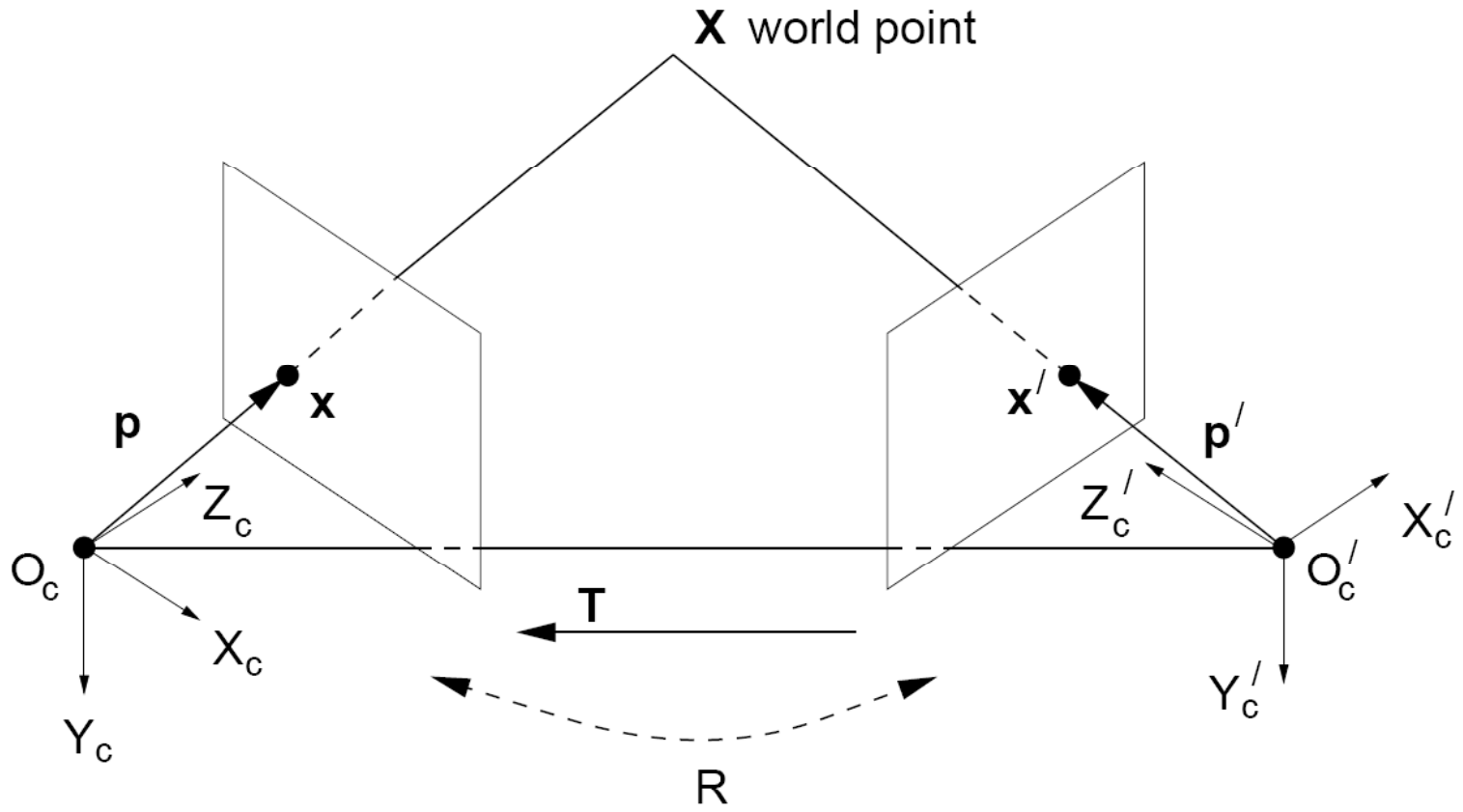


Epipole has same coordinates in both images.
Points move along lines radiating from e: “Focus of expansion”

Figure from Hartley & Zisserman

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

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.

Rotation matrix

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}$$

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

$$R_z(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Express 3d rotation as series of rotations around coordinate axes by angles α, β, γ

Overall rotation is product of these elementary rotations:

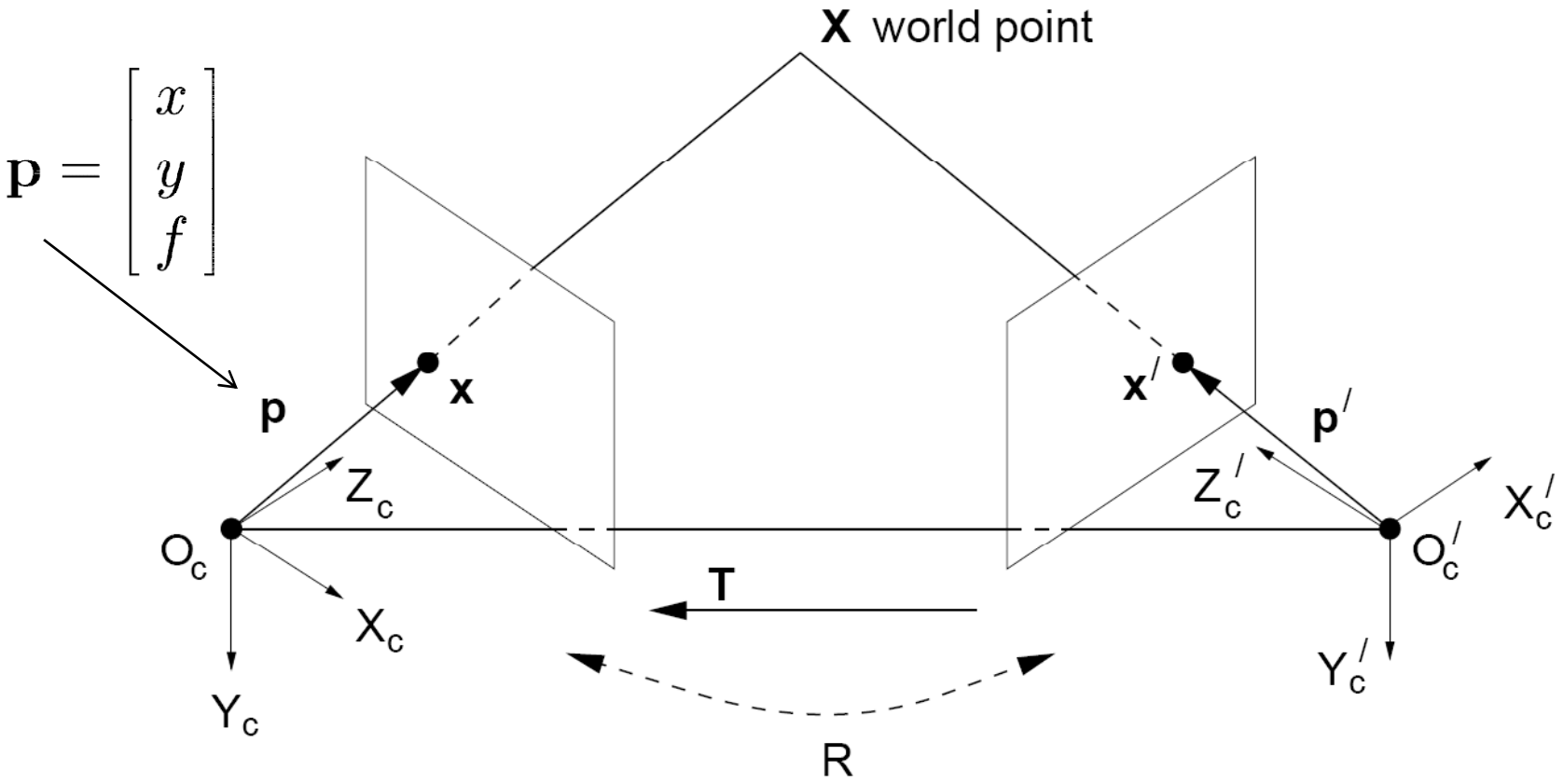
$$\mathbf{R} = \mathbf{R}_x \mathbf{R}_y \mathbf{R}_z$$

3d rigid transformation

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}$$

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

Stereo geometry, with calibrated cameras



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

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

Cross product

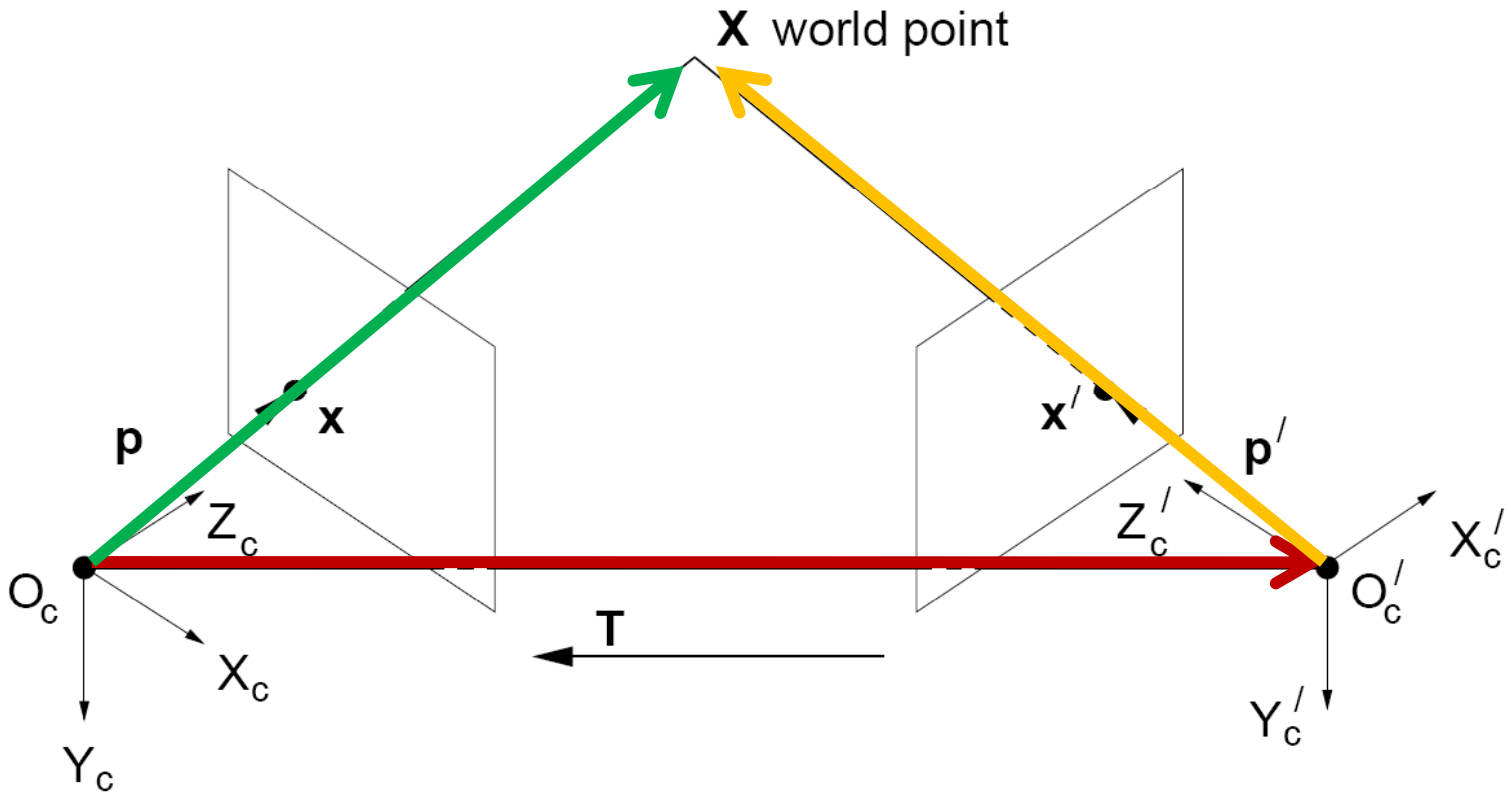
$$\vec{a} \times \vec{b} = \vec{c}$$

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

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.

From geometry to algebra



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

$$\underbrace{\mathbf{T}\times\mathbf{X}'}_{\text{Normal to the plane}} = \mathbf{T}\times\mathbf{R}\mathbf{X}$$

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

$$= 0$$

Matrix form of cross product

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

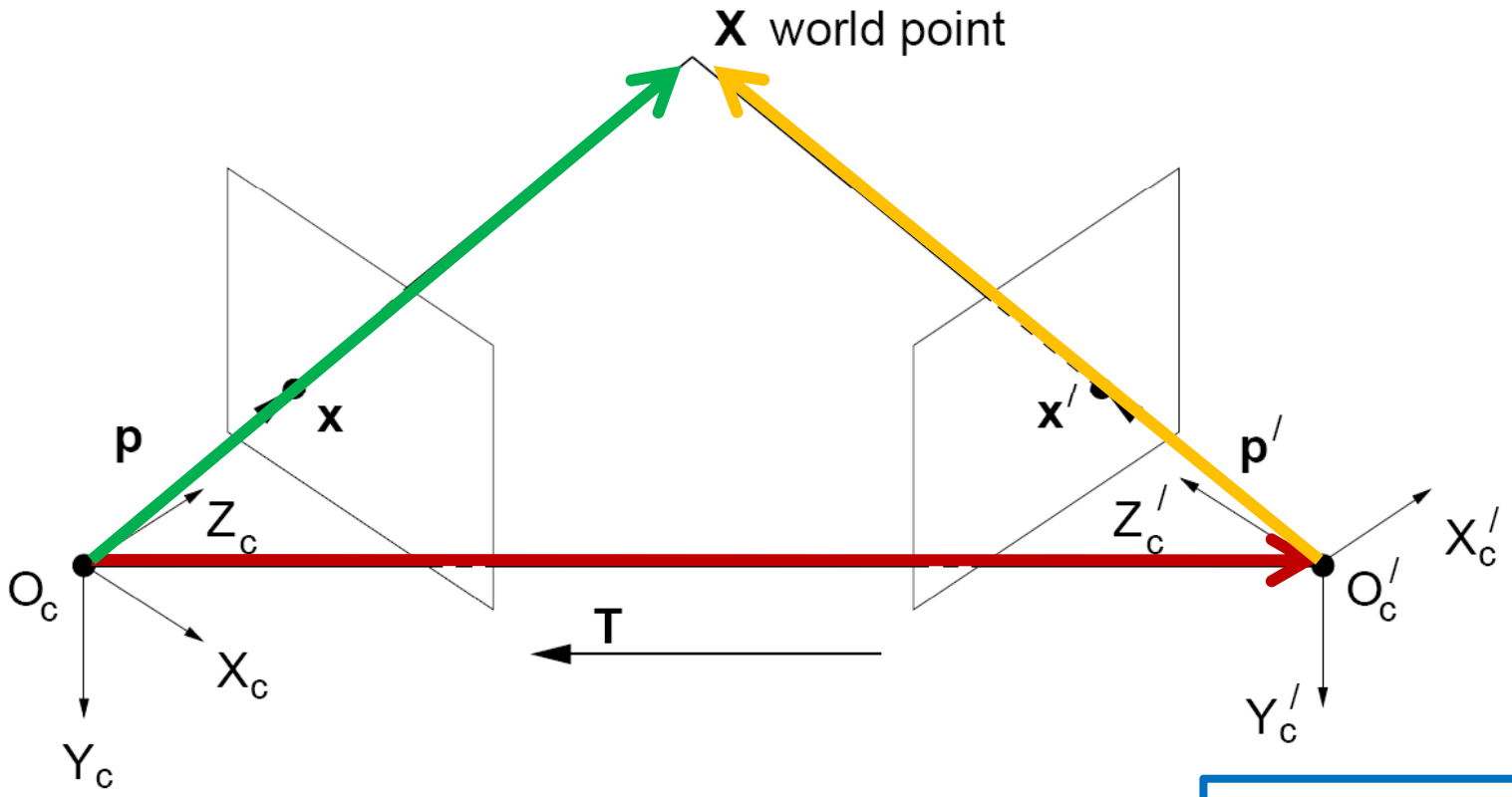
$$\vec{a} \cdot \vec{c} = 0$$
$$\vec{b} \cdot \vec{c} = 0$$

Can be expressed as a matrix multiplication.

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

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

From geometry to algebra



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

$$\underbrace{\mathbf{T}\times\mathbf{X}'}_{\text{Normal to the plane}} = \mathbf{T}\times\mathbf{R}\mathbf{X} + \mathbf{T}\times\mathbf{T}$$

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

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

$$= 0$$

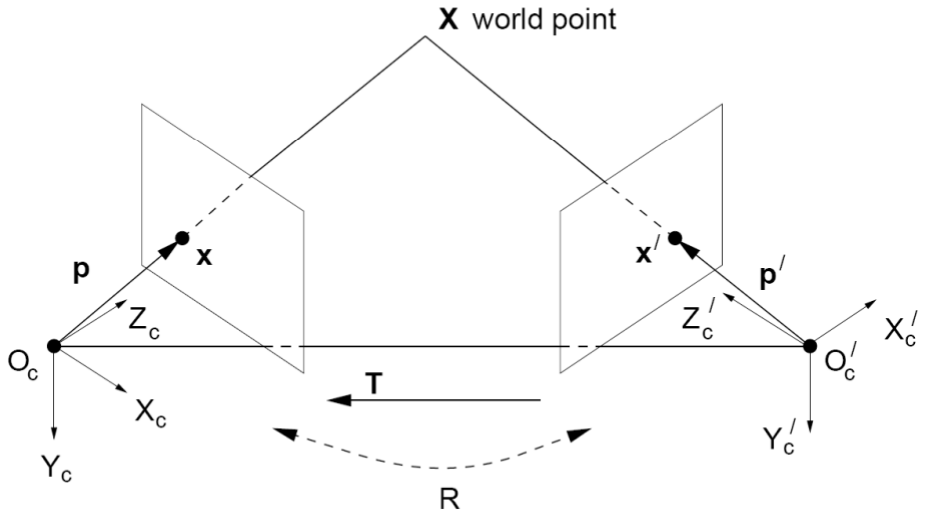
Essential matrix

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

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

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

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



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

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

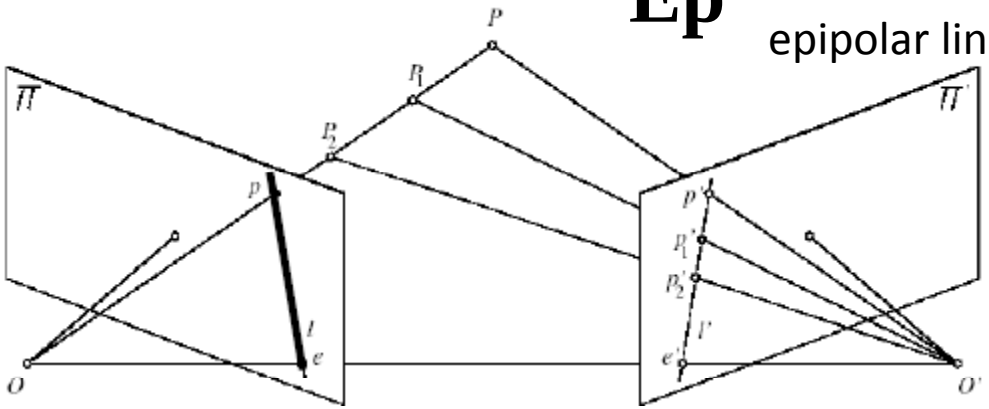
E is called the **essential matrix**, which relates corresponding image points [Longuet-Higgins 1981]

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.

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



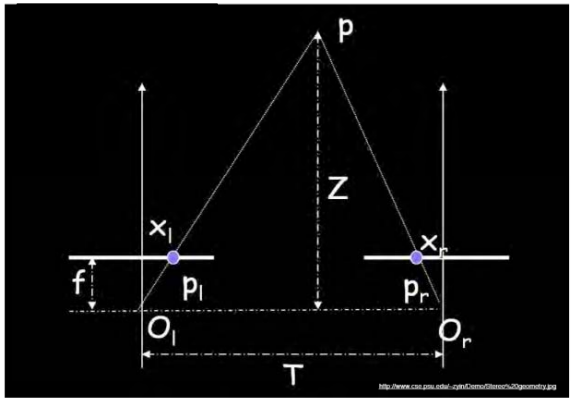
$\mathbf{E}^T\mathbf{p'}$ is the coordinate vector representing the epipolar line associated with point p'

Essential matrix: properties

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

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

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.

image I(x,y)

Disparity map D(x,y)

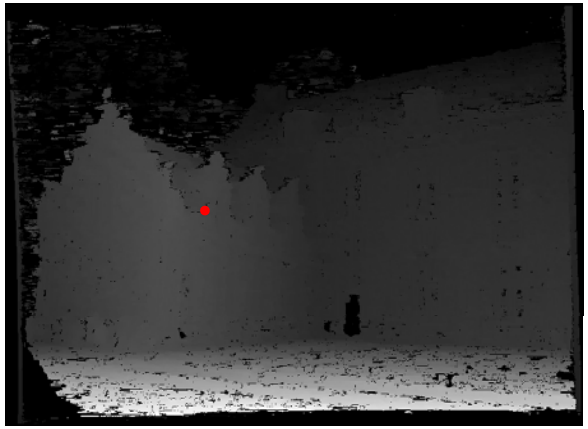


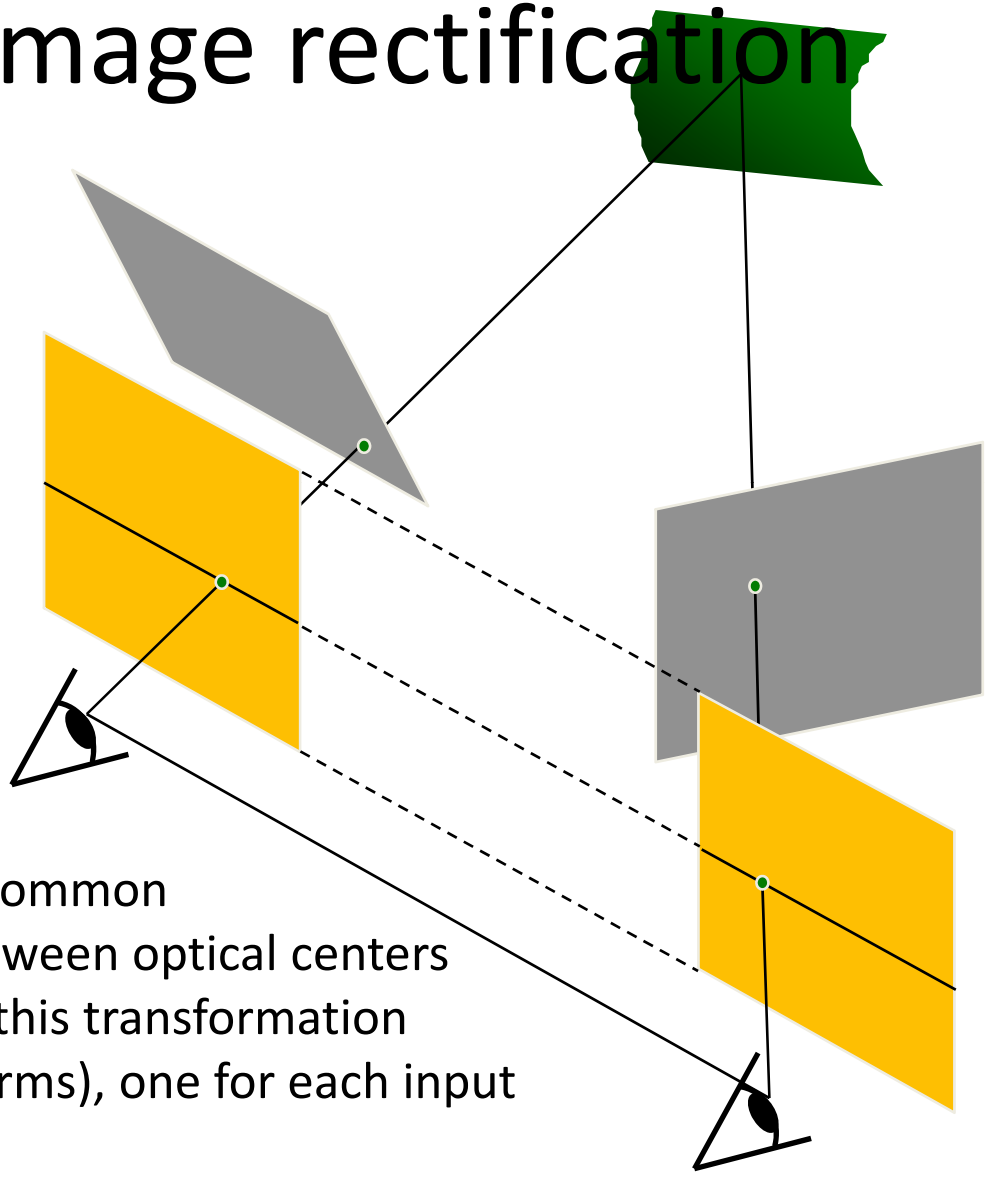
image I´(x´,y´)

$(x´,y´)=(x+D(x,y),y)$

What about when cameras' optical axes are not parallel?

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

Source: Alyosha Efros

Stereo reconstruction: main steps

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

Correspondence problem

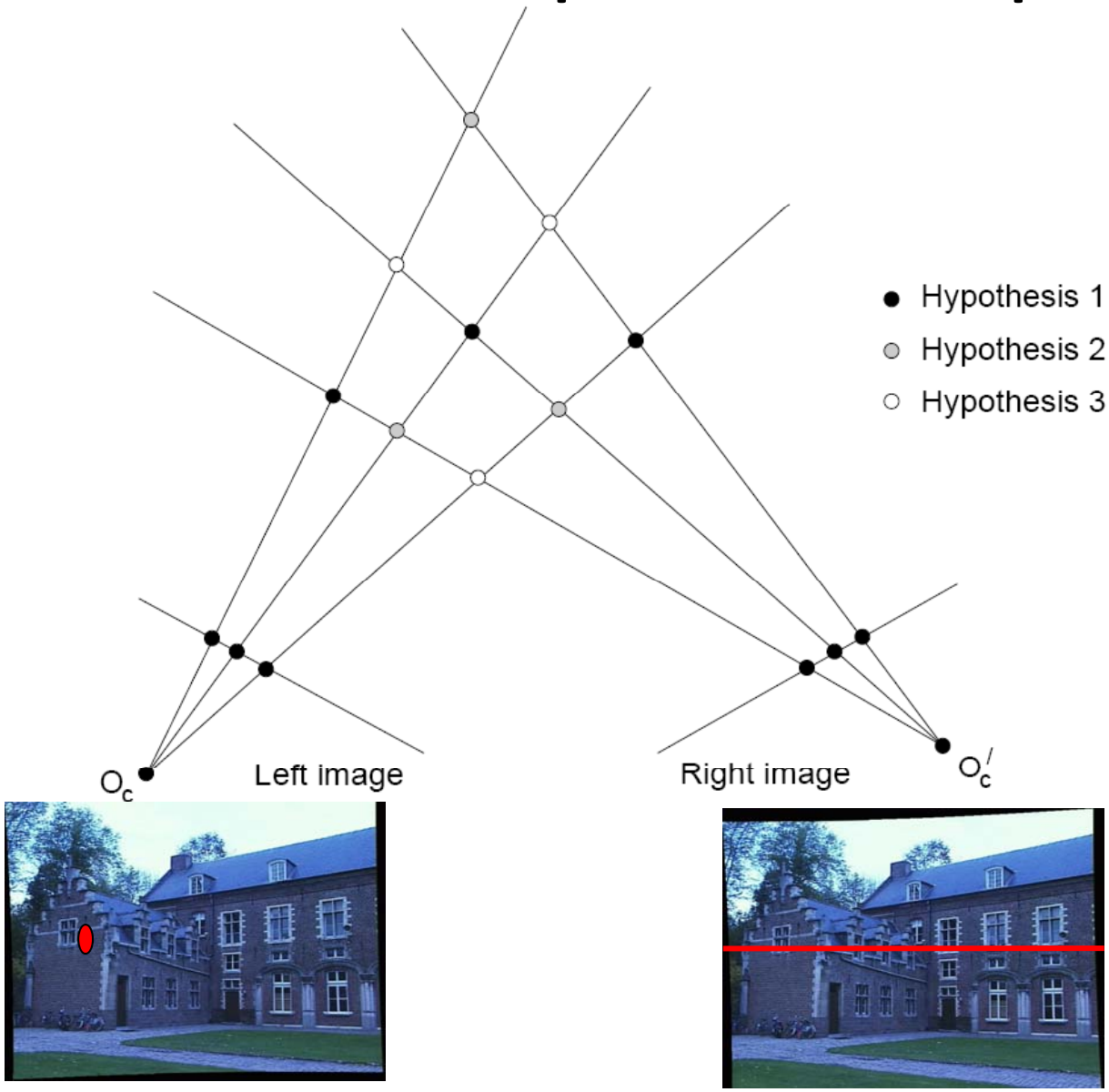


Figure from Gee & Cipolla 1999

Multiple match hypotheses satisfy epipolar constraint, but which is correct?

Correspondence problem

- Beyond the hard constraint of epipolar geometry, there are "soft" constraints to help identify corresponding points
  - Similarity
  - Uniqueness
  - Ordering
  - Disparity gradient

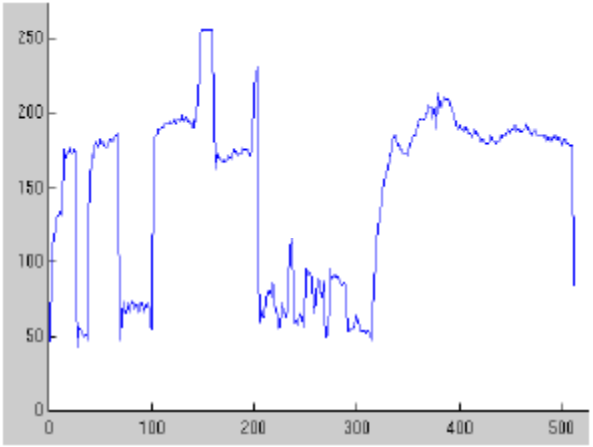
- To find matches in the image pair, we will assume
  - Most scene points visible from both views
  - Image regions for the matches are similar in appearance

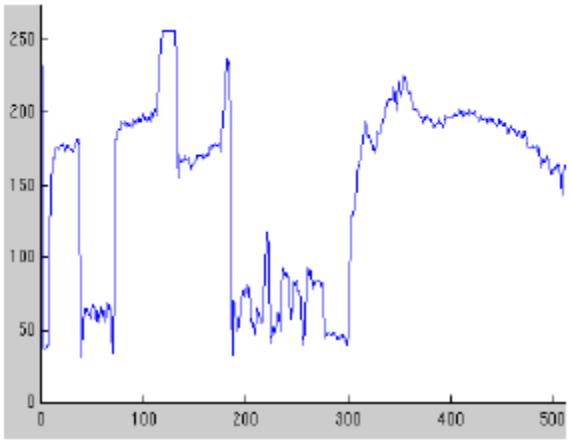
Correspondence problem

Parallel camera example – epipolar lines are corresponding rasters

Source: Andrew Zisserman

Intensity profiles

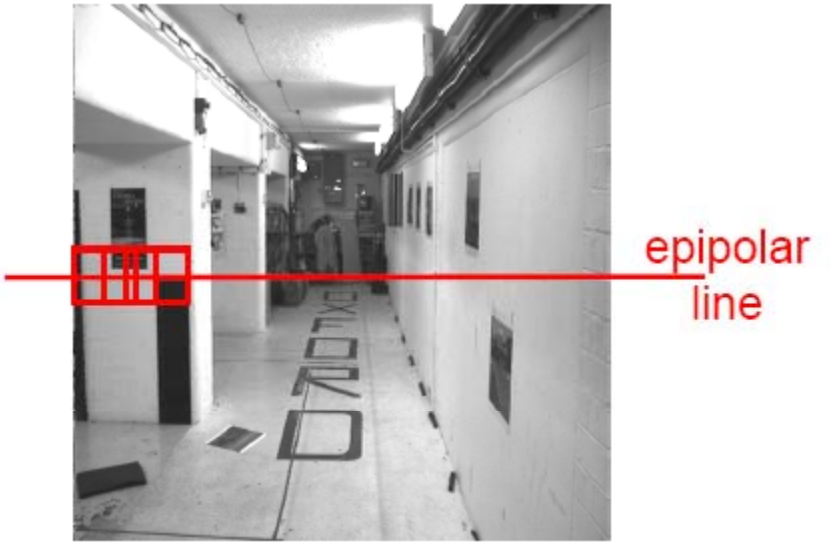




- Clear correspondence between intensities, but also noise and ambiguity

Source: Andrew Zisserman

Correspondence problem



Neighborhood of corresponding points are similar in intensity patterns.

Source: Andrew Zisserman

Normalized cross correlation

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

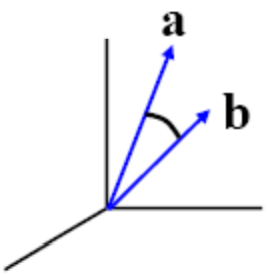
$$\text{NCC} = \frac{\sum_i \sum_j A(i,j)B(i,j)}{\sqrt{\sum_i \sum_j A(i,j)^2}\sqrt{\sum_i \sum_j B(i,j)^2}}$$

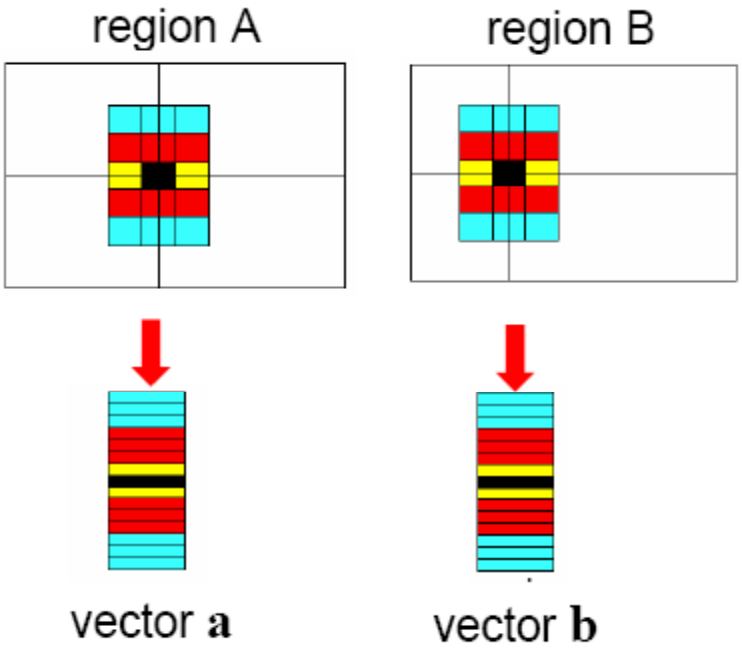
Write regions as vectors

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

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

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





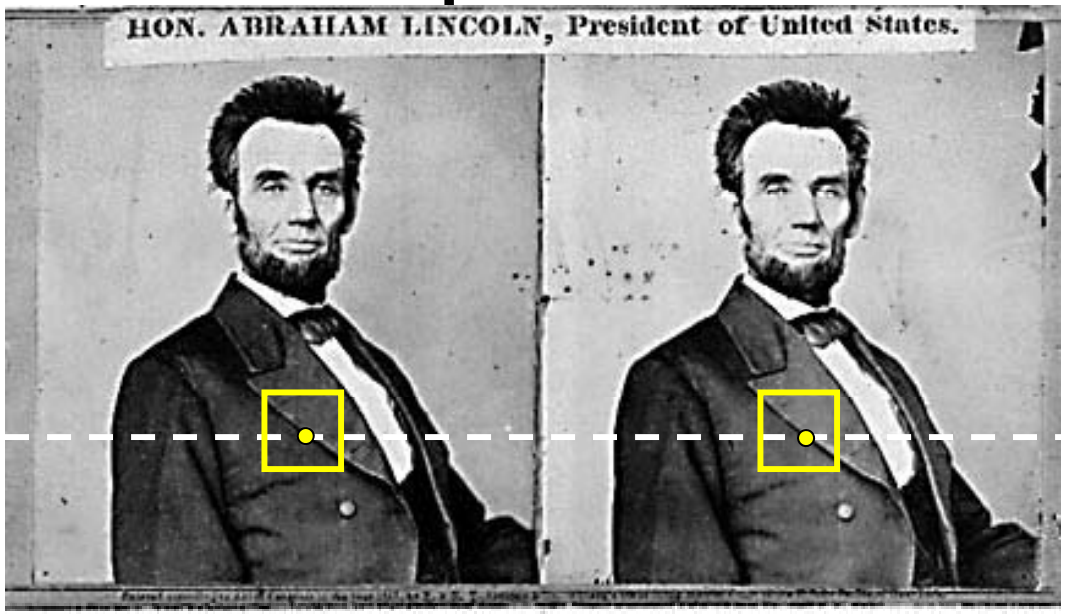
Source: Andrew Zisserman

Correlation-based window matching

left image band (x)

Source: A

Dense correspondence search



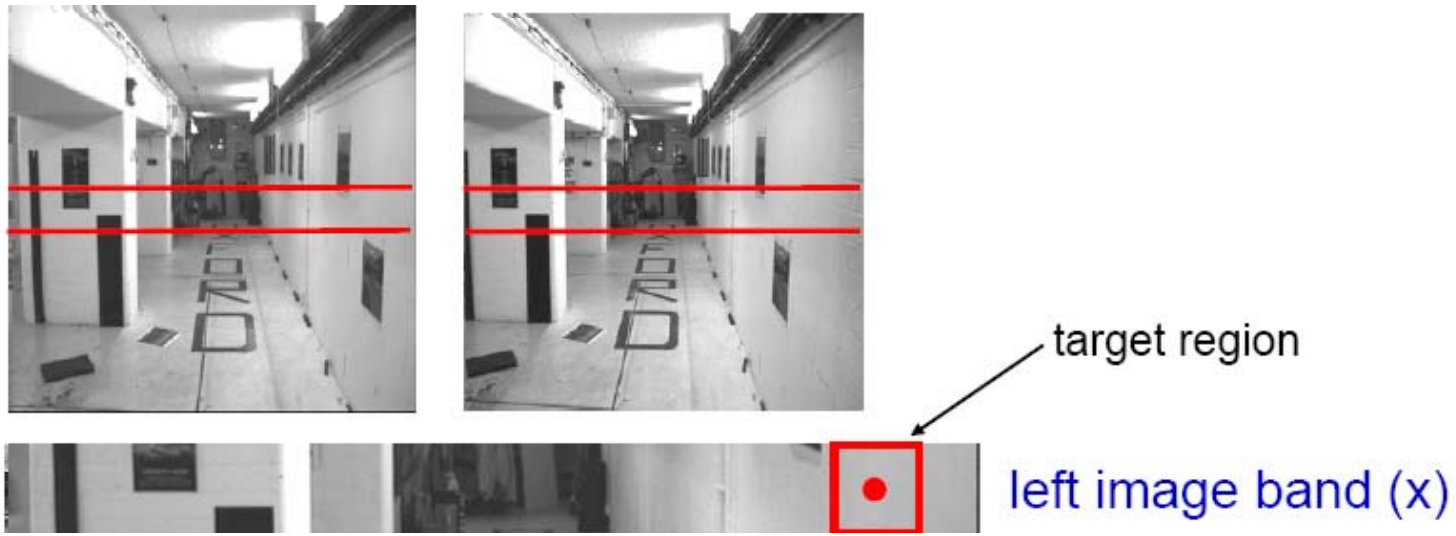
For each epipolar line

For each pixel / window in the left image

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

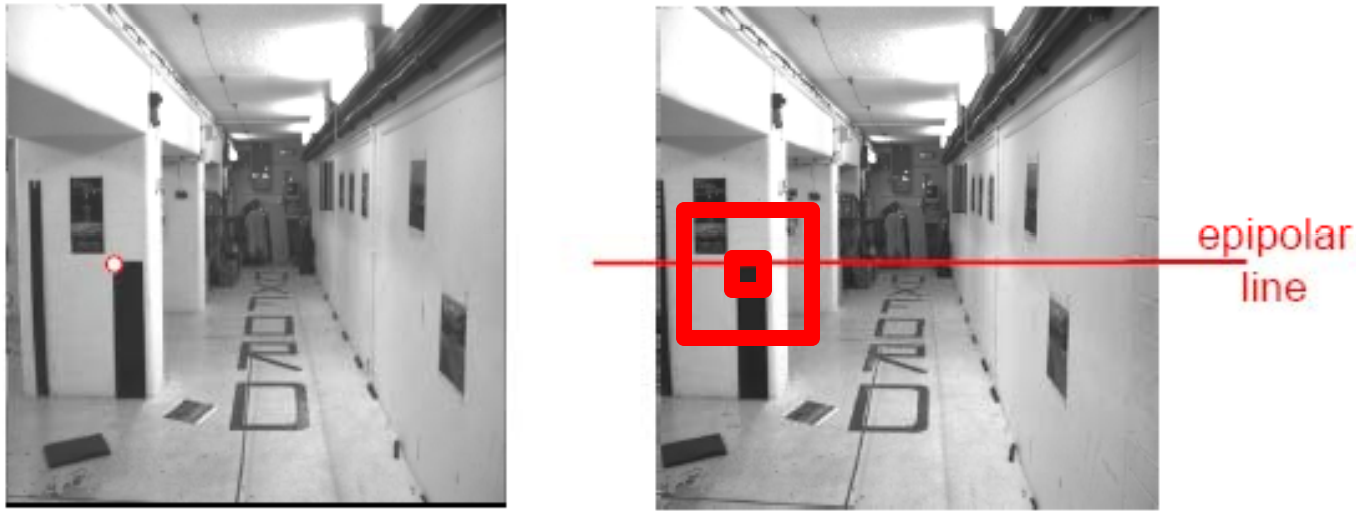
Adapted from Li Zhang

Textureless regions



Source: Andrew Zisserman

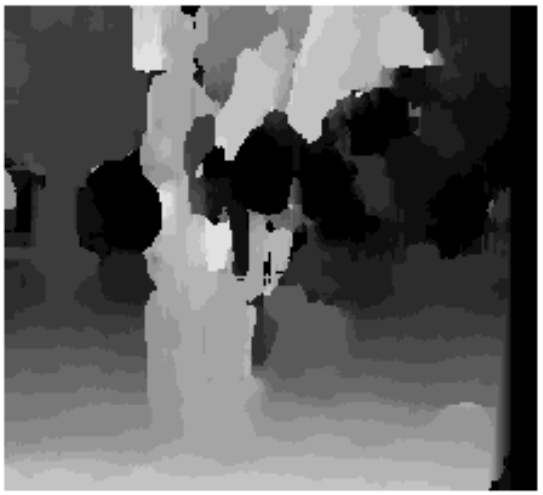
Effect of window size



Source: Andrew Zisserman

Effect of window size

W = 3

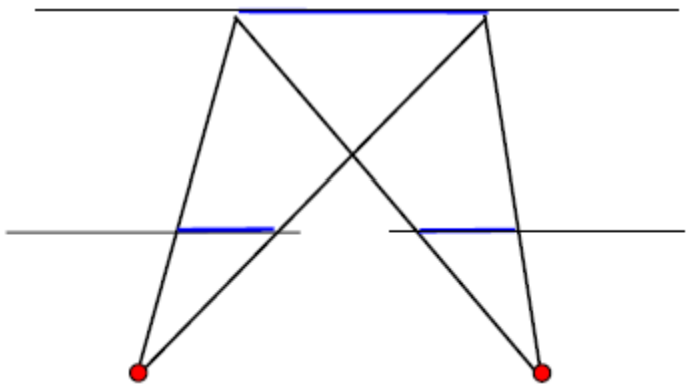


W = 20

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

Figures from Li Zhang

Foreshortening effects

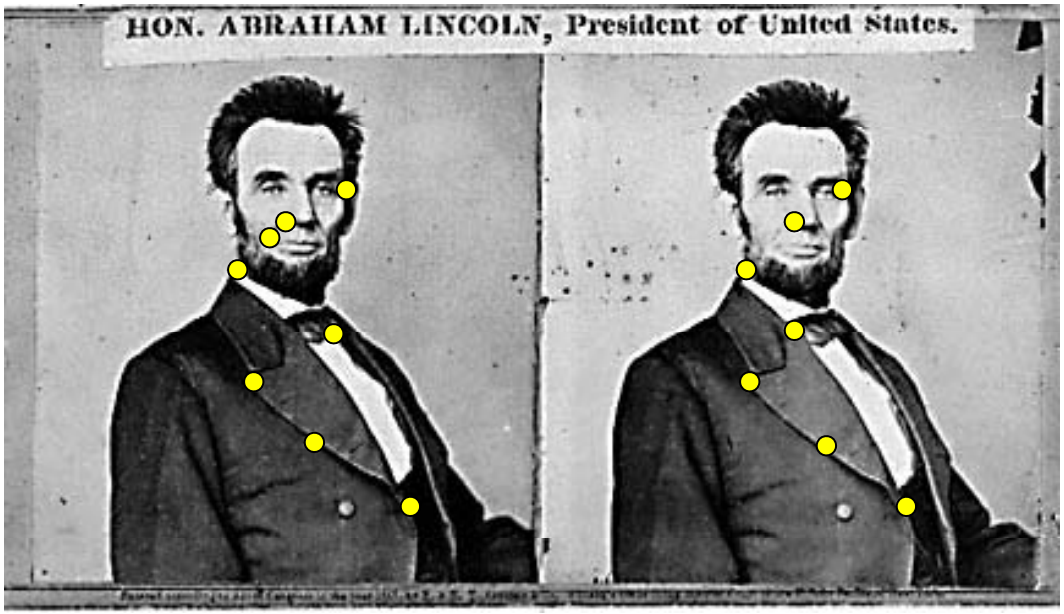


fronto-parallel surface

imaged length the same

Source: Andrew Zisserman

Sparse correspondence search



- Restrict search to sparse set of detected features
- Rather than pixel values (or lists of pixel values) use *feature descriptor* and an associated *feature distance*
- Still narrow search further by epipolar geometry

Uniqueness

- For opaque objects, up to one match in right image for every point in left image

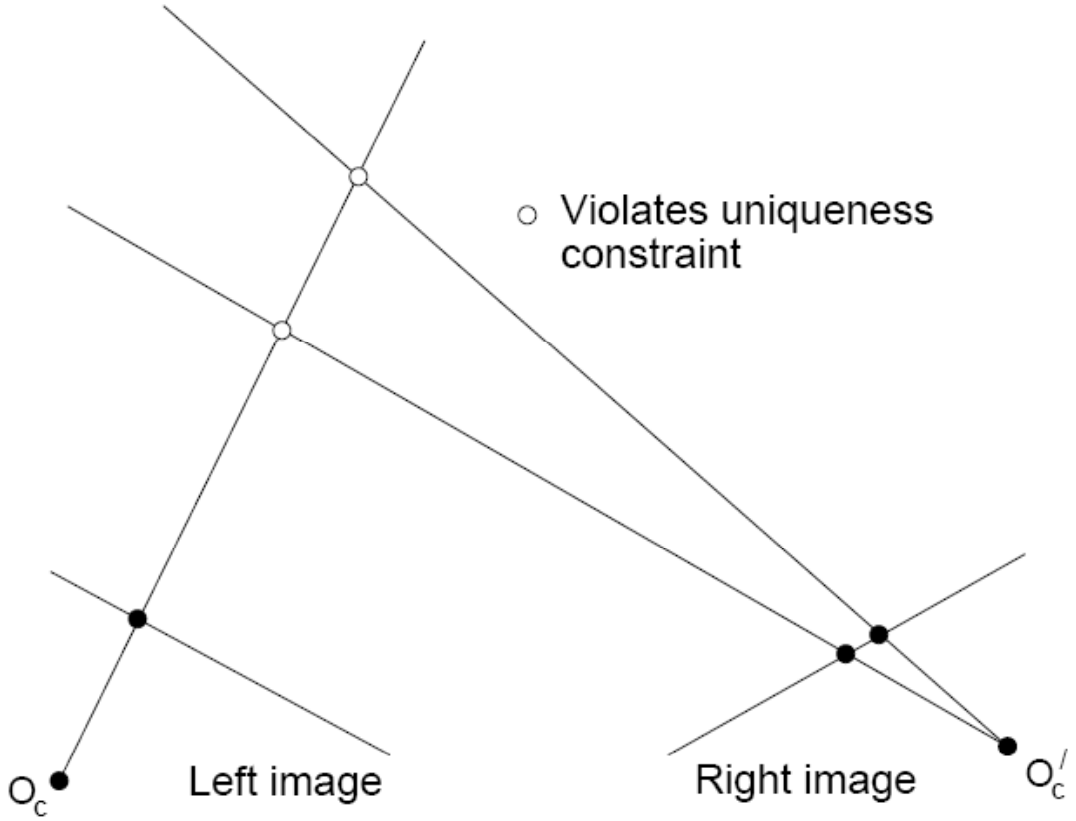


Figure from Gee & Cipolla 1999

Ordering constraint

- Points on **same surface** (opaque object) will be in same order in both views

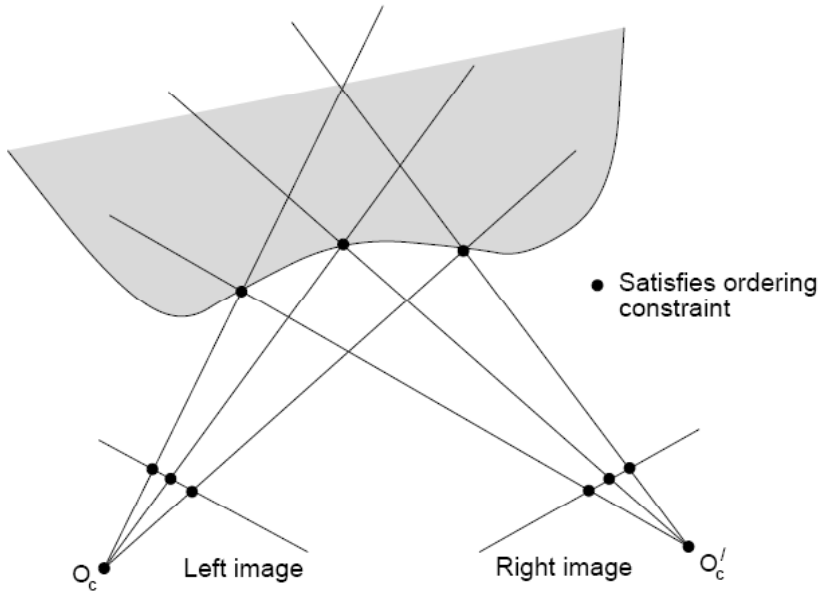
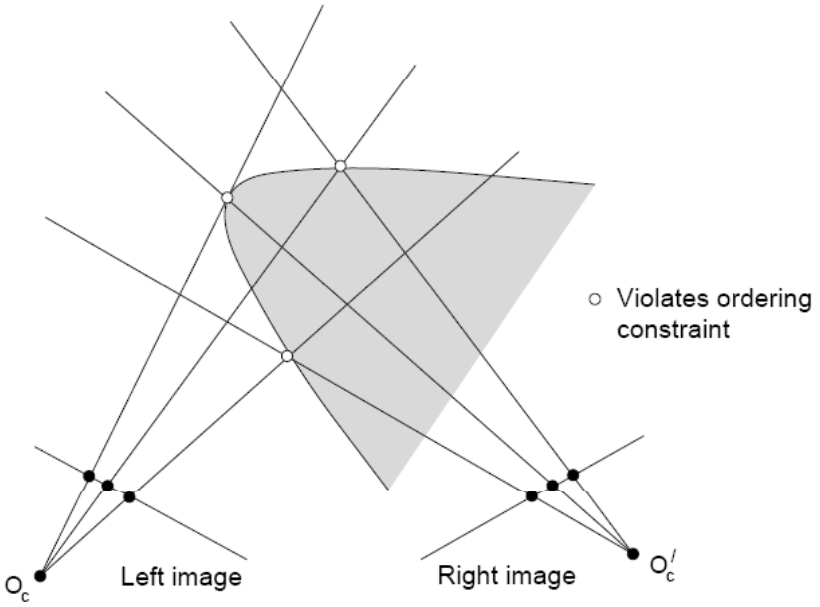
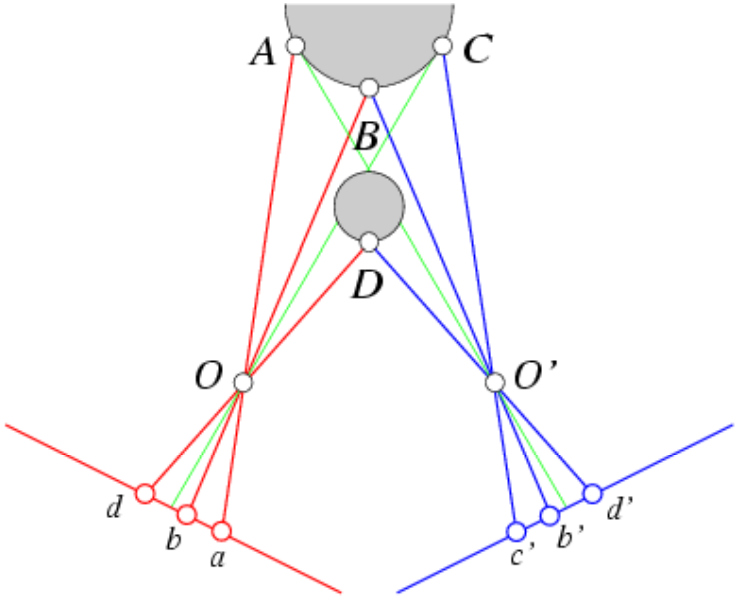


Figure from Gee & Cipolla 1999

Ordering constraint

- Won't always hold, e.g. consider transparent object, or an occluding surface

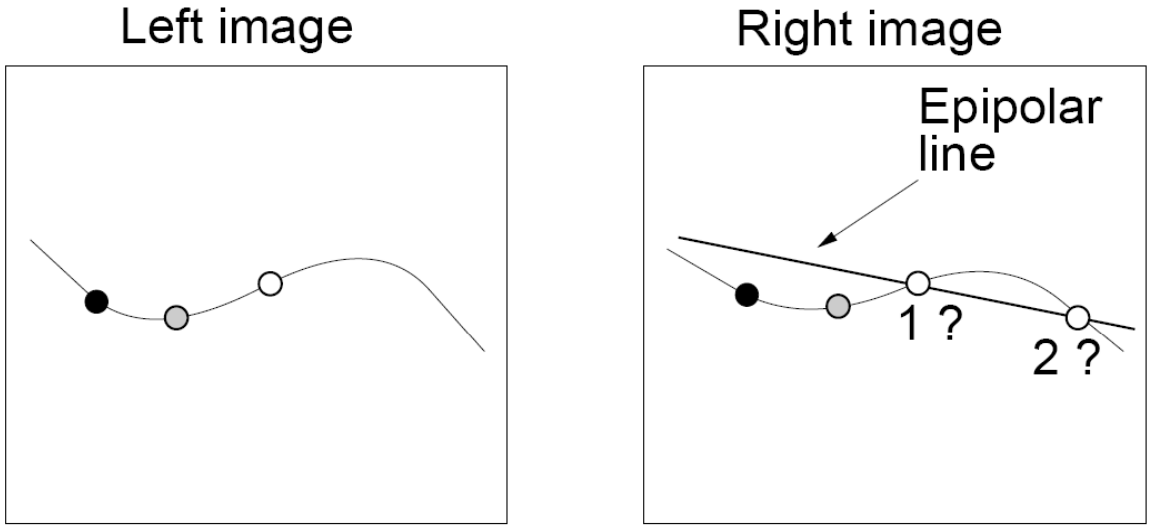




Figures from Forsyth & Ponce

Disparity gradient

- Assume piecewise continuous surface, so want disparity estimates to be locally smooth

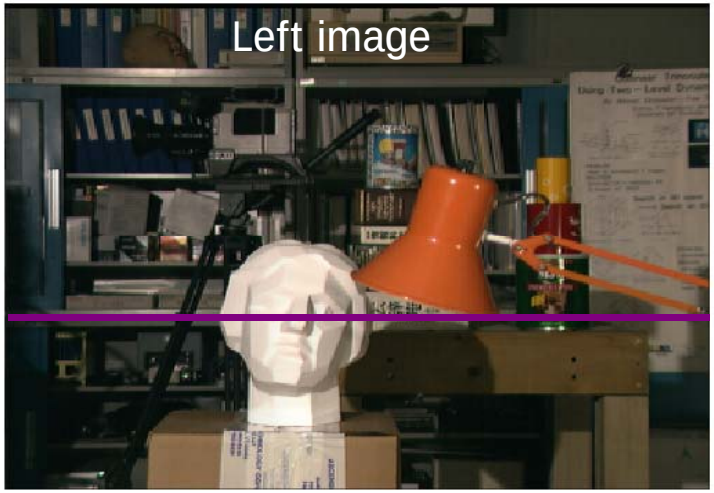


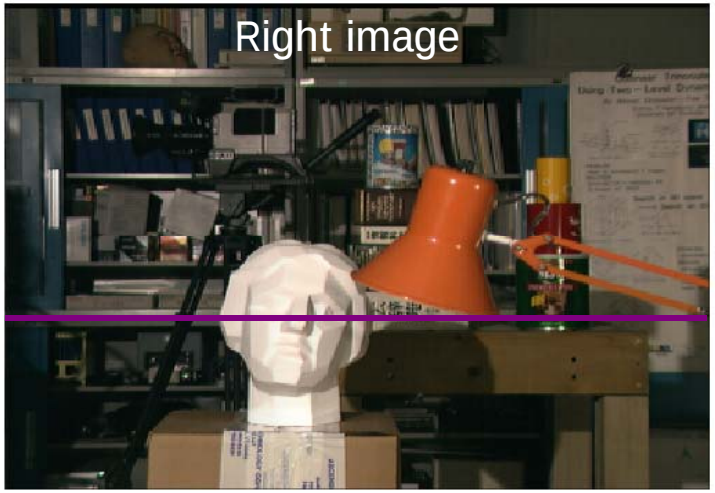
Given matches ● and ◌, point ○ in the left image must match point 1 in the right image. Point 2 would exceed the disparity gradient limit.

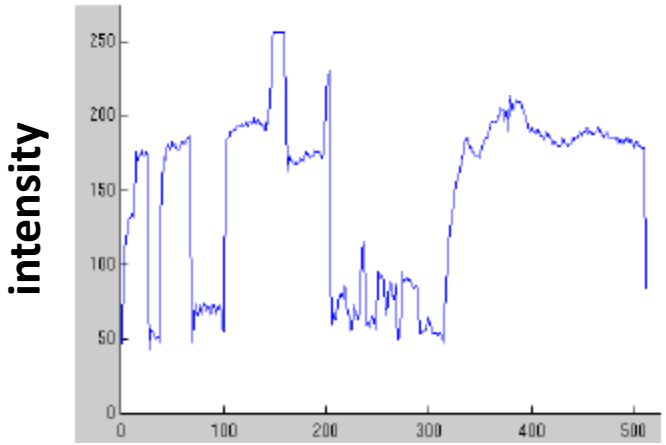
Figure from Gee & Cipolla 1999

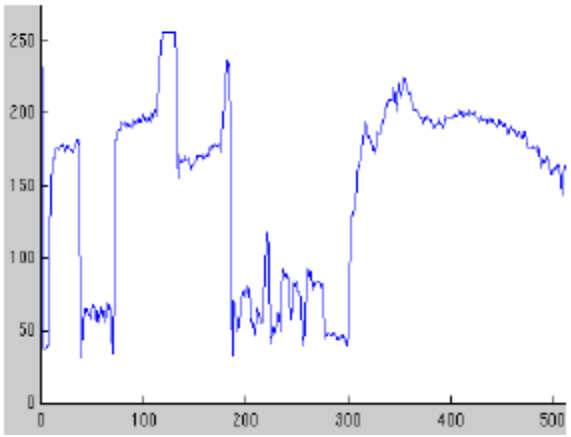
Scanline stereo

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

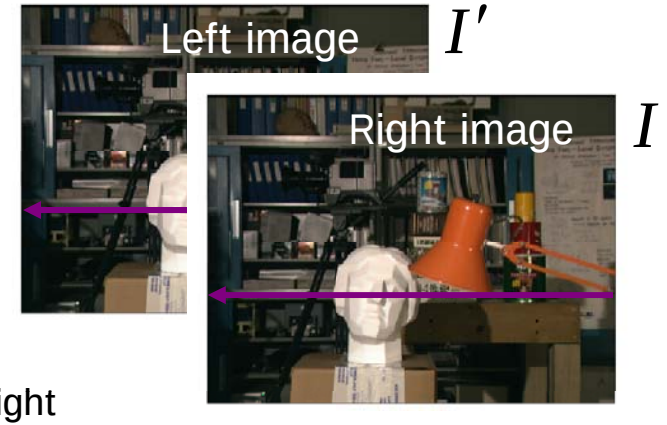


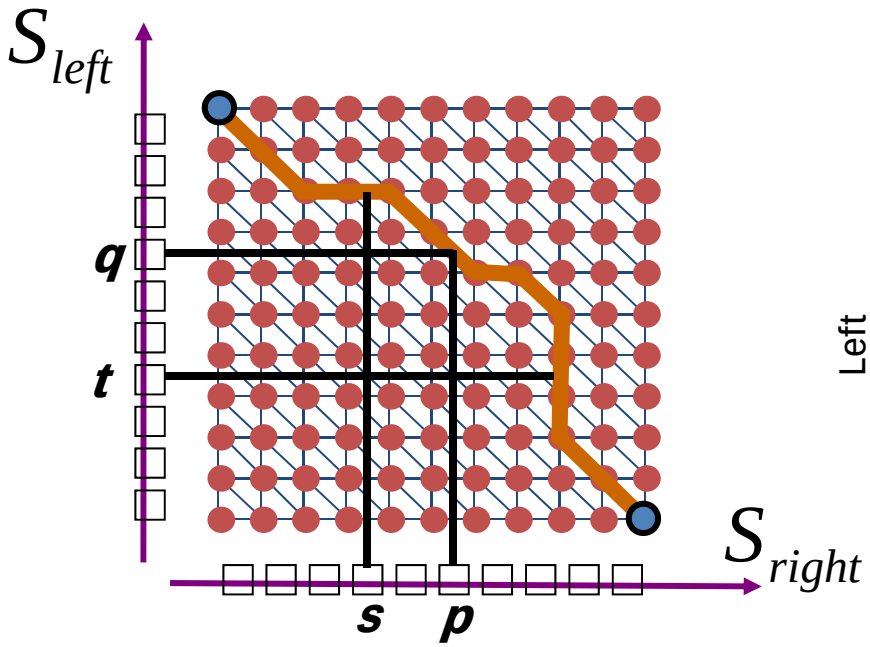


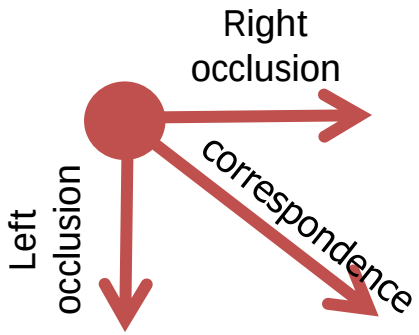




“Shortest paths” for scan-line stereo







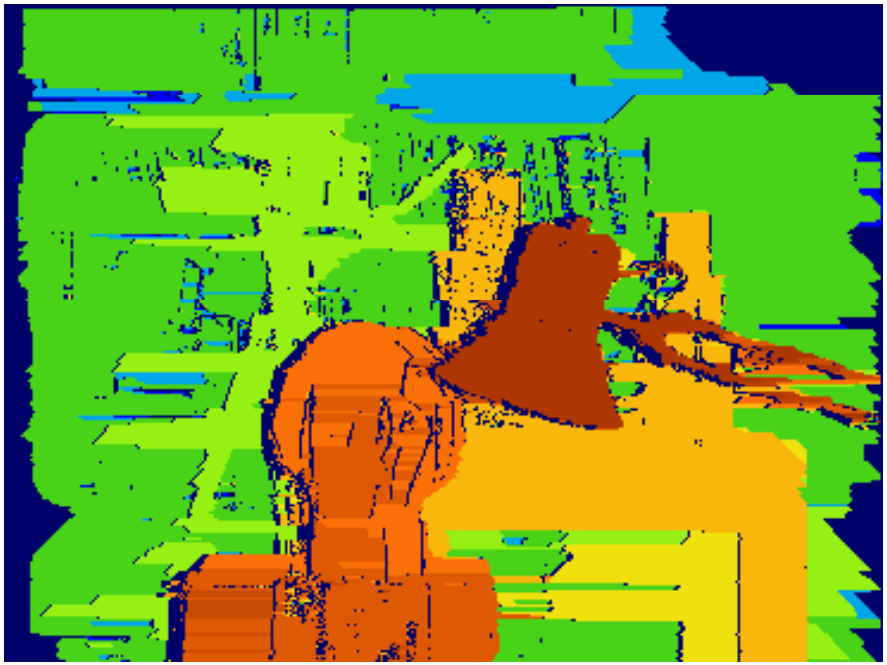
Can be implemented with dynamic programming

Ohta & Kanade ’85, Cox et al. ‘96

Slide credit: Y. Boykov

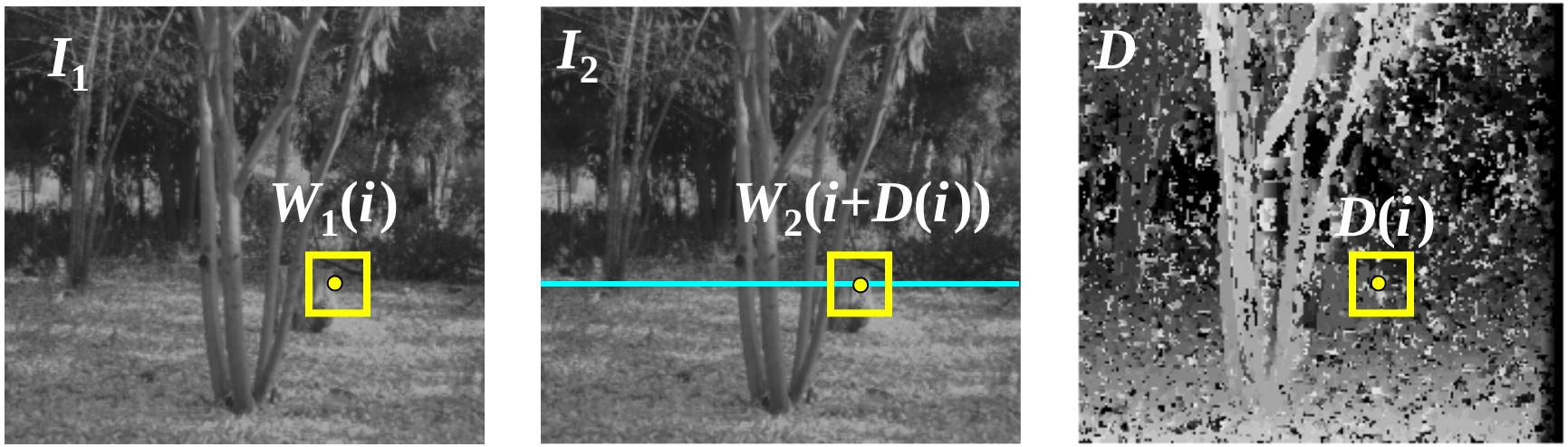
Coherent stereo on 2D grid

- Scanline stereo generates streaking artifacts



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

As energy minimization…

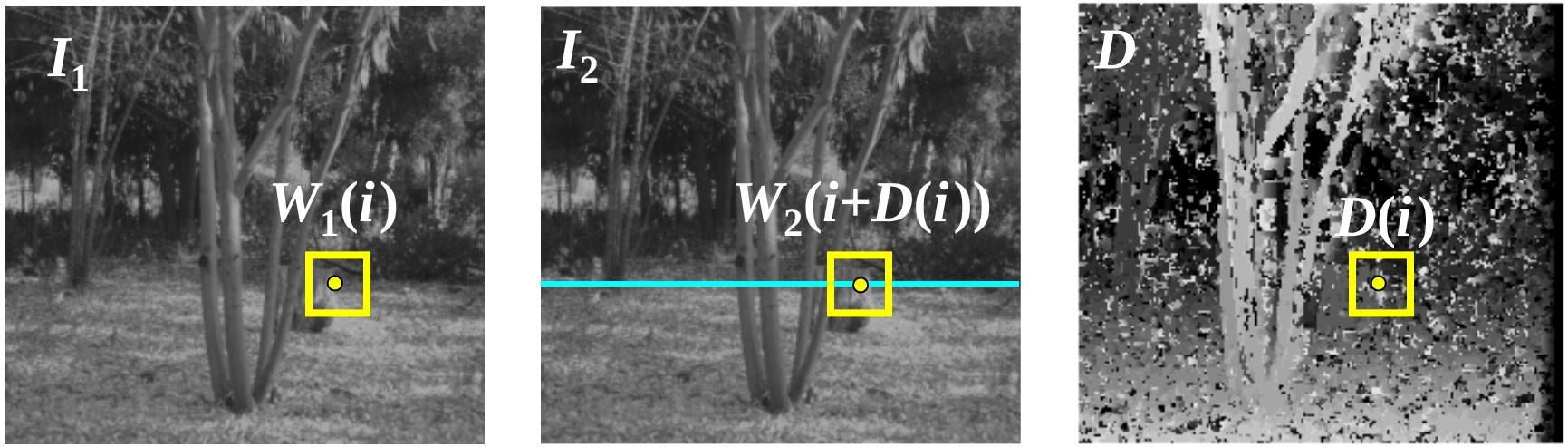


$$E = \alpha E_{\text{data}}(I_1, I_2, D) + \beta E_{\text{smooth}}(D)$$

$$E_{\text{data}} = \sum_i \left(W_1(i) - W_2(i + D(i))\right)^2$$

$$E_{\text{smooth}} = \sum_{\text{neighbors } i,j} \rho\left(D(i) - D(j)\right)$$

As energy minimization…



$$E = \alpha E_{\text{data}}(I_1, I_2, D) + \beta E_{\text{smooth}}(D)$$

$$E_{\text{data}} = \sum_i \left(W_1(i) - W_2(i + D(i))\right)^2$$

$$E_{\text{smooth}} = \sum_{\text{neighbors } i,j} \rho\left(D(i) - D(j)\right)$$

- Energy functions of this form can be minimized using *graph cuts*

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

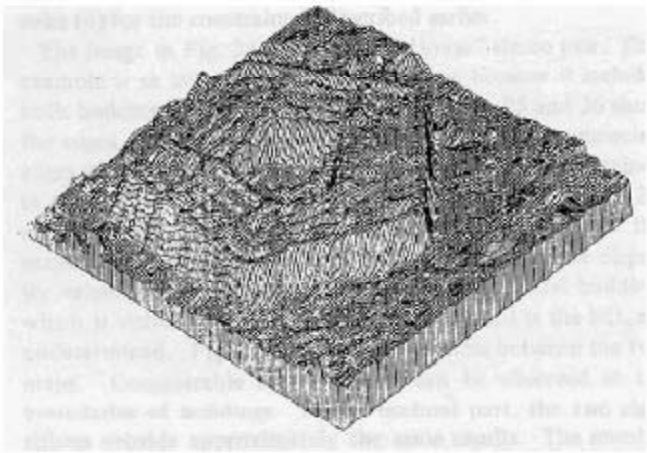
Source: Steve Seitz

Examples…

left image

right image

range map



left image

right image

depth map

intensity = depth

Z-keying for virtual reality

- Merge synthetic and real images given depth maps

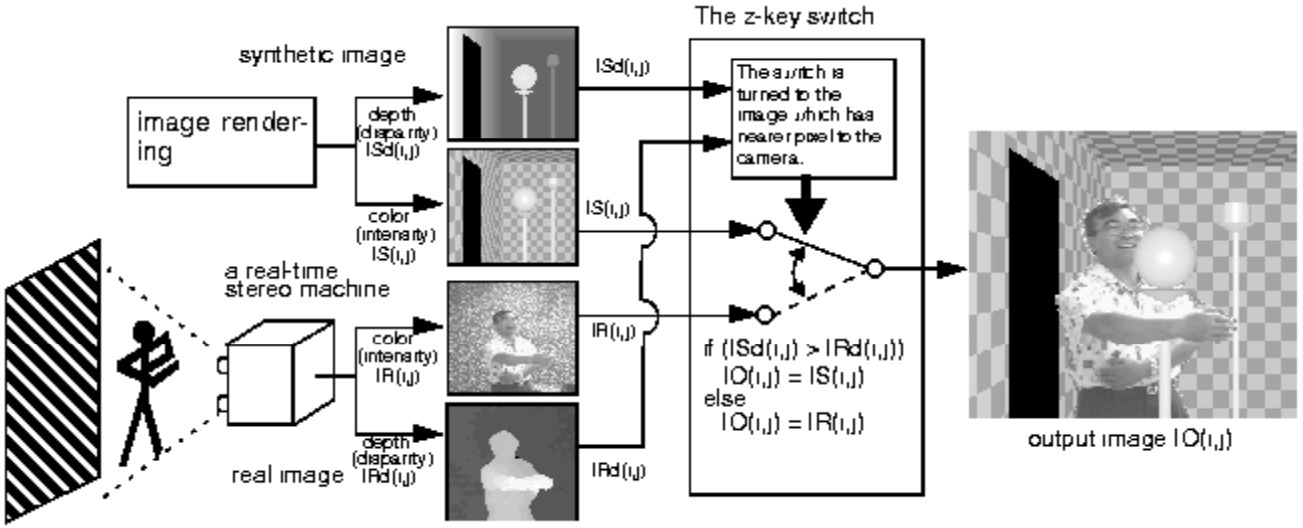
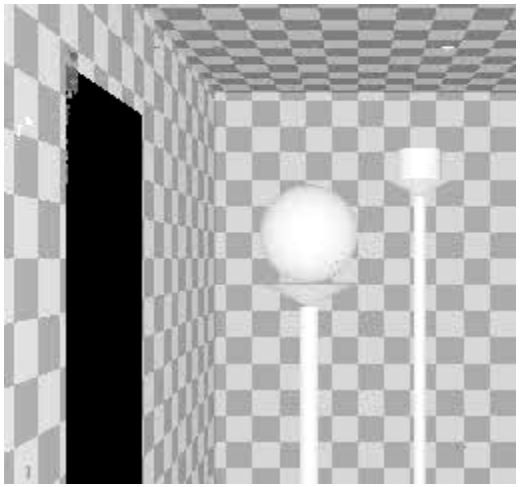


Figure 1: A schema of the z-key method

Kanade et al., CMU, 1995

Z-keying for virtual reality



Kanade et al., CMU, 1995

http://www.cs.cmu.edu/afs/cs/project/stereo-machine/www/z-key.html

Virtual viewpoint video

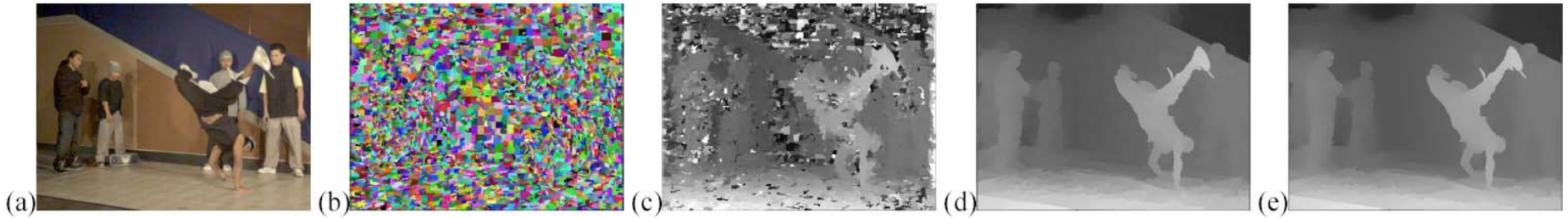


Figure 6: *Sample results from stereo reconstruction stage: (a) input color image; (b) color-based segmentation; (c) initial disparity estimates* $\hat{d}_{ij}$*; (d) refined disparity estimates; (e) smoothed disparity estimates* $d_i(x)$.

C. Zitnick et al, High-quality video view interpolation using a layered representation, SIGGRAPH 2004.

Virtual viewpoint video

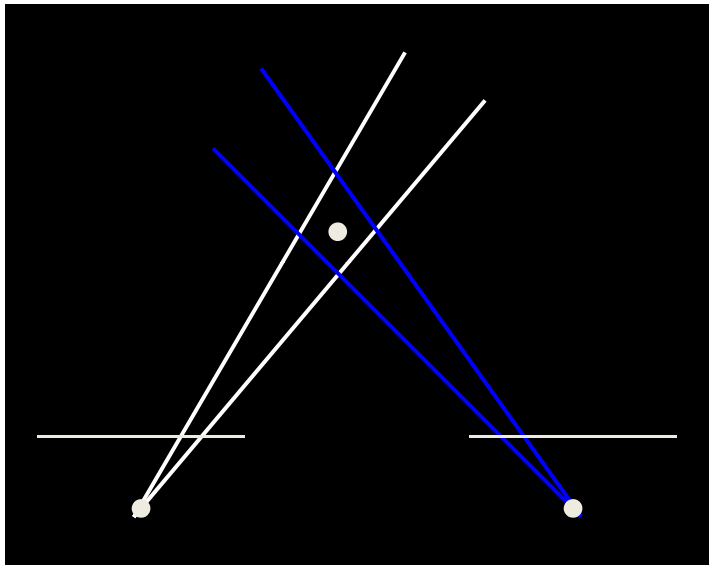
http://research.microsoft.com/IVM/VVV/

Multibaseline Stereo

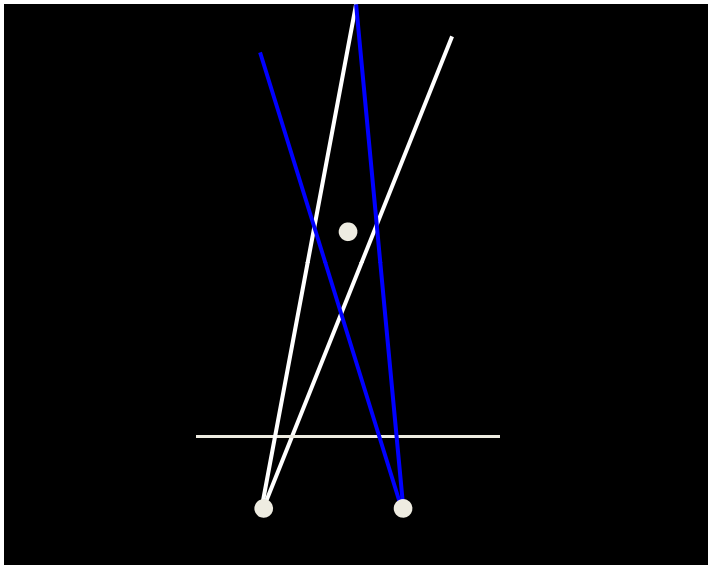
- Basic Approach
  - Choose a reference view
  - Use your favorite stereo algorithm BUT
    - replace two-view SSD with SSD over all baselines

- Limitations
  - Must choose a reference view
  - Visibility: select which frames to match
    [Kang, Szeliski, Chai, CVPR'01]

Choosing the Baseline



Large Baseline

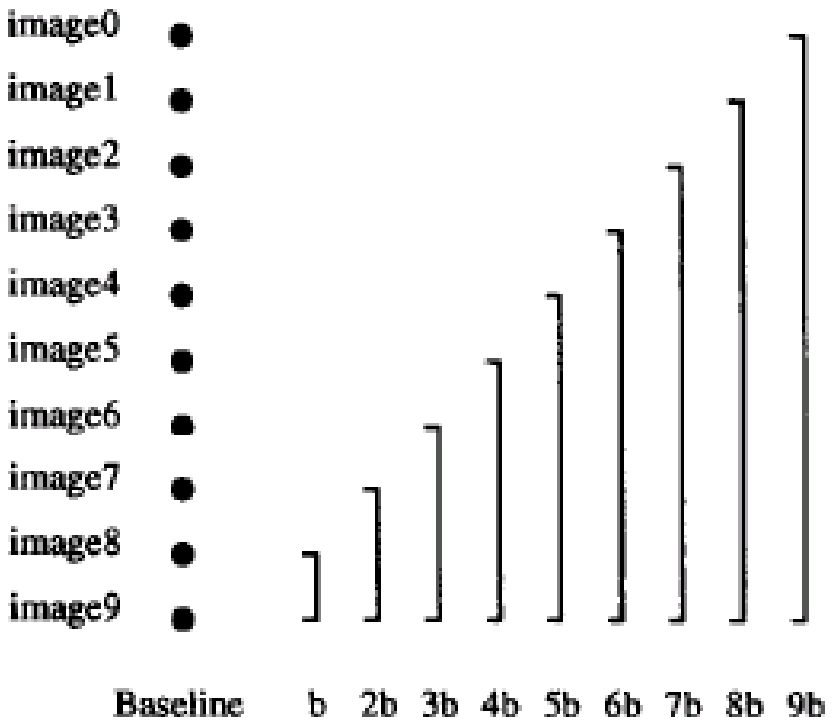


Small Baseline

- What's the optimal baseline?
  - Too small: large depth error
  - Too large: difficult search problem

Effect of Baseline on Estimation

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



Szeliski

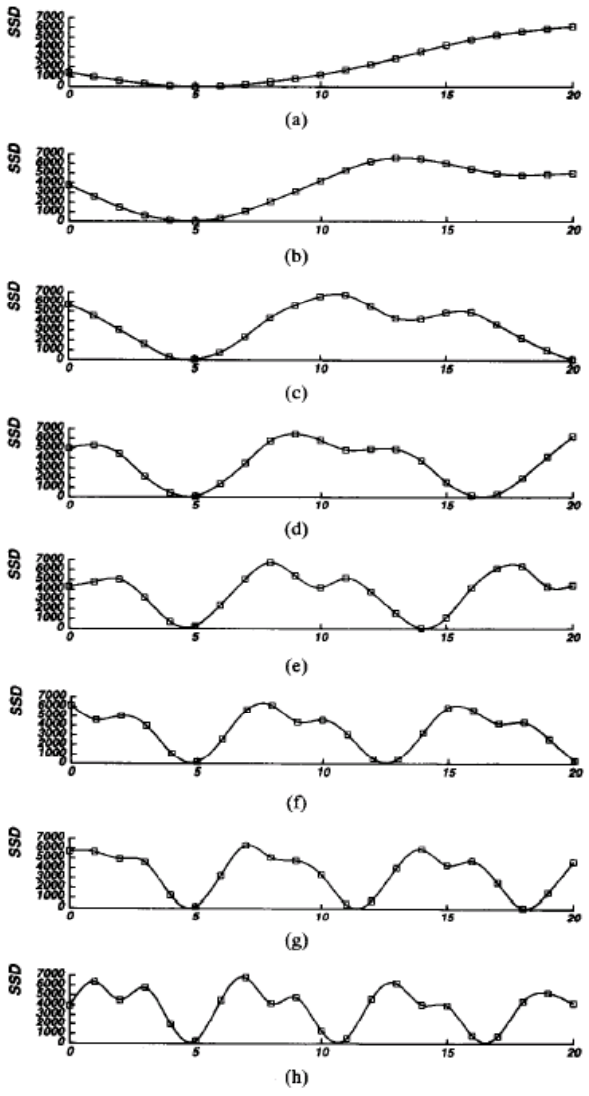


Fig. 5. SSD values versus inverse distance: (a) $B = b$; (b) $B = 2b$; (c) $B = 3b$; (d) $B = 4b$; (e) $B = 5b$; (f) $B = 6b$; (g) $B = 7b$; (h) $B = 8b$. The horizontal axis is normalized such that $8bF = 1$.

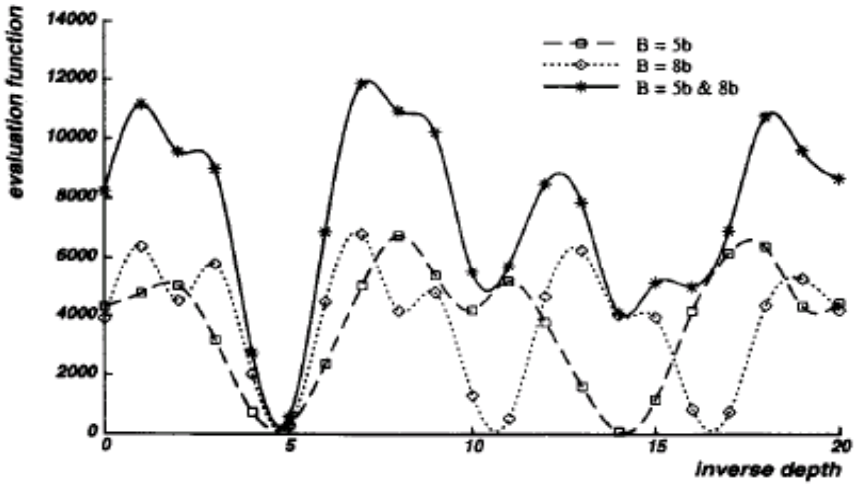


Fig. 6. Combining two stereo pairs with different baselines.

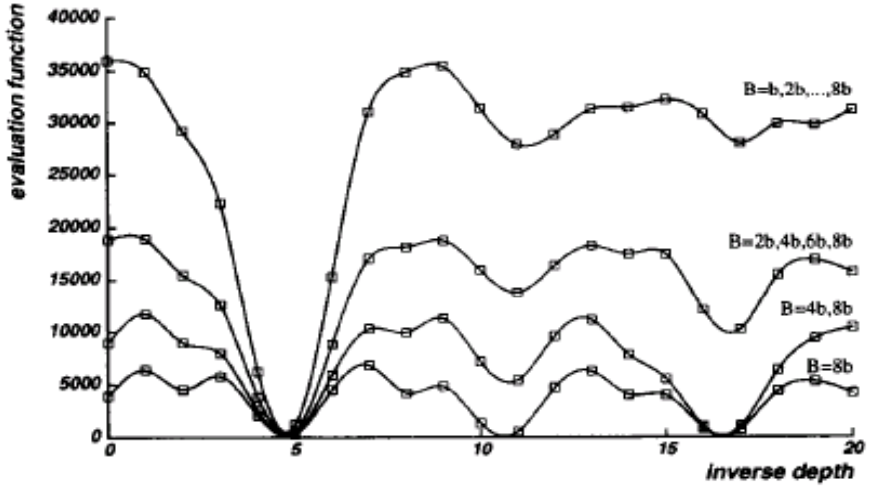
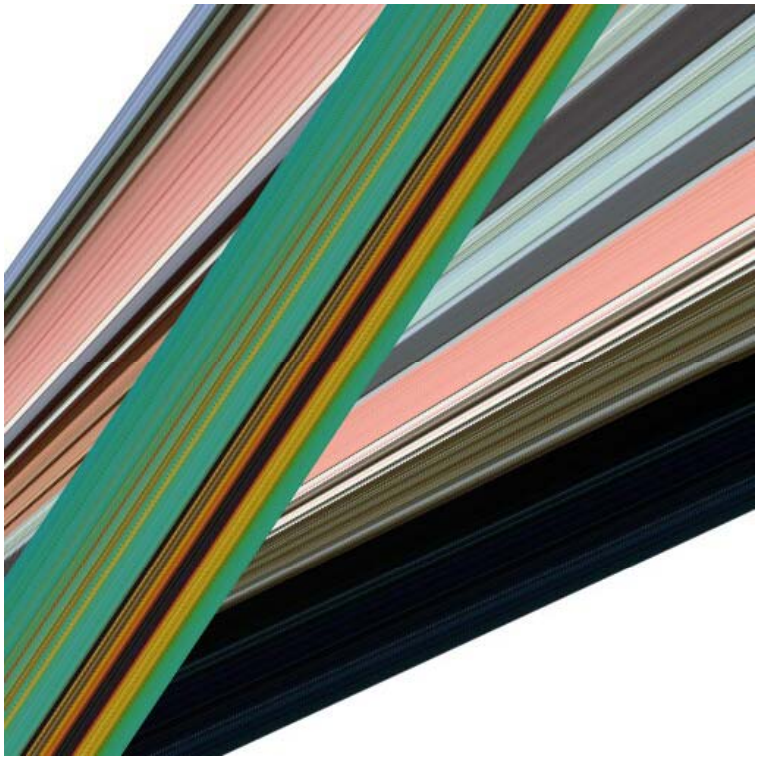


Fig. 7. Combining multiple baseline stereo pairs.

Epipolar-Plane Images [Bolles 87]

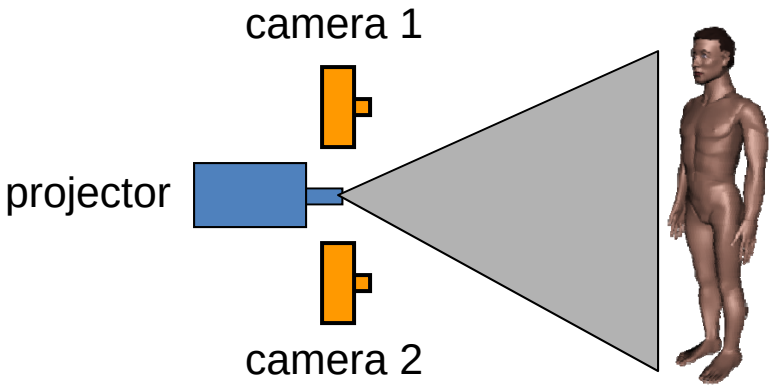
- http://www.graphics.lcs.mit.edu/~aisaksen/projects/drlf/epi/

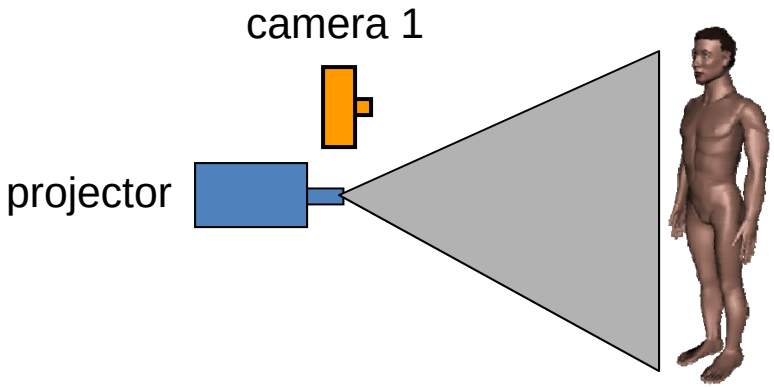


Lesson: Beware of ***occlusions***

Active stereo with structured light

Li Zhang's one-shot stereo



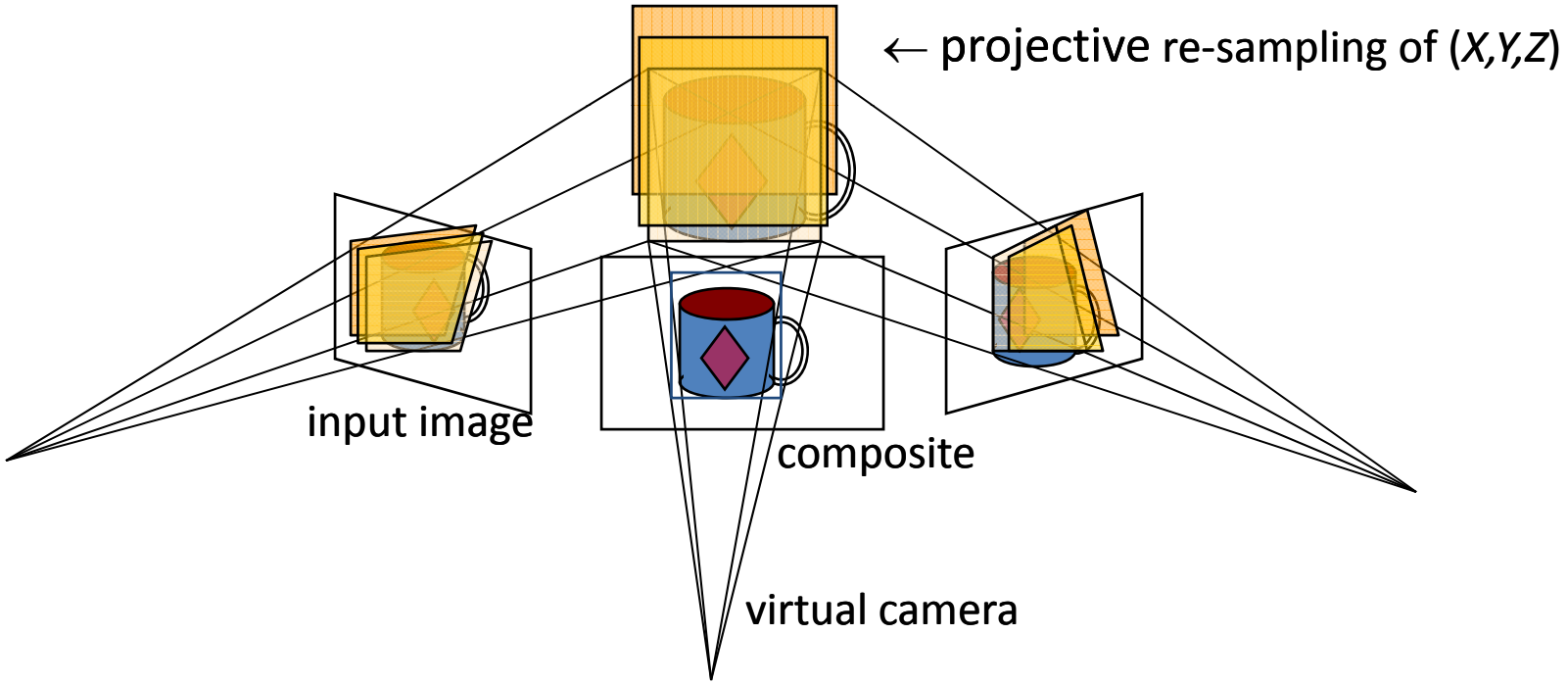


- Project "structured" light patterns onto the object
  - simplifies the correspondence problem

Szeliski

Plane Sweep Stereo

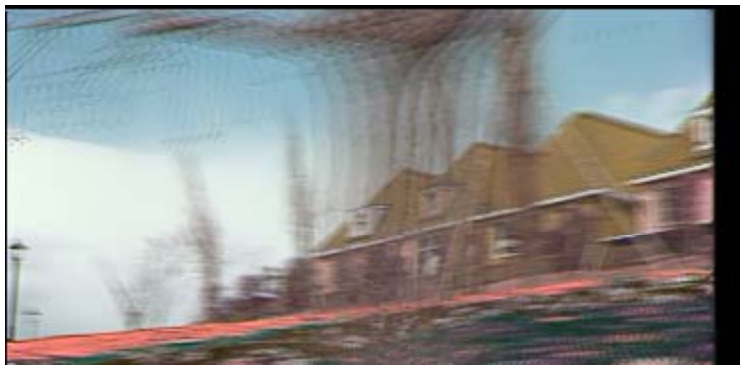
- Sweep family of planes through volume



- each plane defines an image ⇒ composite homography

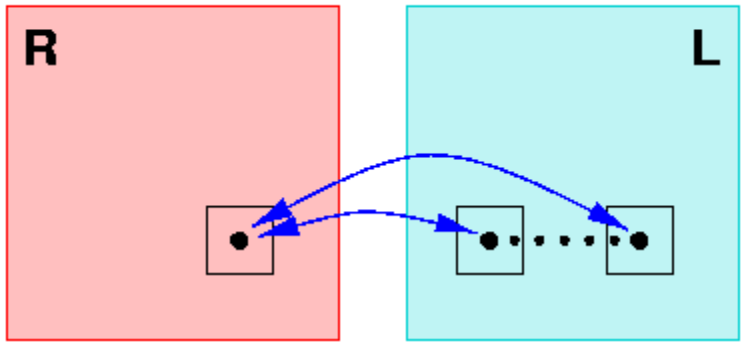
Plane Sweep Stereo

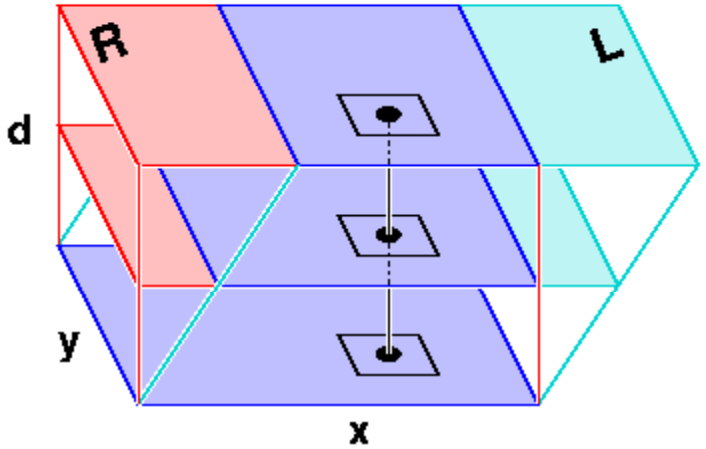
- For each depth plane
  - compute composite (mosaic) image — *mean*



  - compute error image — *variance*
  - convert to confidence and aggregate spatially
- Select winning depth at each pixel

Plane sweep stereo

- Re-order (pixel / disparity) evaluation loops
- 



for every pixel,
for every disparity
compute cost

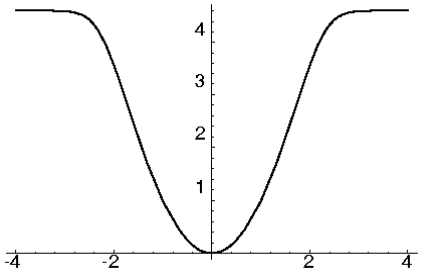
for every disparity
for every pixel
compute cost

framework

1. For every disparity, compute *raw* matching costs

$$E_0(x, y; d) = \rho(I_L(x' + d, y') - I_R(x', y'))$$

   Why use a robust function?
   – occlusions, other outliers

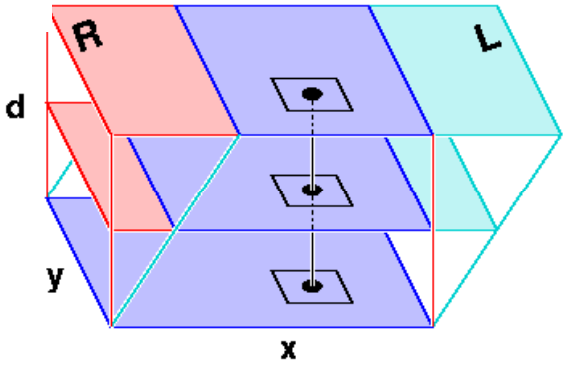


- Can also use alternative match criteria

framework

2. Aggregate costs spatially

$$E(x, y; d) = \sum_{(x',y') \in N(x,y)} E_0(x', y', d)$$



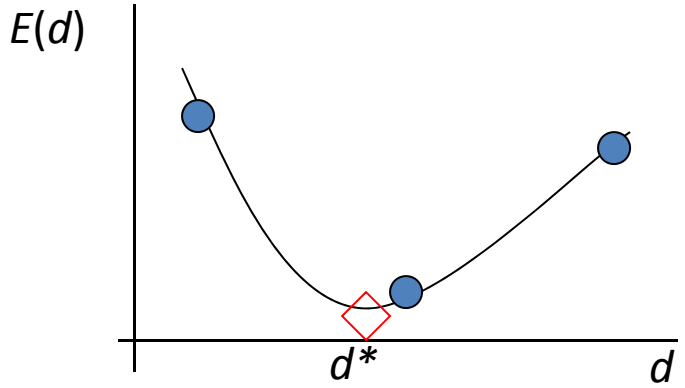
- Here, we are using a *box filter* (efficient moving average implementation)
- Can also use weighted average, [non-linear] diffusion…

framework

3. Choose winning disparity at each pixel

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

4. Interpolate to *sub-pixel* accuracy



Szeliski

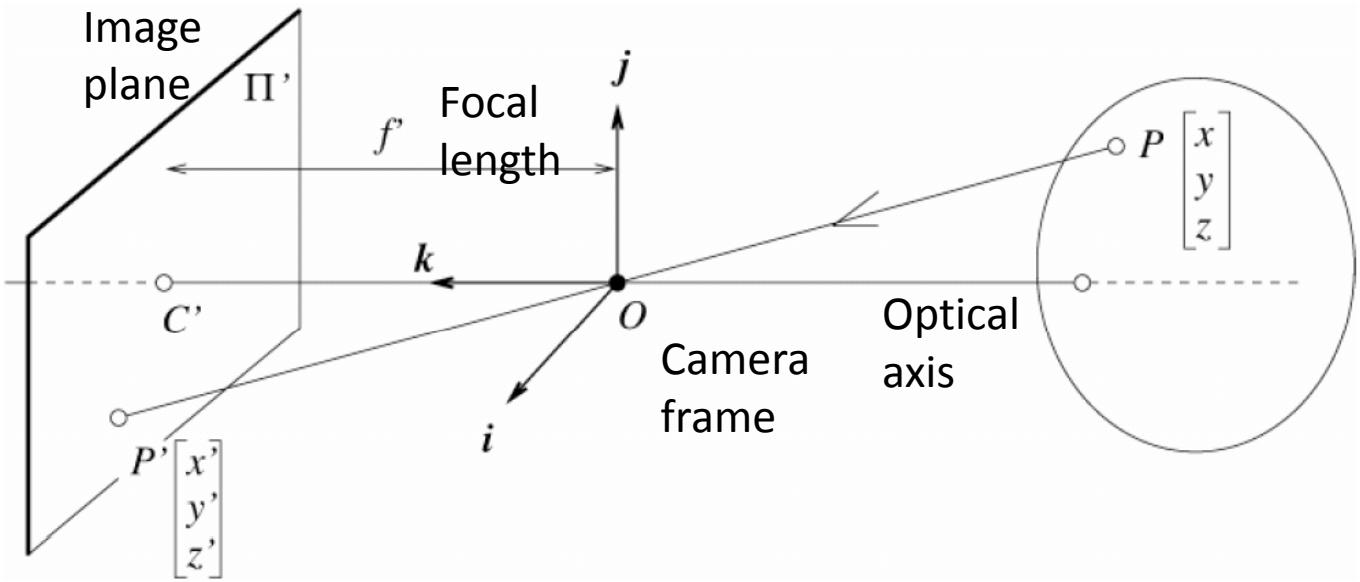
Traditional Analysis

- Advantages:
  - gives detailed surface estimates
  - fast algorithms based on moving averages
  - sub-pixel disparity estimates and confidence
- Limitations:
  - narrow baseline $\Rightarrow$ noisy estimates
  - fails in textureless areas
  - gets confused near occlusion boundaries

Uncalibrated case

- What if we don’t know the camera parameters?

Review: Perspective projection



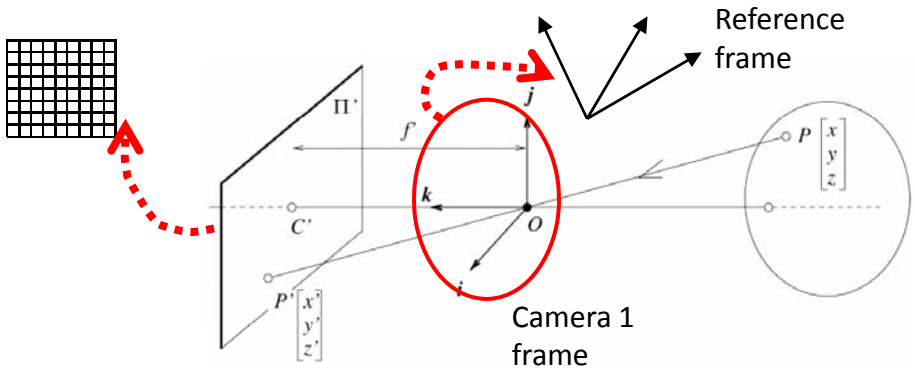
$$(x, y, z) \rightarrow (f\frac{x}{z}, f\frac{y}{z})$$

Scene point $\rightarrow$ Image coordinates

Thus far, in camera's reference frame only.

Review: Camera parameters

- **Extrinsic: location and orientation of camera frame with respect to reference frame**
- Intrinsic: how to map pixel coordinates to image plane coordinates



Review: Extrinsic camera parameters

$$\mathbf{P}_c = \mathbf{R}(\mathbf{P}_w - \mathbf{T})$$

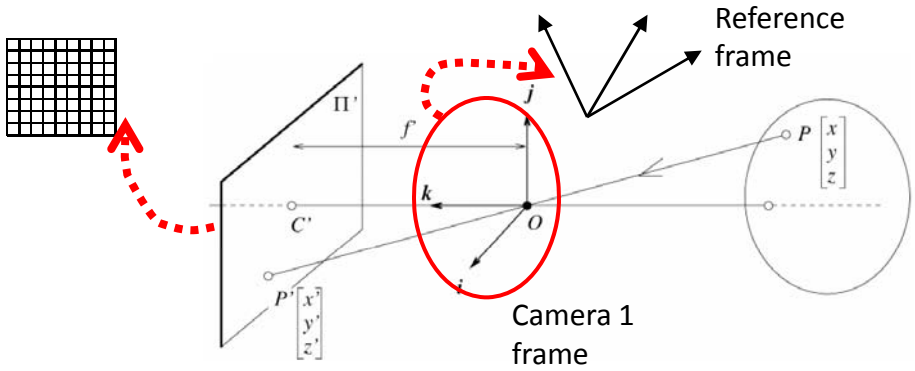
Camera reference frame

World reference frame

$$\mathbf{P}_c = (X, Y, Z)^T$$

Review: Camera parameters

- Extrinsic: location and orientation of camera frame with respect to reference frame
- **Intrinsic: how to map pixel coordinates to image plane coordinates**



Projection matrix for perspective projection

$$x = f\frac{X}{Z}$$

$$y = f\frac{Y}{Z}$$

From pinhole camera model

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$x = \frac{x'}{z'} \quad y = \frac{y'}{z'}$$

$$x = \frac{fX}{Z} \quad y = \frac{fY}{Z}$$

Same thing, but written in terms of homogeneous coordinates

Review: Intrinsic camera parameters

- Ignoring any geometric distortions from optics, we can describe them by:

$$x = -(x_{im} - o_x)s_x$$

$$y = -(y_{im} - o_y)s_y$$

Coordinates of projected point in camera reference frame

Coordinates of image point in pixel units

Coordinates of image center in pixel units

Effective size of a pixel (mm)

Review: Camera parameters

- We know that in terms of camera reference frame:

$$x = f\frac{X}{Z} \qquad y = f\frac{Y}{Z} \qquad \text{and} \qquad \mathbf{P}_c = \mathbf{R}(\mathbf{P}_w - \mathbf{T})$$
$$\mathbf{P}_c = (X, Y, Z)^T$$

- Substituting previous eqns describing intrinsic and extrinsic parameters, can relate *pixels coordinates* to *world points:*

$$-(x_{im} - o_x)s_x = f\frac{\mathbf{R}_1 \cdot (\mathbf{P}_w - \mathbf{T})}{\mathbf{R}_3 \cdot (\mathbf{P}_w - \mathbf{T})}$$

$$-(y_{im} - o_y)s_y = f\frac{\mathbf{R}_2 \cdot (\mathbf{P}_w - \mathbf{T})}{\mathbf{R}_3 \cdot (\mathbf{P}_w - \mathbf{T})}$$

$\mathbf{R}_i$ = Row i of rotat matrix

Review: Projection matrix

- This can be rewritten as a matrix product using homogeneous coordinates:

point in camera coordinates

$$\begin{pmatrix} wx_{im} \\ wy_{im} \\ w \end{pmatrix} = \mathbf{M}_{int}\,\mathbf{M}_{ext} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

$$x_{im} = wx_{im} / w$$

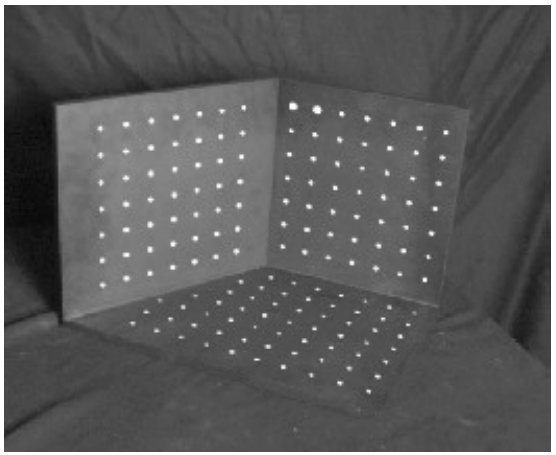
$$y_{im} = wy_{im} / w$$

$$\mathbf{M}_{int} = \begin{pmatrix} -f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{M}_{ext} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & -\mathbf{R}_1{}^{\mathrm{T}}\mathbf{T} \\ r_{21} & r_{22} & r_{23} & -\mathbf{R}_2{}^{\mathrm{T}}\mathbf{T} \\ r_{31} & r_{32} & r_{33} & -\mathbf{R}_3{}^{\mathrm{T}}\mathbf{T} \end{pmatrix}$$

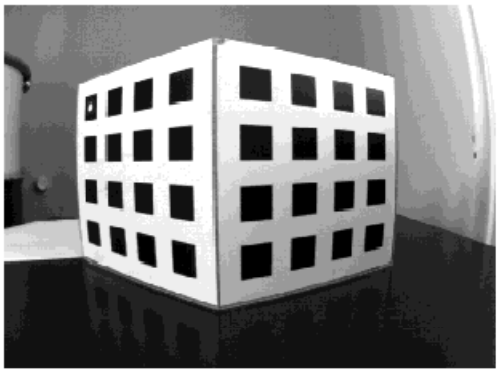
Review: Calibrating a camera

- Compute intrinsic and extrinsic parameters using observed camera data



Main idea

- Place “calibration object” with known geometry in the scene
- Get correspondences
- Solve for mapping from scene to image: estimate $\mathbf{M}=\mathbf{M}_{int}\mathbf{M}_{ext}$



The Opti-CAL Calibration Target Image

Review: Projection matrix

- This can be rewritten as a matrix product using homogeneous coordinates:

$$\begin{pmatrix} wx_{im} \\ wy_{im} \\ w \end{pmatrix} = \underbrace{\mathbf{M}_{int}\,\mathbf{M}_{ext}}_{\mathbf{M}} \overbrace{\begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}}^{\mathbf{P}_w \text{ in homog.}}$$

$$x_{im} = \frac{\mathbf{M}_1 \cdot \mathbf{P}_w}{\mathbf{M}_3 \cdot \mathbf{P}_w} \qquad wx_{im} = \mathbf{M}_1 \cdot \mathbf{P}_w$$

$$y_{im} = \frac{\mathbf{M}_2 \cdot \mathbf{P}_w}{\mathbf{M}_3 \cdot \mathbf{P}_w} \qquad wy_{im} = \mathbf{M}_2 \cdot \mathbf{P}_w \qquad w = \mathbf{M}_3 \cdot \mathbf{P}_w$$

product **M** is single **projection matrix** encoding both extrinsic and intrinsic parameters

Let $\mathbf{M}_i$ be row i of matrix **M**

Review: Estimating the projection matrix

For a given feature point

$$x_{im} = \frac{\mathbf{M}_1 \cdot \mathbf{P}_w}{\mathbf{M}_3 \cdot \mathbf{P}_w} \longrightarrow 0 = (\mathbf{M}_1 - x_{im}\mathbf{M}_3) \cdot \mathbf{P}_w$$

$$y_{im} = \frac{\mathbf{M}_2 \cdot \mathbf{P}_w}{\mathbf{M}_3 \cdot \mathbf{P}_w} \longrightarrow 0 = (\mathbf{M}_2 - y_{im}\mathbf{M}_3) \cdot \mathbf{P}_w$$

Review: Estimating the projection matrix

$$0 = (\mathbf{M}_1 - x_{im}\mathbf{M}_3)\cdot\mathbf{P}_w$$
$$0 = (\mathbf{M}_2 - y_{im}\mathbf{M}_3)\cdot\mathbf{P}_w$$

Expanding this first equation, we have:

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \end{bmatrix} - x_{im}\begin{bmatrix} m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} * \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = 0$$

$$\left(X_w m_{11} - X_w x_{im} m_{31}\right) + \left(Y_w m_{12} - Y_w x_{im} m_{32}\right) \ldots$$
$$\ldots + \left(Z_w m_{13} - Z_w x_{im} m_{33}\right) + \left(m_{14} - x_{im} m_{34}\right) = 0$$

Review: Estimating the projection matrix

$$0 = (\mathbf{M}_1 - x_{im}\mathbf{M}_3)\cdot\mathbf{P}_w$$
$$0 = (\mathbf{M}_2 - y_{im}\mathbf{M}_3)\cdot\mathbf{P}_w$$

$$\begin{pmatrix} X_w & Y_w & Z_w & 1 & 0 & 0 & 0 & 0 & -x_{im}X_w & -x_{im}Y_w & -x_{im}Z_w & -x_{im} \\ 0 & 0 & 0 & 0 & X_w & Y_w & Z_w & 1 & -y_{im}X_w & -y_{im}Y_w & -y_{im}Z_w & -y_{im} \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Review: Estimating the projection matrix

This is true for every feature point, so we can stack up *n* observed image features and their associated 3d points in single equation:

Pm = 0

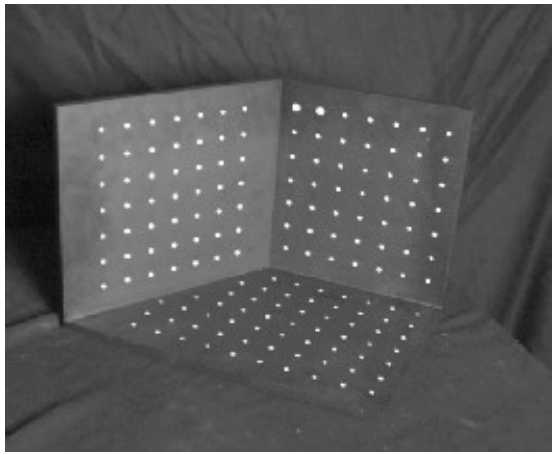
P **m**

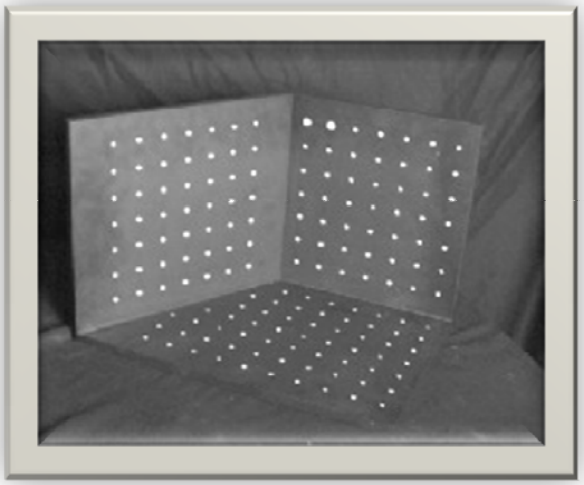
$$\begin{bmatrix} X_w^{(1)} & Y_w^{(1)} & Z_w^{(1)} & 1 & 0 & 0 & 0 & 0 & -x_{im}^{(1)}X_w^{(1)} & -x_{im}^{(1)}Y_w^{(1)} & -x_{im}^{(1)}Z_w^{(1)} & -x_{im}^{(1)} \\ 0 & 0 & 0 & 0 & X_w^{(1)} & Y_w^{(1)} & Z_w^{(1)} & 1 & -y_{im}^{(1)}X_w^{(1)} & -y_{im}^{(1)}Y_w^{(1)} & -y_{im}^{(1)}Z_w^{(1)} & -y_{im}^{(1)} \end{bmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Solve for m_{ij}'s (the calibration information) [F&P Section 3.1]

Summary: camera calibration

- Associate image points with scene points on object with known geometry
- Use together with perspective projection relationship to estimate projection matrix
- (Can also solve for explicit parameters themselves)





When would we calibrate this way?

- Makes sense when geometry of system is not going to change over time

- …When would it change?

Weak calibration

- Want to estimate world geometry without requiring calibrated cameras
  - Archival videos
  - Photos from multiple unrelated users
  - Dynamic camera system

- Main idea:
  - 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

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

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

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

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

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

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.

Stereo pipeline with weak calibration

So, where to start with uncalibrated cameras?

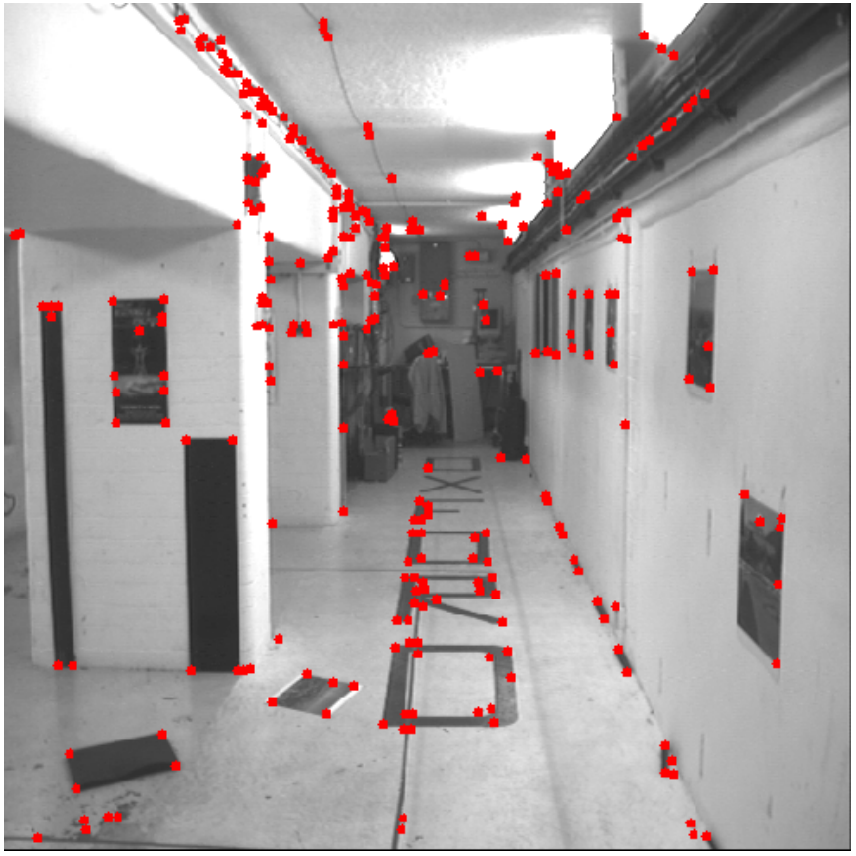
Need to find fundamental matrix F **and** the correspondences (pairs of points (u',v') ↔ (u,v)).

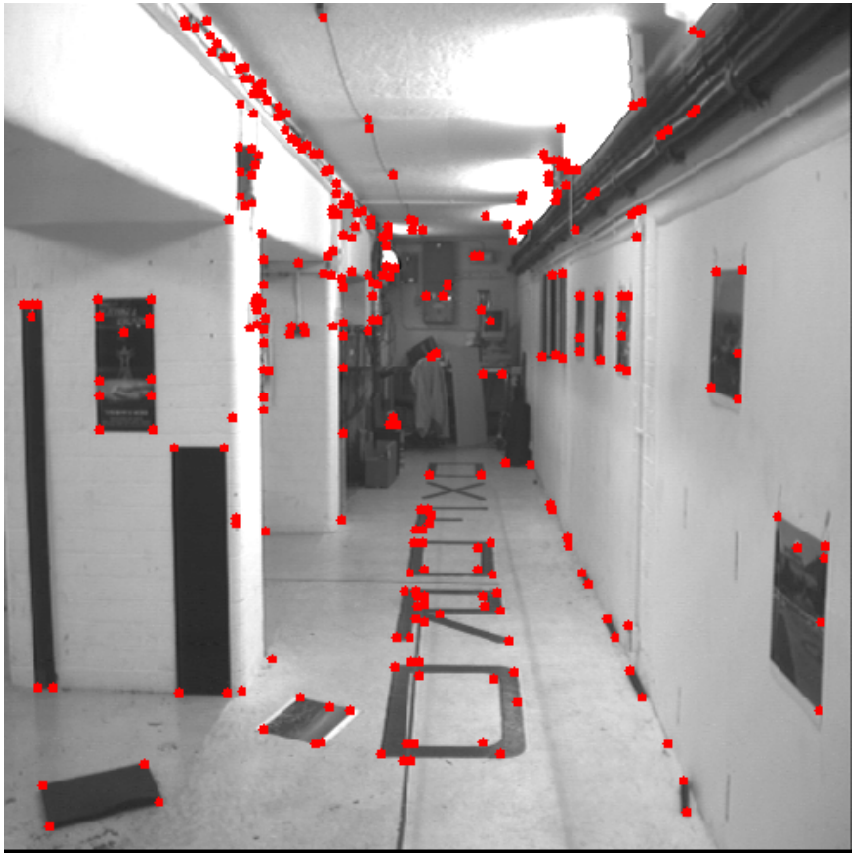
1) Find interest points in image (more on this later)
2) Compute correspondences
3) Compute epipolar geometry
4) Refine

Example from Andrew Zisserman

Stereo pipeline with weak calibration

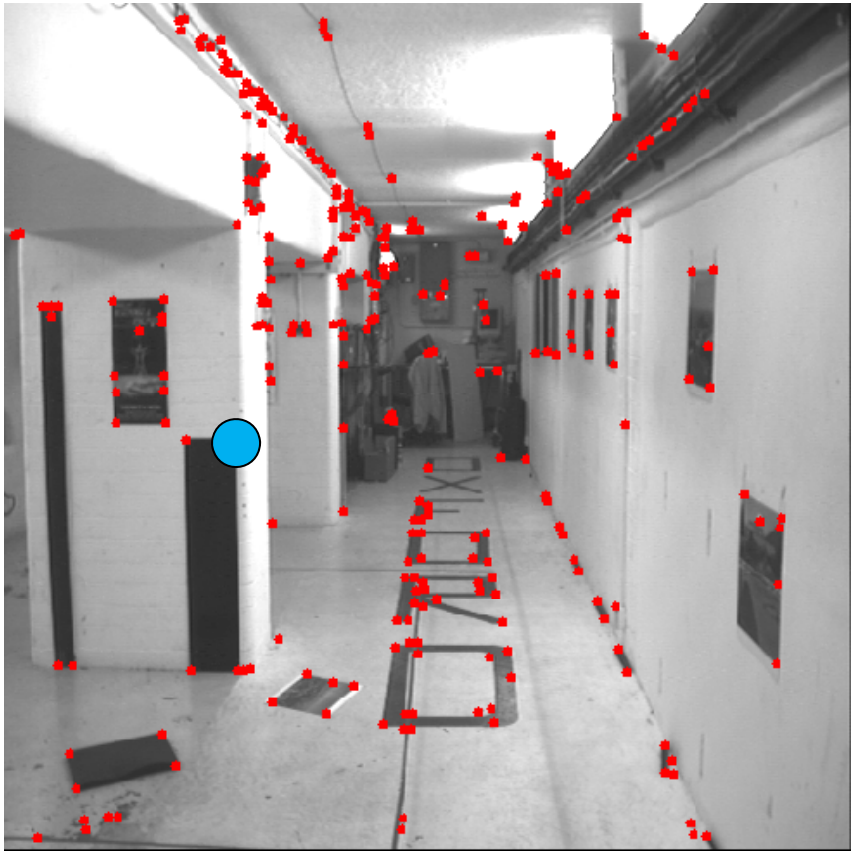
1) Find interest points

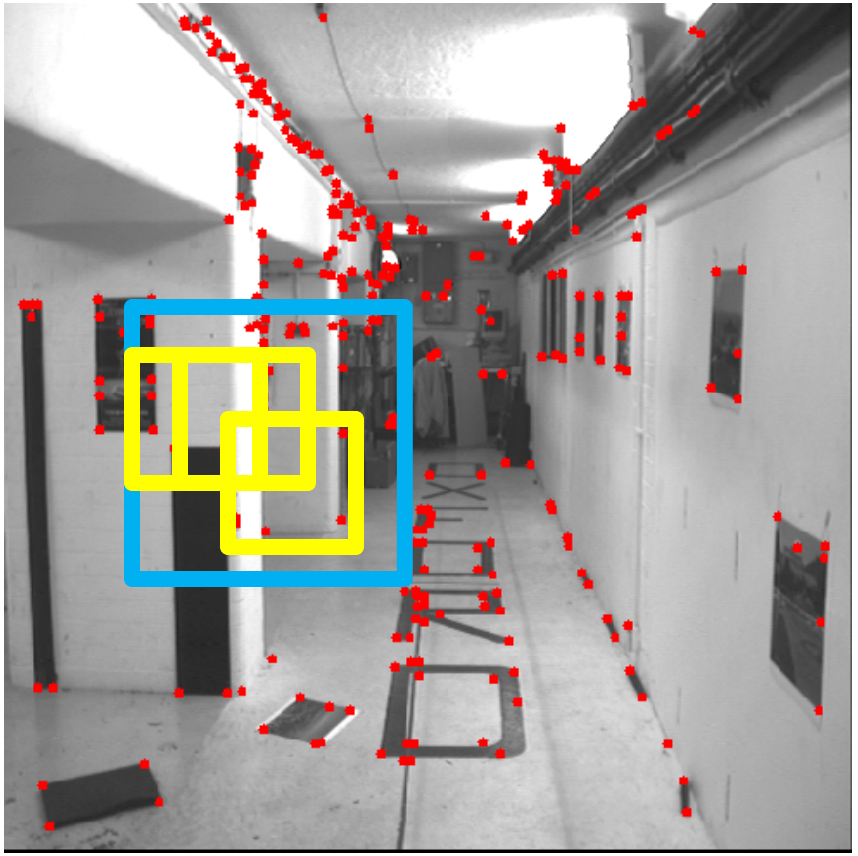




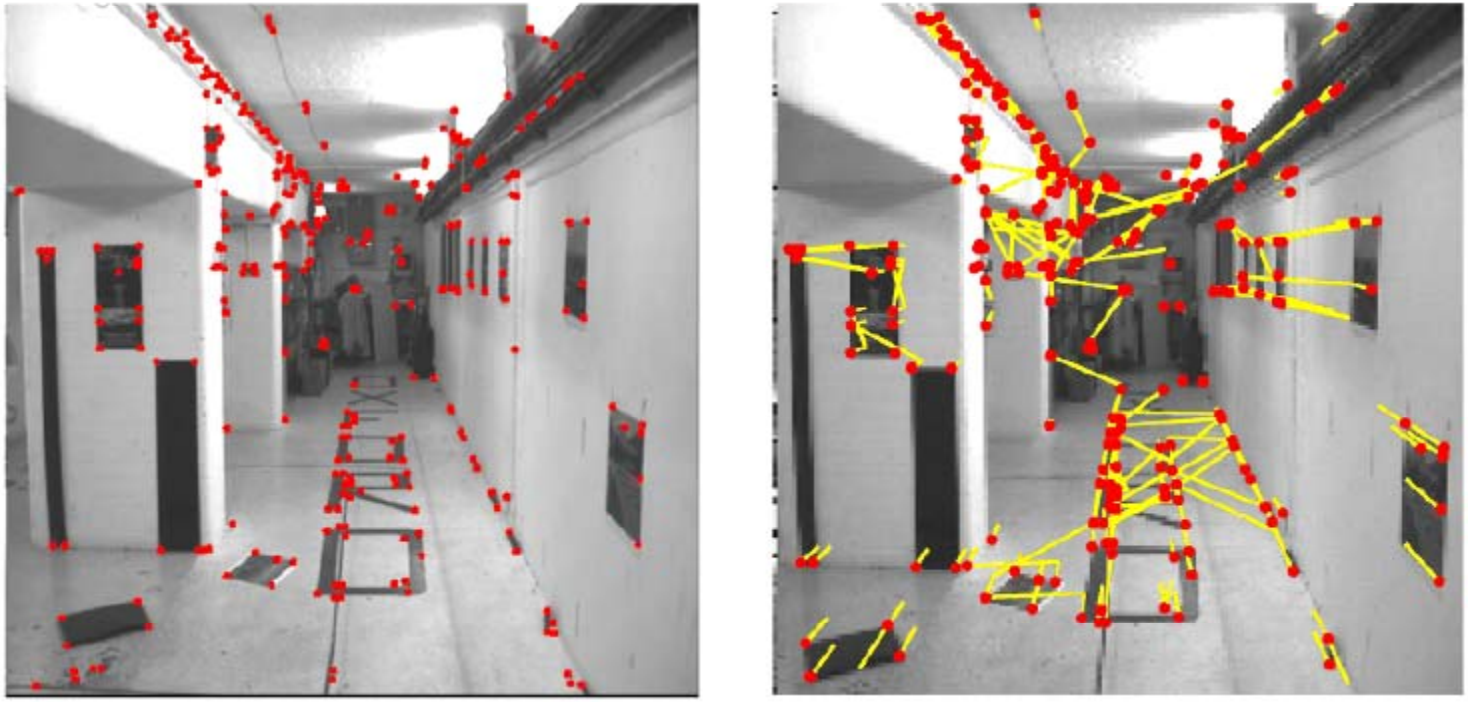
Stereo pipeline with weak calibration

2) Match points only using proximity





Putative matches based on correlation search



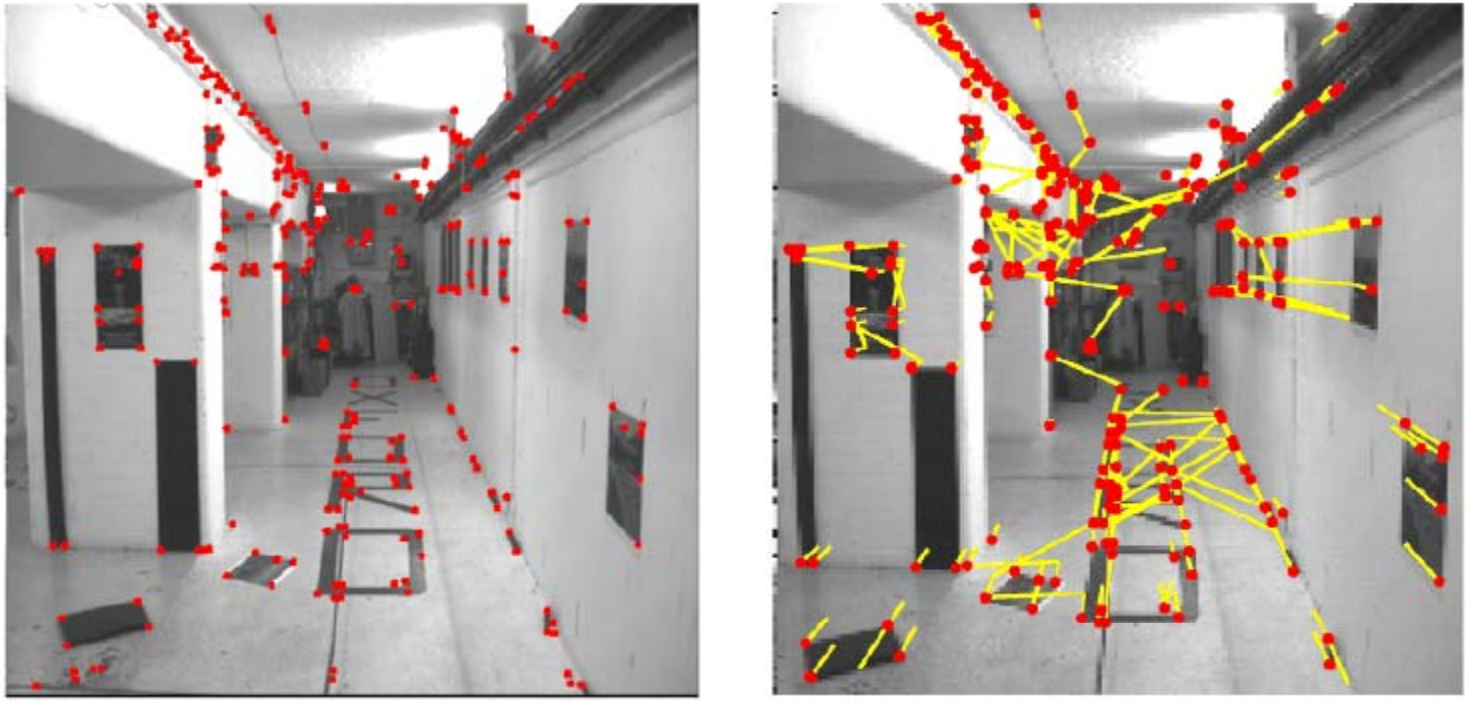
- Many wrong matches (10-50%), but enough to compute F

RANSAC for robust estimation of the fundamental matrix

- Select random sample of correspondences
- Compute F using them
  - This determines epipolar constraint
- Evaluate amount of support – inliers within threshold distance of epipolar line

- Choose F with most support (inliers)

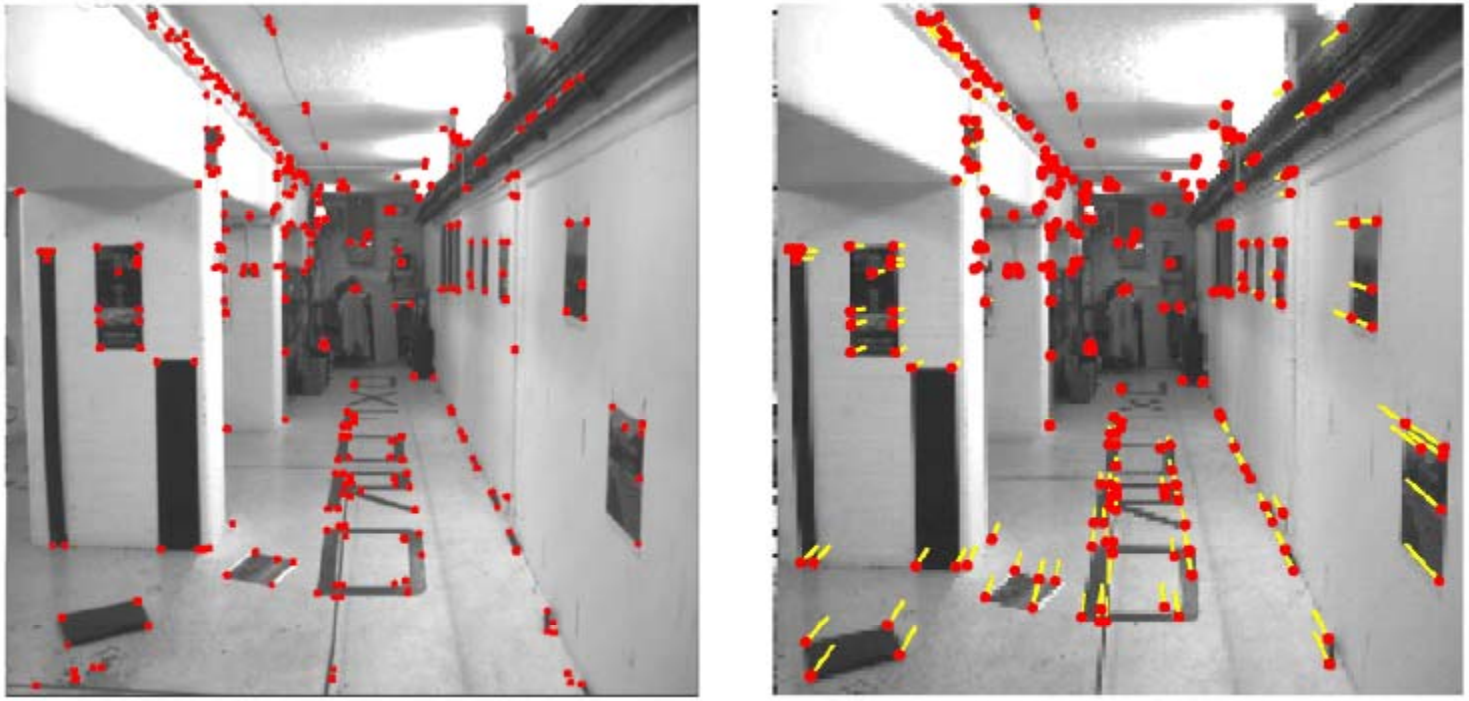
Putative matches based on correlation search



- Many wrong matches (10-50%), but enough to compute F

Pruned matches

- Correspondences consistent with epipolar geometry



- Resulting epipolar geometry

Slide Credits

- Kristen Grauman for most,
- Rick Szeliski and others as noted…

Today: Stereo

- Human stereopsis & stereograms
- Epipolar geometry and the epipolar constraint
  - Case example with parallel optical axes
  - General case with calibrated cameras
- Correspondence search
- The Essential and the Fundamental Matrix
- Multi-view stereo

Roadmap

- Previous: Image formation, filtering, local features, (Texture)…
- Tues: Feature-based Alignment
  - Stitching images together
  - Homographies, RANSAC, Warping, Blending
  - Global alignment of planar models
- **Today: Dense Motion Models**
  - **Local motion / feature displacement**
  - **Parametric optic flow**
- No classes next week: ICCV conference
- Oct 6th: Stereo / 'Multi-view': Estimating depth with known inter-camera pose
- Oct 8th: 'Structure-from-motion': Estimation of pose and 3D structure
  - Factorization approaches
  - Global alignment with 3D point models